Document Title	Requirements on Mode Management
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	069
Document Classification	Auxiliary

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

Document Change History

Date	Version	Changed by	Change Description
02.11.2011	2.1.0	AUTOSAR Administration	• Extension of BswM in order to implement the mode management relevant parts of the Partial Networks concept. • Extension of ComM in order to implement the communication mode management relevant parts of the Partial Networks concept.
30.11.2009	2.0.0	AUTOSAR Administration	• The new module BswM (BSW Mode Manager). • The EcuM-Flex (ECU State Manager Flexible) which has freely configurable states. o New functionality within the EcuM-Flex to support: Multi Core, Alarm Clocks and Defensive Behaviour of BSWMs • Extension of the WdgM (Watchdog Manager) by: o program flow monitoring o windowed watchdogs • Legal disclaimer revised
23.06.2008	1.2.1	AUTOSAR Administration	Legal disclaimer revised

Document Change History			
Date	**Version**	**Changed by**	**Change Description**
26.11.2007	1.2.0	AUTOSAR Administration	• BSW09088 removed due to introduction of Bus-specific State Managers below Communication Manager • BSW09130 and BSW09131 removed due to direct initialization of the communication stack by the ECU State Manager • BSW09170 and BSW09171 added for alive supervision during startup, shutdown, and sleep • BSW09172 added for clarification of Communication Manager behavior • BSW09173 added for clarification of ECU State Manager behavior • Rephrased multiple requirements to clarify behavior and configuration classes • Document meta information extended • Small layout adaptations made
31.01.2007	1.1.0	AUTOSAR Administration	• BSW09075 removed (moved to SRS General) • New requirements BSW09168 and BSW09169 created • References to OSEK OS replaced with references to AUTOSAR OS • Formal adjustments and glossary update • Legal disclaimer revised • Release Notes added • “Advice for users” revised • “Revision Information” added
08.05.2006	1.0.0	AUTOSAR Administration	Initial Release

Disclaimer

Advice for users

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

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

Table of Contents

1 Scope of Document

The goal of this document is to define the functional and non-functional requirements for all modules of the AUTOSAR mode management:

1. ECU State Manager (EcuM) which manages startup and shutdown of the ECU. This includes triggering the shutdown when all requests for run are released;
2. Watchdog Manager (WdgM) which collects the indications of aliveness and correct execution order from the independent applications and relays them to the hardware watchdog in a suitable way;
3. Communication Manager (ComM) which coordinates the communication of the independent applications.
4. Basic Software Mode Manager (BswM) which organizes mode handling and mode related interaction of SW-Cs and the BSW modules.

All modules are needed if more than one independent software component resides on the ECU.
The location of these modules within the whole AUTOSAR ECU SW Architecture is defined in [6].

2 Conventions to be used

- In requirements, the following specific semantics shall be used (based on the Internet Engineering Task Force IETF).
  The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as:
  - SHALL: This word means that the definition is an absolute requirement of the specification.
  - SHALL NOT: This phrase means that the definition is an absolute prohibition of the specification.
  - MUST: This word means that the definition is an absolute requirement of the specification due to legal issues.
  - MUST NOT: This phrase means that the definition is an absolute prohibition of the specification due to legal constraints.
  - SHOULD: This word, or the adjective "RECOMMENDED", mean that there may exist valid reasons in particular circumstances to ignore a particular item, but the full implications must be understood and carefully weighed before choosing a different course.
  - SHOULD NOT: This phrase, or the phrase "NOT RECOMMENDED" mean that there may exist valid reasons in particular circumstances when the particular behavior is acceptable or even useful, but the full implications should be understood and the case carefully weighed before implementing any behavior described with this label.
  - MAY: This word, or the adjective "OPTIONAL", means that an item is truly optional. One vendor may choose to include the item because a particular marketplace requires it or because the vendor feels that it enhances the product while another vendor may omit the same item. An implementation, which does not include a particular option, MUST be prepared to interoperate with another implementation, which does include the option, though perhaps with reduced functionality. In the same vein an implementation, which does include a particular option, MUST be prepared to interoperate with another implementation, which does not include the option (except, of course, for the feature the option provides.)

3 Terminology

3.1 Glossary

Term:	***Description:***
Active wake-up	Wake-up caused by the ECU e.g. by a sensor.
Alive indication	Indication that a supervised SW-C is alive – meaning active – provided from the SW-C itself to the Watchdog Manager.
Application	Applications are used synonymous to a SW-C. A SW-C may span several atomic SW-Cs. AUTOSAR provides the element of a "composition" to formally group several atomic SW-Cs of an application. A composition is used to structure the description and does not affect the generated code. Note: The atomic SW-Cs in a composition can still be mapped to different ECUs. Application Modes are therefore 'local' to an application but can be 'global' to several ECUs.
Application Mode	An Application Mode is a ***mode*** that is not controlled and standardized by the Mode Managers in the BSW. The scope of an Application Mode is a limited number of SW-Cs that belongs to a logical Application. If a ***mode*** only affects one composition, it is an Application Mode. An Application Mode may be distributed across multiple ECUs or may be local to one ECU depending on the distribution of the SW-C belonging to that application. Examples: Normal Operation, Limp Home
Application Mode Manager	An Application Mode Manager is a SW-C therefore a priori not standardized. It collects environmental data and ***Mode Requests*** via application-specific (non-standardized) interfaces from other SW-Cs. It can switch ***Application Modes*** (that are not controlled and standardized by the ***Mode Managers*** in the BSW). It can also request ***modes*** from other ***Mode Managers***, specifically from ***Mode Managers*** in the BSW.
BSW Mode	A BSW Mode is ***mode***, which is controlled and standardized by the Mode Manager in the BSW. A BSW Mode is always local to one ECU. Examples: EcuM Modes, ComM Modes
BSW Mode Manager	The BSW Mode Manager is a BSW module that collects ***mode requests*** via a standardized interface and controls functionality of other BSW modules accordingly. Examples are: LIN Schedule Table Switching, Enabling/Disabling I-PDU Groups, …
Communication mode	Related to a physical channel or to a ***user***, it indicates if it is possible to send/receive, only receive, neither send nor receive.
Communication request	A communication request indicates a demand for communication from a given ***user*** (e.g. a runnable requiring an operational communication stack). However, it can neither be assumed that the request will be fulfilled within a certain time nor that it will be fulfilled at all. The demander itself has to make sure that communication is indeed established, by using query functions or callbacks.
ECU state	General term to indicate the states managed by the ECU State Manager. They represent a structured model that extends the ***power modes*** of the ECU with states and transitions to support necessary activities of software to enter/leave these ***power modes***.
Inoperation	An artificial word to describe the ECU when it is not operational, i.e. not running. It comprises all meanings of *off, sleeping, frozen*, etc. Using this definition is beneficial since it has no predefined meaning.
Low-power mode	All ***power modes*** except "on" (full power).

Mode	A mode is a certain set of states of the various state machines that are running in the vehicle that are relevant to a particular entity, e.g. a SW-C, a BSW module, an application, a whole vehicle In its lifetime, an entity changes between a set of mutually exclusive modes. These changes are triggered by environmental data, e.g. signal reception, operation invocation.
Mode Declaration Group	A Mode Declaration Group is defined in the Software Component Template and implemented by the RTE. A Mode Declaration Group defines a number of mutually exclusive ***modes***. There is no hierarchy of ***modes*** within a Mode Declaration Group. Each Mode Declaration Group describes only one aspect of the whole system. All ***modes*** of all Mode Declaration Groups on an ECU describe the abstract ***mode*** of the ECU. Similarly, all ***modes*** of all Mode Declaration Groups in a system describe the abstract ***mode*** of all ECUs in the system. (The ***mode*** is abstract because it cannot be physically accessed, it just exists conceptually). There is a standardized set of Mode Declaration Groups defined in the BSW, e.g. ComM, WdgM, EcuM. Each system designer is free to extend the number of Mode Declaration Groups and define his/her own ***modes*** in there. Only one Mode Manager is allowed to switch the mode of an instance of a Mode Declaration Group. (Limitation in VFB)
Mode Manager	A Mode Manager is a role that can be taken by either a BSW module (and optionally additionally by an SW-C). The Mode Manager collects requests from ***Mode Requesters***, arbitrates them and switches the ***mode*** of its ***Mode Declaration Group***(s) accordingly. Examples of Mode Managers are: EcuM, ComM, WdgM and BswM Note: If a SW-C Mode Manager switches ***modes*** directly, the mode arbitration in the ***BSW Mode Manager*** cannot resolve conflicts. Thus, it is recommended to use multi-level mode arbitration, of SW-C mode Manager and BSW Mode Manager
Mode Port	A Mode Port is port that has a Sender-Receiver Interface which contains a **Mode Declaration Group**. Note: This is called Mode Port in SWS RTE -> To distinguish it from ***Mode Request Ports***, we should call it Mode Switch Port
Mode Request	A Mode Request is some information communicated to a ***Mode Manager*** that requests the ***Mode Manager*** to switch to a certain ***mode***. For ***Mode Managers*** in the BSW the interface to send a Mode Request is a standardized client-server interface defined by the corresponding ***Mode Manager***. Examples: Client-server interface ComM_UserRequest with operation RequestComMode(…). For ***Application Mode Managers*** the interface can be any client-server or sender-receiver interface defined by the ***Mode Manager***.
Mode Request Port	A Mode Request Port is a port with the special semantics of requesting a ***mode*** from a ***Mode Manager***. It can be a Client-Server or a Sender-Receiver Port. For **Mode Managers** in the BSW the Mode Request Port is standardized. Note: For EcuM, ComM and WdgM it is always a Provided Client-Server Port. For BswM it is a Required Sender-Receiver Ports.
Mode Requester	A Mode Requester is an entity that requests ***modes*** from a ***Mode Manager***.
Mode Switch Port	Favored replacement of phrase ***Mode Port***
Mode User	A Mode User is an entity that reacts on ***mode*** changes.
OFF state	***ECU state***. It denotes the unpowered ECU. Depending on the hardware design, the ECU can or cannot react to passive wake-up in this state (wakeability is not required in this state).

Passive wake-up	Wakeup initiated by another ECU and propagated (e.g. by bus or wakeup-line) to the ECU currently in focus.
Port Group	A Port Group is a logical group of ports that are needed to realize the same functionality in the ***Application***. A port can be a member of multiple Port Groups. A Port Group can be used to request all associated communication resources and to inquire their state. Thereby, the Port Group also defines a mapping between Application Ports and Communication Resources. Thus, the purpose of a port group is to have an abstraction of the required communication resources of that functionality For example, an ***Application*** contains normal-operation functionality and limp-home functionality with reduced communication requirements. Then it is useful to define two Port Groups, A and B. A contains the ports that are necessary for the normal-operation functionality and B contains the ports for the limp-home functionality. Thereby, the ***Application*** can recognize the condition when the communication resources for normal operation are unavailable but are sufficient for limp home. Port Groups are defined on SW-C composition level, which means that they may be distributed on several ECUs. The requests and indications for the Port Groups shall however only affect the portion of the Port Group that is local to an ECU. If distributed control of Port Groups is needed it can be handled on “Application Mode Management” level (above the RTE) using normal communication features.
Power mode	Hardware power mode of the ECU. Typically: on (full power), off, sleep, standby, etc… The last two items can take several forms depending on hardware capabilities (reduced clock, peripheral standby, etc.).
Program flow monitoring	Technique to detect errors that cause a divergence from the valid program sequence seen during valid execution of a program.
RUN state	***ECU state***. The ECU is fully functional, all BSW modules are initialized and application software components are able to run.
Shutdown target	The chosen low-power state (OFF, SLEEP) for the next shutdown. If the ***SLEEP state*** supports several ***sleep modes***, the shutdown target shall indicate the chosen ***sleep mode***.
Sleep mode	The term "mode" is related to the current availability of the ECU's capabilities. “Sleep mode” is the overall abstracted term for a variety of ***low-power modes*** that could exist at different CPU's, which have in common that they currently do not perform code-execution, but are still powered.
SLEEP state	***ECU state***. It is an energy saving state: no code is executed but power is still supplied, and if configured correctly, the ECU is wakeable. The SLEEP state provides a configurable set of ***sleep modes*** which typically are a trade off between power consumption and time to restart the ECU.
State/communication requestor	See ***User***.
Supervised Entity	A **supervised entity** is a software entity which is included in the monitoring of the Watchdog Manager. Each **supervised entity** has exactly one identifier. A **supervised entity** denotes a collection of **checkpoints** within a software component or basic software module. There may be zero, one or more **supervised entities** in a software component or basic software module.
User	Concept for requestors of the ECU State Manager and of the Communication Manager. A user may be a runnable entity, a SW-C/BSW or even a group of SW-Cs/BSWs, which acts as a single unit towards the ECU State Manager and the Communication Manager.

Vehicle Mode	A Vehicle Mode is a ***mode*** that is not controlled and standardized by the Mode Managers in the BSW. The scope of a Vehicle Mode is the whole vehicle. If a ***mode*** affects multiple compositions, it is a Vehicle Mode. A Vehicle Mode is by definition distributed across the whole vehicle. Example: Transport Mode, Power Saving Mode, Ignition On, Ignition Off
Vehicle Mode Manager	A Vehicle Mode Manager is a special kind of ***Application Mode Manager*** that switches ***Vehicle Modes***.
Wake-up event	A physical event which causes a wake-up. A CAN message or a toggling IO line can be wake-up events. Similarly, the internal SW representation, e.g. an interrupt, may also be called a wake-up event.
Wake-up reason	The ***wake-up event*** being actually the cause of the current/last wake-up.
Wake-up source	The peripheral or ECU component which deals with ***wake-up events*** is called a wake-up source.

3.2 Abbreviations

Abbreviation:	***Description:***
BSW	Basic Software
BswM	Basic Software Mode Manager
ComM	Communication Manager
DCM	Diagnostic Communication Manager
DEM	Diagnostic Event Manager
EcuM	ECU State Manager
FiM	Function Inhibition Manager
RE	Runnable Entity
SW-C	Software Component
WdgM	Watchdog Manager

4 Requirement Specification

The Mode Management cluster is in charge of four Basic Software modules:

- → the ECU State Manager, controlling the startup phase of AUTOSAR BSW modules including startup of the OS;
- → the Communication Manager, in charge of the network resource management;
- → the Watchdog Manager, responsible for triggering the watchdog based on the aliveness status and control flow status of application software.
- → the Basic Software Mode Manager, responsible for supporting the mode handling.

In the following, requirements will be assigned to each of these modules.

4.1 ECU State Manager (EcuM)

The ECU State Manager is a basic software module that manages the ECU states and the transitions between these states.
It manages all wake-up events and configures the ECU for SLEEP when requested.

The ECU State Manager shall support individual preparation-activities and transitions to bring up the ECU or put it into a decreased power mode (low-power mode, e.g. sleep / standby). By a well defined usage of ECU State Manager features and capabilities, this module can then be used to perform a pre-defined strategy of power-consumption, thus allowing efficient energy management for the ECU.

4.1.1 Common

4.1.1.1 Functional Requirements

4.1.1.1.1 Configuration (Common)

4.1.1.1.1.1 [BSW09122] Configuration of users of the ECU State Manager

ID:	BSW09122
Initiator:	VW
Date:	16.12.2004
Short Description	Configuration of users of the ECU State Manager
Type:	--
Importance:	High
Description:	**SRS1207**:Users, indicating the operational needs of individual software components, shall be pre compile time configurable attributes of the ECU State Manager. Requests from unknown sources are ignored.
Rationale:	All users are known and configured at compile-time. This requirement supports static administration of (known) state requestors on the ECU; it supports also testing and documentation.
Use Case:	--
Dependencies:	Requesting and releasing states.
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.1.1.1.1.2 [BSW09100] Selection of wake-up sources shall be configurable

ID:	BSW09100
Initiator:	WP Mode Management
Date:	13.12.2004
Short Description:	Selection of wake-up sources shall be configurable
Type:	--
Importance:	High
Description:	**SRS1208**: It shall be possible to configure in pre-compile time which wake-up sources are relevant in which sleep mode, provided that they can be actually assigned to a sleep mode (i.e. they will be enabled when the ECU enters that particular sleep mode).
Rationale:	Wake-up sources cannot be standardized in detail. Their mapping to use cases cannot be standardized neither. Therefore this flexibility is needed.
Use Case:	Depending on the ECU, the allowed wake-up source may vary: – reception of a CAN frame, – direct input(s), – either CAN frames or direct inputs, – etc.

	Which wake-up source has to be activated in which sleep mode, is then a configuration parameter.
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.1.1.1.2 Normal Operation (Common)

4.1.1.1.2.1 [BSW09104] ECU State Manager shall take over control after OS shutdown

ID:	BSW09104
Initiator:	WP Mode Management
Date:	13.12.2004
Short Description:	ECU State Manager shall take over control after OS shutdown
Type:	--
Importance:	High
Description:	**SRS1213**: After OS shutdown, additional shutdown activities shall be taken like preparing hardware for the next wake-up. Finally the ECU has to be switched off or put into a sleep mode. This shall be the task of the ECU State Manager.
Rationale:	- Provide services after OS-shutdown. - Startup/shutdown symmetry.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.1.1.1.2.2 [BSW09128] Support of several shutdown targets

ID:	BSW09128
Initiator:	WP Mode Management
Date:	03.03.2005 / 28.06.2005 (rephrased)
Short Description:	Support of several shutdown targets
Type:	--
Importance:	High
Description:	**SRS1216**: The ECU State Manager shall offer the following targets for shutting down the ECU: Power Off Reset of ECU Sleep Modes The default target shall be defined by configuration. It can be overridden by an API-service. Several Sleep Modes may be supported (BSW09119)
Rationale:	Allow trade off between energy consumption and time to wake-up/startup the ECU.
Use Case:	--
Dependencies:	- [BSW09119] Support of several sleep modes. - [BSW09118] Time triggered increased inoperation.
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.1.1.1.2.3 [BSW09119] Support of several sleep modes

ID:	BSW09119
Initiator:	VW
Date:	16.12.2004
Short Description:	Support of several sleep modes
Type:	--
Importance:	Medium
Description:	**SRS1217**: The ECU State Manager shall be able to handle a pre-defined (either pre-compile or link-time configuration) set of sleep modes.
Rationale:	Support portability of system-strategies according to increased inoperation mechanism.
Use Case:	--
Dependencies:	- The hardware has to support several sleep modes. - Interface of MCU-driver.
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.1.1.1.2.4 [BSW09102] API for selecting the sleep mode

ID:	BSW09102
Initiator:	WP Mode Management
Date:	13.12.2004
Short Description:	API for selecting the sleep mode
Type:	--

Importance:	High
Description:	**SRS1218**: The ECU State Manager shall provide an API to select the sleep mode which will be chosen for next ECU shutdown. Whenever a decision to go to sleep is taken (which is independent from the use of this API; see rather BSW09072, BSW09115, BSW09165, BSW09166, BSW09114 and BSW09116), the current active sleep mode selection shall be used to choose the actual target sleep mode of the shutdown process.
Rationale:	Application needs a way to select the next shutdown target. However, the decision to start the shutdown process will be totally independent from the call of this API.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.1.1.1.2.5 [BSW09072] Force ECU shutdown

ID:	BSW09072
Initiator:	Porsche
Date:	27.09.2004
Short Description:	Force ECU shutdown
Type:	--
Importance:	High
Description:	The ECU State Manager shall provide the means (explicit API or procedure) to force starting the shutdown process BSW09114. This feature shall not be available to the application, but only accessible by BSW.
Rationale:	Controlled de-initialization of basic software.
Use Case:	Force ECU to shutdown because it is assumed that this ECU does not work properly. Putting it into defined testing condition.
Dependencies:	[BSW09114] Starting/invoking the shutdown process.
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.1.1.1.2.6 [BSW09017] Provide ECU state information

ID:	BSW09017
Initiator:	Bosch
Date:	12.02.2004
Short Description:	Provide ECU state information
Type:	--
Importance:	high
Description:	**SRS1221**: The ECU State Manager shall provide an API to query the current ECU state.
Rationale:	Basic functionality. SW may behave differently depending on current ECU state.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--

Contributes to:	--

4.1.1.1.2.7 [BSW09136] Centralized wake-up management

ID:	BSW09136
Initiator:	Porsche
Date:	06.04.2005
Short Description:	Centralized wake-up management
Type:	--
Importance:	high
Description:	**SRS1224**: The ECU State Manager shall be the receiver of all wake-up events which occur on its ECU; it shall react appropriately (e.g. ECU wake-up) and propagate the information to other concerned components.
Rationale:	Unambiguous and defined behavior at wake-up events.
Use Case:	--
Dependencies:	- [BSW00375] Notification of wake-up reason (WP Architecture). - [BSW09113] Initialization of Basic Software modules - [BSW09097] Validation of physical channel wake-up
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.1.1.1.2.8 [BSW09098] Storing the wake-up reasons

ID:	BSW09098
Initiator:	WP Mode Management
Date:	13.12.2004
Short Description:	Storing the wake-up reasons
Type:	--
Importance:	High
Description:	**SRS1225:** The ECU State Manager shall be able to store the current wake-up reason.
Rationale:	Storing the wake-up reason in one module and give every module (BSWs and SWCs) the possibility to query the wake-up reason.
Use Case:	Retrieve the last wake-up reason by means of an external diagnostic tool. Some behaviors (Application mode management may be triggered by the wake up
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.1.1.1.2.9 [BSW09097] Validation of physical channel wake-up

ID:	BSW09097
Initiator:	WP Mode Management
Date:	13.12.2004
Short Description:	Validation of physical channel wake-up
Type:	--
Importance:	High
Description:	The ECU State Manager module shall start a timeout (T_wake_up_timeout greater than zero, statically configurable) after receiving a

	wake-up indication from the bus driver If a valid message is received before the timeout expires, the timer shall be stopped and the ECU State Manager module indicates a system channel (ComMChannel) wake-up. ECU State Manager shall enable the hardware devices (Controller, Transceiver) of the bus system by an extra callout function and shall poll the bus driver if a valid message is received. Messages shall be received by the bus driver but not forwarded to the upper layer before wakeup validation. The ECU State Manager module shall indicate a system channel wake-up only if the first wake-up indication from the bus driver is confirmed by a subsequent valid message within T_wake_up_timeout. In case no validation is possible, ECU State Manager module shall disable the hardware devices, which have been enabled for wake-up validation, again by an extra callout function. If T_wake_up_timeout is set to 0, validation process is disabled and the wake-up is immediately confirmed.
Rationale:	Avoiding wake-up because of erroneous signals (e.g. spikes). If a wake-up occurs but no succeeding activity can be detected on the physical channel, the wake-up is not considered, in order to save power. Communication items are forwarded only after validation, in order not to forward inconsistent or meaningless signals
Use Case:	--
Dependencies:	Wakeup validation is currently only needed for CAN networks.
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.1.1.1.2.10 [BSW09126] Provide an API for querying the wake-up reason

ID:	BSW09126
Initiator:	BMW
Date:	11.02.2004
Short Description:	Provide an API for querying the wake-up reason
Type:	--
Importance:	High
Description:	The ECU State Manager shall provide an API for querying the wake-up event causing the last ECU wake-up.
Rationale:	For diagnostics purposes.
Use Case:	At initialization, application needs to perform different treatments depending on the wake-up reason.
Dependencies:	[BSW00375] Notification of wake-up reason.
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.1.1.1.2.11 [BSW09101] Provide an API to query the reset reason

ID:	BSW09101
Initiator:	WP Mode Management
Date:	31.01.2006
Short Description:	Provide an API to query the reset reason
Type:	--
Importance:	High
Description:	**SRS1233**: The ECU State Manager shall provide an API to query the reason of the last reset.
Rationale:	This feature is needed for setting up different startup scenarios depending if the reset was intentional or unintentional.
Use Case:	Reset loop detection.
Dependencies:	This feature can only be provided if supported by the hardware.
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.1.1.1.2.12 [BSW09127] De-initialization of Basic Software modules

ID:	BSW09127
Initiator:	WP Mode Management
Date:	03.03.2005
Short Description:	De-Initialization of Basic Software modules
Type:	--
Importance:	High
Description:	**SRS1215**: The ECU State Manager shall de-initialize Basic Software modules where appropriate during the shutdown process.
Rationale:	Coordinated ECU shutdown.
Use Case:	--
Dependencies:	[BSW09147] Configuration of de-initialization process of Basic Software modules.
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.1.1.1.2.13 [BSW09116] Requesting and releasing the RUN state

ID:	BSW09116
Initiator:	VW
Date:	16.12.2004
Short Description	Requesting and releasing the RUN state
Type:	--
Importance:	High
Description:	**SRS1211**: The ECU State Manager shall provide services to request and release the RUN state.
Rationale:	The decision to hold an ECU in the RUN state is always related to the application context, which cannot be standardized. Therefore the ECU State Manager needs to get "user-requests" from application perspective to be

	aware of the current situation. It is recommended that independent applications (SWCs), having independent RUN state needs, shall control their "user request" by usage of the provided interface of the ECU State Manager.
Use Case:	A wake-up event (e.g. door contact) related with a preheating functionality of the catalytic exhaust system occurs on the ECU which covers and controls this feature by software. With this wake-up event, a request to enter the RUN state on this ECU shall be established at the ECU State Manager. As soon as the preheating is finished, the preheating functionality shall release its demand to stay in the RUN state. If no other application software components on this ECU are requesting to stay in the RUN state, the ECU State Manager will further proceed with the shutdown process.
Dependencies:	[BSW09115] Evaluate condition to stay in the RUN state.
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.1.2 Fixed

The ECU State Manager Fixed is a variant of the EcuM which has a fixed set of states: OFF, RUN, SLEEP and transient states: STARTUP, WAKEUP, SHUTDOWN).
In detail, the ECU State Manager Fixed:

- is responsible for the initialization and de-initialization of all basic software modules, including OS and RTE;
- cooperates with the so-called Resource Managers (e.g. the Communication Manager) to shut down the ECU when needed;
- Time triggered increased inoperation[1].

[1] This mechanism is intended to wake-up the ECU after a certain time-period automatically, if no other wake-up reasons have occurred, in order to transfer the ECU into another sleep mode, which may be related to further decreased capabilities and power-consumption.

4.1.2.1 Functional Requirements

4.1.2.1.1 Configuration (EcuM Fixed)

[BSW09120] Configuration of initialization process of Basic Software modules

ID:	BSW09120
Initiator:	VW
Date:	16.12.2004
Short Description	Configuration of initialization process of Basic Software modules
Type:	--
Importance:	High
Description:	The Basic Software modules with their initialization routines and their initialization order shall be pre-compile time configurable attributes of the ECU State Manager.
Rationale:	The ECU State Manager shall have knowledge about the initialization-routines from Basic Software modules and their predecessors and successor relationships. However, it shall be possible to adapt the concrete initialization process of the ECU State Manager to the project specific initialization demands of the Basic Software modules.
Use Case:	System A requires the initialization of the watchdog driver as soon as possible to monitor the system startup whereas in system B the watchdog driver shall be initialized last, in order to avoid resets based on long lasting hardware initializations. Integration of supplier specific driver (complex driver), e.g. Display driver, into the startup sequence.
Dependencies:	Initialization of Basic Software modules.
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.1.2.1.1.1 [BSW09147] Configuration of de-initialization process of Basic Software modules

ID:	BSW09147
Initiator:	PSA
Date:	13.07.2005
Short Description	Configuration of de-initialization process of Basic Software modules
Type:	--
Importance:	High
Description:	**SRS1206**: The Basic Software modules with their de-initialization routines and their de-initialization order shall be pre-compile time configurable attributes of the ECU State Manager.
Rationale:	The ECU State Manager shall have knowledge about the de-initialization-routines from Basic Software modules and their predecessors and successor relationships. However, it shall be possible to adapt the concrete de-initialization process of the ECU State Manager to the project specific de-initialization demands of the Basic Software modules.

Use Case:	--
Dependencies:	De-initialization of Basic Software modules.
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.1.2.1.1.2 [BSW09146] Configuration of time triggered increased inoperation

ID:	BSW09146
Initiator:	PSA
Date:	08.07.2005
Short Description	Configuration of time triggered increased inoperation
Type:	--
Importance:	Medium
Description:	**SRS1209**: It shall be possible to enable or disable the time triggered increased inoperation at pre compile time. This feature shall stay of course disabled if the time triggered increased inoperation is not supported by the hardware (the wake-up/sleep capability and several sleep modes should be available on the ECU; see BSW09118
Rationale:	Making this feature configurable will save resources on ECUs which do not need it.
Use Case:	--
Dependencies:	- Wake-up/sleep capability shall be available on the ECU. - Time-triggered wake-up events shall be available on the ECU. - [BSW09119] Support of several sleep modes. - [BSW09118] Time triggered increased inoperation.
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.1.2.1.2 Normal Operation (EcuM Fixed)

4.1.2.1.2.1 [BSW09001] Standardization of state relations

ID:	BSW09001
Initiator:	DAI
Date:	13.02.2004
Short Description:	Standardization of state relations
Type:	--
Importance:	High
Description:	**SRS1210**: The number and names of main states and the transitions between main states shall be standardized.
Rationale:	The state machine is standardized.
Use Case:	Relocatability of software components.
Dependencies:	--
Conflicts:	--
Supporting Material:	--

Contributes to:	--

4.1.2.1.2.2 [BSW09173] Minimum duration of Run State

ID:	BSW09173
Initiator:	WP COM Stack
Date:	25.07.2007
Short Description	Minimum duration of Run State
Type:	--
Importance:	High
Description:	After entering the Run state, the ECU State Manager shall maintain this state for a minimum amount of time. The minimum duration to stay in Run state shall be configurable.
Rationale:	The ECU State Manager has to stay in Run state for a period of time in order to avoid immediate shutdown and to give users the chance/time to request RUN themselves
Use Case:	--
Dependencies:	[BSW09113] Initialization of Basic Software modules [BSW09114] Starting/invoking the shutdown process
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.1.2.1.2.3 [BSW09114] Starting/invoking the shutdown process

ID:	BSW09114
Initiator:	VW
Date:	16.12.2004
Short Description	Starting/invoking the shutdown process
Type:	--
Importance:	High
Description:	After the release of the last request for RUN state OR if the BSW forces the shutdown, the ECU State Manager shall leave the RUN state and start / invoke the shutdown process.
Rationale:	The ECU State Manager is the BSW module controlling the states of the ECU. Therefore the ECU State Manager is responsible to initiate the state transitions. Defined responsibility to initiate the shutdown process.
Use Case:	--
Dependencies:	The OS shall provide an interface/service to invoke its own shutdown. [BSW09072] Force ECU shutdown [BSW09173] Minimum duration of Run State
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.1.2.1.2.4 [BSW09113] Initialization of Basic Software modules

ID:	BSW09113
Initiator:	VW

Date:	15.12.2004
Short Description	Initialization of Basic Software modules
Type:	--
Importance:	High
Description:	**SRS1214**: The ECU State Manager shall initialize the Basic Software modules. When the init process has finished, the RUN state shall be entered. Nevertheless, the ECU State Manager is not responsible for the initialization of the application software components; this responsibility lies with the RTE.
Rationale:	Organize the initialization process of the BSW modules and give SWCs the possibility to request the RUN state.
Use Case:	Defined initialization process of the basic software architecture.
Dependencies:	- Predecessor and successor relationships of the Basic Software modules during the initialization process. - [BSW09120] Configuration of initialization process of Basic Software modules. - [BSW09173] Minimum duration of Run State
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.1.2.1.2.5 [BSW09009] Activation of software when entering/leaving ECU states

ID:	BSW09009
Initiator:	DAI
Date:	13.02.2004
Short Description:	Activation of software when entering/leaving ECU states
Type:	--
Importance:	High
Description:	**SRS1220**: The ECU State Manager shall provide the ability to execute external, statically-configured code at each transition between ECU states. This will be likely implemented by a callback when entering/leaving ECU states.
Rationale:	Basic Functionality.
Use Case:	Launching initialization/de-initialization routines when entering/leaving the RUN state.
Dependencies:	[BSW101] Initialization interface (WP Architecture).
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.1.2.1.2.6 [BSW09118] Time triggered increased inoperation

ID:	BSW09118
Initiator:	VW
Date:	16.12.2004
Short Description	Time triggered increased inoperation
Type:	--
Importance:	Medium

Description:	**SRS1227**: The ECU State Manager shall provide a mechanism to enter a step by step decreasing power mode (sleep modes), based on timeouts. This mechanism can only be available if the wake-up/sleep capability is available on the ECU; several sleep modes are available on the ECU. It shall be a configurable feature of the ECU State Manager (see BSW09146).
Rationale:	If the ECU provides a set of different sleep modes, it should be possible to define a time-triggered sequence to change the sleep modes without additional interaction from outside of the ECU. It is assumed that the different sleep modes represent a different level of low-power consumption. The change of sleep mode should lead into a step by step decreasing power-consumption of the ECU, while the ECU is not needed in the system.
Use Case:	If the car is left for a longer time – e.g. 3 days typically – more and more systems are shut off to save energy. The saving effect can be increased by shutting off more and more resources of the ECU itself: PLLs, clocks, sensors, etc. The aliveness of a comfort-body unit (ECU), for example, is not needed the whole time, because the user doesn't place any requests on the functionality of this unit (e.g. pressing a button) while the system is running. After a (configurable) period of time this unit could reduce its power consumption by entering a given sleep mode. Further it goes down in power-consumption by using the time triggered increased inoperation mechanism, while no user-requests are detected on this unit. Build up a system strategy with shortened startup, if the system was in usage immediately before. The longer the system is not in usage, the power-consumption is reduced by means of the time triggered increased inoperation, but as a consequence also the time for startup increases.
Dependencies:	- Wake-up/sleep capability shall be available on the ECU. - Time-triggered wake-up events shall be available on the ECU. - [BSW09119] Support of several sleep modes. - [BSW09146] Configuration of time triggered increased inoperation.
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.1.2.1.2.7 [BSW09115] Evaluate condition to stay in the RUN state

ID:	BSW09115
Initiator:	VW
Date:	16.12.2004
Short Description	Evaluate condition to stay in the RUN state
Type:	--
Importance:	High
Description:	**SRS1230**: The ECU State Manager shall include a mechanism to evaluate the condition to stay in the RUN state, derived from current demands of the application on the ECU.
Rationale:	The ECU State Manager is responsible for managing the ECU states. Therefore, it is necessary to evaluate the condition to stay in the RUN state, which has to be in relationship with the needs of the application software components.
Use Case:	--
Dependencies:	[BSW09116] Requesting and releasing the RUN state.
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.1.2.1.2.8 [BSW09164] Shutdown synchronization support for SW-Components

ID:	BSW09164
Initiator:	VW
Date:	28.11.2005
Short Description	Shutdown synchronization support for SW-Components
Type:	--
Importance:	Medium
Description:	When leaving the RUN state, the ECU State Manager shall provide a synchronization point for SWCs shutdown by defining a POST-RUN state. In this state, BSW and the underlying hardware are still fully functional. This state can only be entered when all users have released their RUN request. It is expected that the users release their POST-RUN request after finalization of their shutdown preparation activities, thus allowing the subsequent de-initialization of BSW.
Rationale:	This state will be requested by SWCs needing shutdown synchronization to perform shutdown activities that are required before their access to resources is decreased (i.e. Stop of RTE; Stop of OS).
Use Case:	A digital filter algorithm that does not have the knowledge when its functionality is needed but needs to perform activities to prepare for shutdown.
Dependencies:	[BSW09001] Standardisation of state relations. [BSW09116] Requesting and releasing the RUN state. [BSW09165] Requesting and releasing the POST-RUN state. [BSW09166] Evaluate condition to stay in the POST-RUN state.
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.1.2.1.2.9 [BSW09165] Requesting and releasing the POST-RUN state

ID:	BSW09165
Initiator:	DAI
Date:	28.11.2005
Short Description	Requesting and releasing the POST-RUN state
Type:	--
Importance:	Medium
Description:	The ECU State Manager shall provide services to request and release the POST-RUN state.
Rationale:	The decision to hold an ECU in the POST-RUN state is always related to the application context, which cannot be standardized. Therefore the ECU State Manager needs to get "user-requests" from application perspective to be aware of the current situation. It is recommended that independent applications (SWCs), having independent POST-RUN state needs, shall control their "user request" by usage of the provided interface of the ECU State Manager.
Use Case:	--
Dependencies:	[BSW09164] Shutdown synchronization support for SW-Components. [BSW09166] Evaluate condition to stay in the POST-RUN state.
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.1.2.1.2.10 [BSW09166] Evaluate condition to stay in the POST-RUN state

ID:	BSW09166
Initiator:	PSA
Date:	28.11.2005
Short Description	Evaluate condition to stay in the POST-RUN state
Type:	--
Importance:	Medium
Description:	The ECU State Manager shall include a mechanism to evaluate the condition to stay in the POST-RUN state, derived from current demands of the application on the ECU.
Rationale:	The ECU State Manager is responsible for managing the ECU states. Therefore, it is necessary to evaluate the condition to stay in the POST-RUN state, which has to be in relationship with the needs of the application software components.
Use Case:	--
Dependencies:	[BSW09164] Shutdown synchronization support for SW-Components. [BSW09116] Requesting and releasing the RUN state. [BSW09165] Requesting and releasing the POST-RUN state.
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.1.2.1.2.11 [BSW09145] Support of wake-sleep operation

ID:	BSW09145
Initiator:	WP Mode Management
Date:	28.06.2005
Short Description	Support of wake-sleep operation
Type:	--
Importance:	High
Description:	**SRS1228:** The ECU State Manager shall support a sequence consisting of: o partial startup o execution of limited jobs (see use case) o quick shutdown without OS context. This behaviour will be called "wake-sleep operation" in the context of the ECU State Manager.
Rationale:	Energy saving.
Use Case:	A cyclic wake-up by a timer-event, which should only do a short check of system wake-up conditions of non wakeable-sources (standard IO-ports) or execute a cyclic alive-feedback to customer, like i.e. a blinking LED.
Dependencies:	- Reduced capabilities of basic software architecture, according to non full startup for wake-sleep operation. - The wake-up/sleep hardware capability shall be available on the ECU.
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.1.3 Flex

The ECU State Manager Flex is a variant of the EcuM which can control the ECU-states in a very flexible way (partial run of BSW and Applications). This provides the basis to implement project specific energy saving strategies. To do so, it make use of the BswM (Basic Software Mode Manager).

4.1.3.1 Functional Requirements

4.1.3.1.1 Normal Operation (EcuM Flexible)

4.1.3.1.1.1 [BSW09234] Initialization of Basic Software modules

ID:	BSW09234
Initiator:	VW
Date:	15.12.2004
Short Description	Initialization of Basic Software modules
Type:	--
Importance:	High
Description:	The EcuM handles initialization until the mode management by the BswM starts.
Rationale:	Organize the initialization process of the BSW.
Use Case:	Defined initialization process of the basic software architecture.
Dependencies:	- [BSW09120] Configuration of initialization process of Basic Software modules.
Conflicts:	The ECU State Manager is not responsible for the initialization of the application software components; this responsibility lies with the RTE.
Supporting Material:	--
Contributes to:	--

4.1.3.1.1.2 [BSW09235] Support of several shutdown targets

ID:	BSW09235
Initiator:	WP Mode Management
Date:	28.06.2005
Short Description:	Support of several shutdown targets
Type:	--
Importance:	High
Description:	The ECU State Manager shall offer at least the following targets for shutting down the ECU: • Power Off • ECU Reset The default target shall be defined by configuration. There shall be an interface to change the target.
Rationale:	Allow trade off between energy consumption and time to wake-up/startup the ECU.
Use Case:	-- shutdown after error, -- shutdown for flashing
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.1.3.1.2 Configuration

ID:	--
Initiator:	VW
Date:	16.12.2004
Short Description	Configuration of users of the ECU State Manager
Type:	--
Importance:	High
Description:	Users, indicating the operational needs of individual software components, shall be pre compile time configurable attributes of the ECU State Manager. Requests from unknown sources are ignored.
Rationale:	All users are known and configured at compile-time. This requirement supports static administration of (known) state requestors on the ECU; it supports also testing and documentation.
Use Case:	--
Dependencies:	Requesting and releasing states.
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.1.3.1.2.1 [BSW09227] Configuration of privileged users

ID:	BSW09227
Initiator:	WP Functional Safety
Date:	11.07.2008
Short Description	Configuration of privileged users
Importance:	High
Description:	It shall be possible to configure a list of privileged users per protected service. Protected services (by sense of this configuration issue) are: - EcuM_SelectShutdownTarget The configuration shall be possible at link time.
Rationale:	To support defensive behavior of BSW modules it is necessary to declare the authorized users of a service at system generation time.
Use Case:	- A not allowed access to one of the protected services will be rejected without execution of the requested service. - An allowed user requests the execution of a protected service.
Dependencies:	--
Conflicts:	--
Supporting Material:	See BSW09198 Covered feature request: • BRF00128 (Protection against Unauthorized Use of BSW) WP Vehicle Mode Management and Application Mode Management: EcuM_KillAllRUNRequests is no longer considered, because it is not a service.
Contributes to:	--

4.1.3.1.3 Alarm Clock

4.1.3.1.3.1 [BSW09185] Provide a persistent Alarm Clock to be used by local SW-Cs

ID:	BSW09185

Initiator:	WP Architecture
Date:	29.05.2008
Short Description:	Provide a persistent Alarm Clock to be used by local SW-Cs
Importance:	High
Description:	The ECU State Manager shall provide a persistent alarm clock that tracks time and remains “active” even during sleep and allows for wakeup services to be requested based on a future time to guarantee the ECU is in a RUN state if requested by a local SW-C.
Rationale:	Allow internal wakeup requests based on time to leave a sleep state instead of only external I/O events.
Use Case:	--
Dependencies:	BSW09186
Conflicts:	--
Supporting Material:	Persistence does not mean non volatile during a powerless phase, but it shall be persistent while the ECU is in any sleep mode and still connected to a power supply. Covered feature request: o BRF00196 (Alarm Clock)
Contributes to:	--

4.1.3.1.3.2 [BSW09186] Alarm Clock shall be active while the ECU is powered

ID:	BSW09186
Initiator:	WP Architecture
Date:	29.05.2008
Short Description:	Alarm Clock shall be active while the ECU is powered
Importance:	High
Description:	The used time base shall count time while the ECU is powered. Each (re-) connect to a power supply shall reset the internal time to Zero, cancel all former active alarms and restart the time counting immediately.
Rationale:	It is whether possible to estimate passed time while the ECU has been not powered nor to initiate any action in this case.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	It is not necessary to keep armed alarms in memory during a powerless phase. Covered feature request: o BRF00196 (Alarm Clock)
Contributes to:	--

4.1.3.1.3.3 [BSW09187] Cancellation of Alarms in Case of wakeup

ID:	BSW09187
Initiator:	WP Architecture
Date:	29.05.2008
Short Description:	Cancellation of alarms in case of wakeup
Importance:	High
Description:	When the ECU is waking up, all alarms of the alarm clock shall be canceled.
Rationale:	This avoids undesired behavior in case of unexpected events which wakes up the ECU. The SW-Cs are responsible to arm the alarm clock while they

	are executed.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	Covered feature request: o BRF00196 (Alarm Clock)
Contributes to:	--

4.1.3.1.3.4 [BSW09188] Cancellation of Alarms in Case of startup

ID:	BSW09188
Initiator:	WP Architecture
Date:	29.05.2008
Short Description:	Cancellation of alarms in case of startup
Importance:	High
Description:	When the ECU is going through a power on reset, all alarms of the alarm clock shall be canceled.
Rationale:	This avoids undesired behavior in case of unexpected events which wakes up the ECU. The SW-Cs are responsible to arm the alarm clock while they are executed.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	Covered feature request: o BRF00196 (Alarm Clock)
Contributes to:	--

4.1.3.1.3.5 [BSW09189] Consideration of the earliest expiring Wakeup only

ID:	BSW09189
Initiator:	WP Architecture
Date:	29.05.2008
Short Description:	Consideration of the earliest expiring wakeup alarm only
Importance:	High
Description:	Consecutive requests shall honor the earliest expiring alarm only.
Rationale:	This avoids undesired behavior in case of unexpected events which wakes up the ECU. The SW-Cs are responsible to arm the alarm clock while they are executed.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	Covered feature request: o BRF00196 (Alarm Clock)
Contributes to:	--

4.1.3.1.3.6 [BSW09190] Provision of an Interface to set relative Alarms

ID:	BSW09190
Initiator:	WP Architecture

Date:	29.05.2008
Short Description:	Provision of an interface to set relative alarms
Importance:	high
Description:	The alarm clock service shall allow setting an alarm relative to the current time using a time resolution of seconds. There is no access restriction except that is allowed only to use this interface while the ECU is leaving the RUN mode (POSTRUN).
Rationale:	The SW-Cs shall be able to set an alarm
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	Covered feature request: o BRF00196 (Alarm Clock)
Contributes to:	--

4.1.3.1.3.7 [BSW09199] Provision of an Interface to set absolute Alarms

ID:	BSW09199
Initiator:	WP Architecture
Date:	29.05.2008
Short Description:	Provision of an interface to set absolute alarms
Importance:	High
Description:	The alarm clock service shall allow setting an alarm absolute by using an absolute time (but relative to start of the timer) with a resolution of seconds. There is no access restriction except that is allowed only to use this interface while the ECU is leaving the RUN mode (POSTRUN).
Rationale:	The SW-Cs shall be able to set an alarm
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	Covered feature request: o BRF00196 (Alarm Clock)
Contributes to:	--

4.1.3.1.3.8 [BSW09194] Provision of an Interface to set the Clock

ID:	BSW09194
Initiator:	WP Architecture
Date:	29.05.2008
Short Description:	Provision of an interface to set the clock
Importance:	High
Description:	The alarm clock service shall allow setting the clock. There is no access restriction. This could be used to synchronize the alarm clock with any other clocks (e.g. received by a GPS receiver).
Rationale:	The SW-Cs shall be able to synchronize the alarm clock with other time sources.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	Covered feature request:

	o BRF00196 (Alarm Clock)
Contributes to:	--

4.1.3.1.4 Defensive Behavior of BSWMs

4.1.3.1.4.1 [BSW09195] Protection against untimely Call of EcuM_Init

ID:	BSW09195
Initiator:	WP Functional Safety
Date:	11.06.2008
Short Description:	Protection against untimely call of EcuM_Init
Importance:	High
Description:	To ensure protection against untimely call of EcuM_Init, the ECU State Manager shall check its internal state before performing its initialization. If the ECU State Manager has already been initialized, it shall return without any treatment and inform DEM of the problem.
Rationale:	Protection against faulty initialization of the ECU
Use Case:	o Corruption of the program counter / instruction pointer can cause an illegal execution of a piece of code. o Wrong initialization of a functions pointers array can cause an illegal execution of a piece of code.
Dependencies:	--
Conflicts:	--
Supporting Material:	Covered feature request: o BRF00128 (Protection against Unauthorized Use of BSW)
Contributes to:	--

4.1.3.1.4.2 [BSW09197] Protection against erroneous Call of EcuM_KillAllRUNRequests

ID:	BSW09197
Initiator:	WP Functional Safety
Date:	11.06.2008
Short Description:	Protection against erroneous call of EcuM_KillAllRUNRequests
Importance:	High
Description:	To ensure protection against erroneous call of EcuM_KillAllRUNRequests service, the ECU State Manager shall check if the caller of this service is a registered user. If it is not the case, the ECU State Manager shall return without any treatment and inform DEM of the problem.
Rationale:	Protection against faulty shutdown of the ECU
Use Case:	o A non registered BSW module OR CDD calls this service Corruption of the program counter / instruction pointer can cause an illegal execution of a piece of code. o Wrong initialization of a functions pointers array can cause an illegal execution of a piece of code.
Dependencies:	BSW09227
Conflicts:	--
Supporting Material:	Covered feature request: o BRF00128 (Protection against Unauthorized Use of BSW) Rejected by WP Vehicle Mode Management and Application Mode

	Management: This call is not a AUTOSAR service and we don't expect not intended calls by BSW Modules.
Contributes to:	--

4.1.3.1.4.3 [BSW09198] Protection against erroneous Call of EcuM_SelectShutdownTarget

ID:	BSW09198
Initiator:	WP Functional Safety
Date:	11.06.2008
Short Description:	Protection against erroneous call of EcuM_SelectShutdownTarget
Importance:	High
Description:	To ensure protection against erroneous call of EcuM_SelectShutdownTarget service, the ECU State Manager shall check if the caller of this service is a registered user and if this registered user has permission to ask for this shutdown target. If it is not the case, the ECU State Manager shall return without any treatment and inform DEM of the problem.
Rationale:	To avoid the ECU to be put in sleep mode instead of reset and vice versa
Use Case:	o Energy management needs permission for each kind of available shutdown target whereas a monitoring entity will only need permission for reset.
Dependencies:	BSW09227
Conflicts:	--
Supporting Material:	Covered feature request: o BRF00128 (Protection against Unauthorized Use of BSW)
Contributes to:	--

4.1.3.1.5 Multi Core

4.1.3.1.5.1 [BSW09236] Distinguish cores

ID:	BSW09236
Initiator:	BOSCH
Date:	2.4.2009
Short Description:	EcuM_Init has to distinguish between the different cores.
Type:	new
Importance:	high
Description:	There shall be one instance of the function EcuM_Init that distinguishes between the different cores by means of the OS service "`GetCoreID`".
Rationale:	The "InitSequence 1" can be core specific. The philosophy of the Multi-Core architecture is to have one binary file, hence "EcuM_Init" exists only once.
Use Case:	• Core specific hardware initialisation • One core starts the other cores while the other cores do not start further cores.
Dependencies:	--
Conflicts:	--
Supporting Material:	AUTOSAR Specification of Multi-Core OS Architecture
Contributes to:	--

4.1.3.1.5.2 [BSW09237] Starting of RTE

ID:	BSW09237
Initiator:	BOSCH
Date:	2.4.2009
Short Description:	RTE_Start shall be called on each core.
Type:	new
Importance:	Medium
Description:	RTE_Start shall be called locally on each core by means of tasks triggered by the BswM on the main core through dedicated available actions.
Rationale:	It is necessary to coordinate the RTE starts on the cores, as the RTEs on the slave cores might request use of modules on the main core which may not be initialised yet. All the same, it is necessary to avoid a cross core propagation of the RTE, which would increase its complexity and make it much more than a “macro layer” as it now is.
Use Case:	Without coordination, start of RTE on slave cores could happen much earlier than on the main core, on which much more modules are running, each requiring init time.
Dependencies:	--
Conflicts:	--
Supporting Material:	AUTOSAR Specification of Multi-Core OS Architecture
Contributes to:	--

4.1.3.1.5.3 [BSW09238] State changes are ECU global

ID:	BSW09238
Initiator:	BOSCH
Date:	2.4.2009
Short Description:	State changes are ECU global
Type:	New
Importance:	High
Description:	State changes shall be ECU global (all cores are switching into a new state). Only the entire ECU changes into either "off", "fully functional" or "sleeping" (halted or set to poll)
Rationale:	The complexity and problems that would be introduced with multiple parallel existing states shall be reduced/avoided. (at least for R 4.0)
Use Case:	Every state change
Dependencies:	WP Vehicle Mode Management and Application Mode Management assumes that the MCU-Driver handles calls to Mcu_PerformReset and Mcu_SetMode from core 0 consistently across all cores
Conflicts:	--
Supporting Material:	AUTOSAR Specification of Multi-Core OS Architecture
Contributes to:	--

4.1.3.1.5.4 [BSW09239] Synchronized Shutdown

ID:	BSW09239
Initiator:	BOSCH
Date:	2.4.2009
Short Description:	Call `ShutdownAllCores` on the master core to shutdown the ECU
Type:	New
Importance:	High
Description:	To go Down (Shutdown the ECU) , `ShutdownAllCores` shall be called

	on the master core after synchronizing all cores
Rationale:	It is in the responsibility of the application developer/system integrator to make sure that any preparations for shutdown on application and basic software level are completed before calling "`ShutdownAllCores`". Core individual shutdown is not supportded by AUTOSAR R4.0
Use Case:	Synchronized shutdown of the whole ECU
Dependencies:	- State changes are ECU global - Multiple EcuM Instances
Conflicts:	--
Supporting Material:	AUTOSAR Specification of Multi-Core OS Architecture
Contributes to:	--

4.2 Watchdog Manager

4.2.1 Functional Overview

The Watchdog Manager is a basic software module of the standardized basic software architecture of AUTOSAR (service layer). It is intended to supervise the reliability of application execution with respect to timing constraints (temporal program flow monitoring) and with respect to the correct sequence of execution (logical program flow monitoring).
Derived from the layered architecture approach, a decoupling between application timing constraints and watchdog hardware timing constraints becomes possible. Based on this decoupling the Watchdog Manager provides temporal program flow monitoring (alive-supervision) of several independent applications as well as logical program flow monitoring (the supervision of the correct execution order) while triggering the watchdog hardware.
The following features are provided by the Watchdog Manager:

- Supervision of multiple individual applications located on the ECU, having independent timing constraints and requiring a special supervision of runtime behaviour and aliveness.
- Logical supervision of safety-related tasks and periodic functions (main functions).
- Fault-reaction mechanism for each independent supervised entity.
- Possibility to switch off supervision of individual applications, without violating the watchdog triggering (e.g. for inhibited applications).
- Triggering of internal or external, standard or window, watchdog, via a watchdog driver. The access to the internal or external watchdog will be handled by the Watchdog Interface.
- Selection of the watchdog mode (Off Mode, Slow Mode, Fast Mode) depending on the ECU state and the hardware capabilities.

Much more details about these functionalities can be found in the Watchdog Manager SWS specification [3].

4.2.2 Functional Requirements

4.2.2.1 Initialization

4.2.2.1.1 [BSW09107] Watchdog Manager initialization

ID:	BSW09107
Initiator:	WP Mode Management
Date:	05.10.2004
Short Description:	Watchdog Manager initialization
Type:	--
Importance:	high
Description:	The Watchdog Manager shall provide an initialization service. This service allows the selection of one of the statically configured Watchdog Manager modes.
Rationale:	Basic functionality.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.2.2.1.2 [BSW09109] Opportunity to prohibit the disabling of watchdog

ID:	BSW09109
Initiator:	WP Mode Management
Date:	21.10.2004
Short Description:	Opportunity to prohibit the disabling of watchdog
Type:	--
Importance:	high
Description:	It shall be possible to configure, by means of a pre-processor switch, whether the Watchdog Manager initialization service and the Watchdog Manager mode selection service allow the disabling of the watchdog or not.
Rationale:	Avoid the presence of code sequences in a safety relevant ECU that disable the watchdog.
Use Case:	Usage within safety relevant systems.
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.2.2.2 Normal Operation

4.2.2.2.1 [BSW09112] Temporal program flow monitoring (Check aliveness of application)

ID:	BSW09112
Initiator:	WP Mode Management
Date:	05.10.2004
Short Description:	Temporal program flow monitoring (Check aliveness of application)
Type:	--
Importance:	high
Description:	The Watchdog Manager shall cyclically check the periodicity of the supervised entities, in order to detect aliveness of application.
Rationale:	This requirement is needed by AUTOSAR Software Components requiring a special supervision of runtime behavior and aliveness.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.2.2.2.2 [BSW09221] Logical program flow monitoring (Check correct execution sequence of supervised entities)

ID:	BSW09221
Initiator:	WP Mode Management
Date:	16.06.2008
Short Description:	Logical program flow monitoring (Check correct execution sequence of supervised entities)
Type:	--
Importance:	High
Description:	The Watchdog Manager shall check the correct sequence of code execution in supervised entities.
Rationale:	To enable to detect if a program runs in an incorrect sequence.
Use Case:	SW-Cs and BSW modules call logical program flow monitoring in order to detect the deviation from the correct execution sequence.
Dependencies:	--
Conflicts:	--
Supporting Material:	Covered feature request: o BRF00131 (Logical Program Flow Monitoring)
Contributes to:	--

4.2.2.2.3 [BSW09125] Update temporal program flow monitoring

ID:	BSW09125
Initiator:	WP Mode Management
Date:	05.10.2004
Short Description:	Update temporal program flow monitoring
Type:	--
Importance:	High
Description:	The Watchdog Manager shall provide a service allowing supervised entities to forward an alive indication, thus updating the temporal program flow monitoring for the given entity.

Rationale:	This requirement is needed by AUTOSAR Software Components requiring a special supervision of runtime behavior and aliveness
Use Case:	--
Dependencies:	This could also be done via the service in BSW09222
Conflicts:	Provided only for backward compatibility with earlier versions of the Watchdog Manager
Supporting Material:	--
Contributes to:	--

4.2.2.2.4 [BSW09222] Update logical program flow monitoring

ID:	BSW09222
Initiator:	WP Mode Management
Date:	16.06.2008
Short Description:	Service for logical program flow monitoring
Type:	--
Importance:	High
Description:	The Watchdog Manager shall provide a service allowing SW-Cs and BSW modules to indicate that they have reached a checkpoint in their code execution.
Rationale:	This requirement is needed by SW-Cs and BSW modules requiring a special supervision of execution sequence.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	Covered feature request: o BRF00131 (Logical Program Flow Monitoring)
Contributes to:	--

4.2.2.2.5 [BSW09160] Indication of failed temporal monitoring

ID:	BSW09160
Initiator:	WP Mode Management
Date:	16.11.2005
Short Description	Indication of failed temporal monitoring
Type:	--
Importance:	high
Description:	The Watchdog Manager shall be able to notify the application when the monitoring of a supervised entity by temporal program flow monitoring fails. This information shall be forwarded through the RTE in order to allow fault reaction by software. In this case, the information about which supervised entity has failed shall also be made available to the application.
Rationale:	This gives the opportunity to establish some fault-reactions by software before a watchdog-reset occurs (fault recovery/reporting to the Diagnostic Event Manager).
Use Case:	--
Dependencies:	Temporal program flow monitoring of the watchdog-manager. [BSW09112] Temporal program flow monitoring of application.
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.2.2.2.6 [BSW09225] Indication of failed logical monitoring

ID:	BSW09225
Initiator:	WP Functional Safety
Date:	09.07.2008
Short Description	Indication of failed logical monitoring
Type:	--
Importance:	High

Description:	The Watchdog Manager shall be able to notify the application when the monitoring of a supervised entity by logical program flow monitoring fails. This information shall be forwarded through the RTE in order to allow fault reaction by software. In this case, the information about which supervised entity ID and which check point has failed shall also be made available to the application.
Rationale:	This gives the opportunity to establish some fault-reactions by software before a watchdog-reset occurs (fault recovery/reporting to the Diagnostic Event Manager).
Use Case:	--
Dependencies:	[BSW09221] Logical program flow monitoring
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.2.2.2.7 [BSW09161] Condition to reset the triggering condition in the Watchdog Driver in Case of temporal failure

ID:	BSW09161
Initiator:	PSA
Date:	03.11.2005
Short Description	Condition to reset the triggering condition in the Watchdog Driver in case of temporal failure
Type:	--
Importance:	High
Description:	The Watchdog Manager shall reset via the Watchdog Interface the triggering condition in the Watchdog Driver (to the value 0) if the temporal monitoring of a given supervised entity fails continuously during a configurable amount of time (this amount of time can be set to 0 by configuration, thus excluding any software recovery possibility).
Rationale:	This means the monitoring failure is now considered unrecoverable. Then, a watchdog reset is the only solution.
Use Case:	--
Dependencies:	[BSW09162] Indication of an upcoming watchdog reset.
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.2.2.2.8 [BSW09226] Condition to reset the triggering condition in the Watchdog Driver in Case of logical program flow failure

ID:	BSW09226
Initiator:	WP Functional Safety
Date:	09.07.2008
Short Description	Condition to reset the triggering condition in the Watchdog Driver in case of logical program flow violation
Type:	--
Importance:	High
Description:	The Watchdog Manager shall reset via the Watchdog Interface the triggering condition in the Watchdog Driver (to the value 0) if the monitoring of the execution sequence fails.
Rationale:	This means the monitoring failure is now considered unrecoverable. Then, a

	watchdog reset is the only solution.
Use Case:	--
Dependencies:	[BSW09162] Indication of an upcoming watchdog reset.
Conflicts:	--
Supporting Material:	It shall not be possible to configure a number of times similar to BSW09161.
Contributes to:	--

4.2.2.2.9 [BSW09169] Immediate reset of the Watchdog Manager

ID:	BSW09169
Initiator:	WP Mode Management
Date:	01.08.2006
Short Description	Immediate reset of the Watchdog Manager
Type:	--
Importance:	High
Description:	The Watchdog Manager shall be able to immediately reset the MCU when a temporal or logical program flow monitoring failure is detected, without waiting for the hardware watchdog to expire. This feature shall be configurable. If this feature is enabled, no notification will be sent to the application (see BSW09162).
Rationale:	When the Watchdog Manager has detected an unrecoverable fault it will simply reset the triggering condition of the watchdog(s). A reset will only occur when the first watchdog expires. In some use cases it is necessary to reset the ECU as soon as the unrecoverable fault has been detected. In these cases the WdgM shall perform a reset as soon as possible. This shall be a configurable feature.
Use Case:	--
Dependencies:	[BSW09162] Indication of an upcoming watchdog reset.
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.2.2.2.10 [BSW09162] Indication of an upcoming watchdog reset

ID:	BSW09162
Initiator:	PSA
Date:	03.11.2005
Short Description	Indication of an upcoming watchdog reset
Type:	--
Importance:	high
Description:	The Watchdog Manager shall be able to notify the software (BSW and SW-C) when the decision to reset the triggering condition has been taken due to a temporal or logical program flow monitoring failure. This information shall be forwarded through the RTE in order to allow software preparing to the imminent reset.
Rationale:	Some applications need to perform some activity before a reset.
Use Case:	Storing of persistent data in the NV-RAM.
Dependencies:	[BSW09163] Delay before provoking a watchdog reset.
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.2.2.2.11 [BSW09163] Delay before provoking a watchdog reset

ID:	BSW09163
Initiator:	PSA
Date:	03.11.2005

Short Description	Delay before provoking a watchdog reset
Type:	--
Importance:	High
Description:	It shall be possible to configure a delay between the moment the decision has been taken to reset the triggering condition due to a temporal or logical program flow monitoring failure and the moment the Watchdog Manager actually resets the triggering condition.
Rationale:	Giving the time to the whole software to prepare properly to the upcoming reset.
Use Case:	--
Dependencies:	[BSW09162] Indication of an upcoming watchdog reset
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.2.2.2.12 [BSW09143] Behavior of the Watchdog Manager during inactive monitoring

ID:	BSW09143
Initiator:	VW
Date:	24.05.2005
Short Description:	Behaviour of the Watchdog Manager during inactive monitoring
Type:	--
Importance:	High
Description:	The Watchdog Manager set the triggering condition in the Watchdog Driver also when no monitoring is active/necessary.
Rationale:	Prevent unintended watchdog resets.
Use Case:	There are no supervised entities or for all supervised entities monitoring is switched off.
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.2.2.2.13 [BSW09231] Setting the triggering condition

ID:	BSW09231
Initiator:	WP Functional Safety
Date:	09.12.2008
Short Description:	Setting the triggering condition
Type:	--
Importance:	High
Description:	The Watchdog Manager shall periodically set the triggering condition in the Watchdog Driver (via the Watchdog Interface) as long as the monitoring of the supervised entities has not failed, thus preventing the hardware watchdog from expiring.
Rationale:	Basic functionality.
Use Case:	The Watchdog Manager sets the triggering condition to a configured value. The Watchdog Driver triggers the hardware watchdog cyclically and decrements the triggering condition. If the triggering condition reaches zero, the Watchdog Driver stops triggering the hardware watchdog. If the

	Watchdog Manager periodically sets the triggering condition, the hardware watchdog will never expire. Even if the Watchdog Manager fails, it will also fail to set the triggering condition, and thus cause a watchdog reset.
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.2.2.2.14 [BSW09110] Watchdog Manager mode selection service

ID:	BSW09110
Initiator:	WP Mode Management
Date:	21.10.2004
Short Description:	Watchdog Manager mode selection service
Type:	--
Importance:	High
Description:	The watchdog Manager shall provide a service interface, to select a mode of the Watchdog Manager. Each mode corresponds to a watchdog driver mode.
Rationale:	The watchdog triggering will be provided by three modes of the watchdog driver (off, slow, fast), which will be related to the length of the time-period before the watchdog expires (no expiration, short time-period and long time-period). The decision about which mode is necessary could not be done internally by the Watchdog Manager. Therefore, the Watchdog Manager provides an interface to perform the watchdog driver mode selection.
Use Case:	Appropriate modes of the watchdog are typically related to the current state of the ECU. The ECU State Manager could forward a mode-selection from slow to fast, when initialization phase is finished.
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.2.2.2.15 [BSW09028] Support multiple watchdog instances

ID:	BSW09028
Initiator:	Siemens VDO Automotive
Date:	09.08.2004
Short Description:	Support multiple watchdog instances
Type:	--
Importance:	High
Description:	The Watchdog Manager shall support multiple watchdog instances.
Rationale:	There are ECUs including both an internal and an external watchdog for monitoring the system.
Use Case:	Due to the usage of the same clock, the internal watchdog doesn't recognize the "hang-up" of a system. To achieve a higher robustness an external watchdog is used too.
Dependencies:	[BSW09167] Independent timing-constraints for watchdog-instances
Conflicts:	--
Supporting Material:	--

Contributes to:	--

4.2.2.2.16 [BSW09233] Independent triggering condition values for watchdog instances

ID:	BSW09233
Initiator:	WP Functional Safety
Date:	08.12.2008
Short Description	Independent triggering condition values for watchdog instances
Type:	--
Importance:	Medium
Description:	The Watchdog Manager shall support independent triggering condition values for each watchdog instance under its control.
Rationale:	If multiple watchdog instances exist on one platform (e.g. internal and external watchdog) it's very likely, that these watchdogs will come up with different timing constraints. Thus, the values of the triggering condition need to be adapted to the different cycles in the Watchdog Drivers.
Use Case:	HW-platform with internal and external watchdog
Dependencies:	[BSW09028] Support multiple watchdog instances
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.2.2.2.17 [BSW09232] Service to cause a watchdog reset

ID:	BSW09232
Initiator:	WP Architecture
Date:	25.11.2008
Short Description:	Service to cause a watchdog reset
Importance:	High
Description:	The Watchdog Manager shall provide a service that can be used to cause a watchdog reset independent of the monitoring states of supervised entities.
Rationale:	DCM needs to cause a watchdog reset when it switches between application and boot loader mode.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	Covered feature request: o [BRF00034] Bootloader Interaction
Contributes to:	--

4.2.2.3 Configuration

4.2.2.3.1 [BSW09106] Configuration of Watchdog Manager

ID:	BSW09106
Initiator:	WP Mode Management
Date:	05.10.2004
Short Description:	Configuration of Watchdog Manager
Type:	--
Importance:	High
Description:	The list of entities supervised by the Watchdog Manager shall be configurable at pre-compile time. The usage of additional configuration classes is not restricted
Rationale:	Application supervising.
Use Case:	--
Dependencies:	For the RTE interface to be generated, the supervised entities have to be known.
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.2.2.3.2 [BSW09220] Configuration of all transition relations

ID:	BSW09220
Initiator:	WP Functional Safety
Date:	16.06.2008
Short Description:	Configuration of all transition relations
Type:	--
Importance:	High
Description:	It shall be possible to configure the transition relations for each supervised entity and global transitions between supervised entities.

Rationale:	There are two possibilities to establish logical supervision of the execution sequence: • One global transition relation for the whole ECU • One transition relation for each supervised entity. Both of the possibilities shall be supported.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	Covered feature request: o BRF00131 (Logical Program Flow Monitoring)
Contributes to:	--

4.2.2.3.3 [BSW09158] Post build time and mode dependent selectable configuration sets for the Watchdog Manager

ID:	BSW09158
Initiator:	PSA
Date:	11.10.2005
Short Description:	Post build time and mode dependent selectable configuration sets for the Watchdog Manager
Type:	--
Importance:	high
Description:	The Watchdog Manager shall support several mode dependent selectable configuration sets on one ECU. These configuration sets shall be post build time configurable and shall include the individual timing constraints of each supervised entity. It shall also be possible to switch between these different configurations sets at runtime (e.g. depending on the current scheduling table).
Rationale:	Adapting temporal monitoring to the current schedule.
Use Case:	- Schedule table switch between STARTUP II and RUN states (see ECU State Manager SWS). - Allowing a limp-home mode, having a totally different schedule.
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.2.2.3.4 [BSW09223] Post build time and mode dependent selectable configuration of transition relations

ID:	BSW09223
Initiator:	WP Functional Safety
Date:	10.07.2008
Short Description:	Post build time and mode dependent selectable configuration of transition relations
Type:	--
Importance:	High
Description:	The Watchdog Manager shall support several mode dependent selectable configuration sets on one ECU. These configuration sets shall be post build time configurable and shall include the transition relations of each supervised entity. It shall also be possible to switch between these different configurations sets

	at runtime (e.g. depending on the current scheduling table).
Rationale:	Adapting logical program flow monitoring to the current schedule.
Use Case:	• Schedule table switch between STARTUP II and RUN states (see ECU State Manager SWS). • Allowing a limp-home mode, having a totally different schedule.
Dependencies:	--
Conflicts:	--
Supporting Material:	Covered feature request: o BRF00131 (Logical Program Flow Monitoring)
Contributes to:	--

4.2.3 Fault Operation

4.2.3.1.1 [BSW09159] Reporting failure of temporal or program flow monitoring to DEM

ID:	BSW09159
Initiator:	WP Mode Management
Date:	18.10.2005
Short Description:	Reporting failure of temporal or program flow monitoring to DEM
Type:	--
Importance:	High
Description:	The Watchdog Manager shall report to DEM the failure of temporal or program flow monitoring (When a failure is detected). The report to DEM shall be a configurable option of the Watchdog Manager.
Rationale:	Error tracing, Diagnostics.
Use Case:	--
Dependencies:	The current API of the DEM does not allow reporting the information about which supervised entity has failed.
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.3 Communication Manager

4.3.1 Functional Overview

The Communication Manager collects and coordinates the communication access requests from communication requestors (users, see glossary).
The purpose of the Communication Manager is:

- Simplifying the usage of the communication protocol stack for the user. This includes the starting and stopping physical channel communication and a simplified network management handling.
- Temporarily disabling sending of messages to prevent the ECU from (actively) waking up the physical channel(s).
- Controlling of more than one physical channels of an ECU by implementing a state machine for every physical channel.
- Requesting the appropriate communication state to the BusState Manager
- Simplifying the resource management by allocating all resources necessary for the requested communication mode.

In order to do so, the Communication Manager defines “communication modes”, indicating if a given physical channel is available for the application and how (send/receive; only receive, neither send nor receive).
Much more details about these functionalities can be found in the Communication Manager SWS specification [1].

4.3.2 Functional Requirements

4.3.2.1 Normal Operation

4.3.2.1.1 [BSW09078] Coordinating communication requests

ID:	BSW09078
Initiator:	BMW
Date:	01.12.2004
Short Description:	Coordinating communication requests
Type:	--
Importance:	high
Description:	The Communication Manager shall coordinate multiple communication requests for multiple physical channels independently. The rule for coordination is that the highest requested communication mode is the target state of the physical channel (see BSW09083 for a brief description of the communication modes).
Rationale:	Main functionality of Communication Manager. A SW-C cannot know with which other SW-C it will share an ECU in a particular configuration. Coordination of communication requests has to be provided by BSW.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.3.2.1.2 [BSW09246] Coordinating communication requests on Physical Channels and PNCs

ID:	BSW09246
Initiator:	WP1.1.1-EEM
Date:	16.03.2011
Short Description:	Coordinating communication requests on physical channels and PNCs
Type:	--
Importance:	high
Description:	The communication manager shall arbitrate and coordinate requests from users on physical channel and users on PNCs.
Rationale:	PNCs are virtual channel running on physical channels. More than one PNC can be located on a single physical channel and the communication mode of the channel has an effect on the PNCs running on it.
Use Case:	A channel is used directly by a single use and indirectly by one or more PNC users. The channel remains in Full Comm as long as the channel user remains in Full Comm, even in the case all the PNC user have released their communication.

Dependencies:	--
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.3.2.1.3 [BSW09247] Independent representation of PNC activation state

ID:	BSW09247
Initiator:	WP1.1.1-EEM
Date:	16.03.2011
Short Description:	Independent representation of PNC activation state
Type:	--
Importance:	high
Description:	For each configured PNC an independent state machine shall be instantiated, in order to keep track of the activation state of each PNC.
Rationale:	PNC must be able to be activated and deactivated independently of each other, in the same way physical channels are independent of each other.
Use Case:	On a specific physical channel, three PNCs are mapped and each can have a different activation status at any given moment.
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.3.2.1.4 [BSW09248] Internal distinction between internal and external PNC activation requests

ID:	BSW09248
Initiator:	WP1.1.1-EEM
Date:	16.03.2011
Short Description:	Internal distinction between internal and external PNC activation requests
Type:	--
Importance:	high
Description:	When a PNC is in full communication mode, there shall be substates indicating whether the activation request is originating from the ECU local applciation or from an external PNC activator.
Rationale:	Decision whether to release a PNC or not depends on the origin of the activation request and therefore it must be taken track of it.
Use Case:	A PNC is requested from the ECU local application, as long as this is true, the PNC shall not be deactivated. In case it is required from outside, it shall only remain active as long as it is constantly requested.
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.3.2.1.5 [BSW049] Initiating wake-up and keeping awake physical channels

ID:	BSW049
Initiator:	DAI
Date:	09.01.2004
Short Description:	Initiating wake-up and keeping awake physical channels
Type:	--
Importance:	high
Description:	The Communication Manager controls the Network Management modules assigned to the physical channels according to the target communication modes of these physical channels. As soon as a user requests communication, the Communication Manager shall initiate the physical channel wake-up by switching the affected physical channel's NM to the AWAKE state (requesting communication for this physical channel).
Rationale:	If the Network Management has set the communication system into sleep mode the Communication Manager shall wake it up again if at least one ECU/Application requires physical channel communication. Basic functionality, responsibility to initiate physical channel wake-up.
Use Case:	--
Dependencies:	[BSW09087] Minimum duration of communication request after wakeup.
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.3.2.1.6 [BSW09080] Physical channel independency

ID:	BSW09080
Initiator:	BMW
Date:	01.12.2004
Short Description:	Physical channel independency
Type:	--
Importance:	high
Description:	Each physical channel shall be controlled by an independent communication mode.
Rationale:	Possibility to have partial networks at physical channel level. Unnecessary physical channels shall not be activated.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.3.2.1.7 [BSW09081] API for requesting communication

ID:	BSW09081
Initiator:	BMW
Date:	01.12.2004
Short Description:	API for requesting communication
Type:	--

Importance:	high
Description:	The Communication Manager shall provide an API allowing collecting communication requests.
Rationale:	Means of interaction necessary
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.3.2.1.8 [BSW09083] Support of different communication modes

ID:	BSW09083
Initiator:	BMW
Date:	01.12.2004
Short Description:	Support of different communication modes
Type:	--
Importance:	high
Description:	The Communication Manager shall support Two communication modes for each physical channel. Theses modes are: • full communication (send & receive operations) • no communication (neither send nor receive operations) The modes have an implicit order with full communication being the “highest” mode and no communication the “lowest” mode.
Rationale:	Full communication mode should be self explanatory, needed for normal operation of distributed functions.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.3.2.1.9 [BSW09084] API for querying the current communication mode

ID:	BSW09084
Initiator:	BMW
Date:	01.12.2004
Short Description:	API for querying the current communication mode
Type:	--
Importance:	high
Description:	The Communication Manager shall provide an API which allows application to query the current communication mode.
Rationale:	SW-Cs shall have the possibility to query the real mode because it is unknown if and when a requested mode is reached. The querying of the requested mode is added for completeness (see BSW09149).
Use Case:	--
Dependencies:	[BSW09149] API for querying the requested communication mode.
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.3.2.1.10 [BSW09172] Evaluation of current communication mode

ID:	BSW09172
Initiator:	WP COM Stack
Date:	09/07/07
Short Description:	Evaluation of current communication mode
Type:	--
Importance:	High
Description:	If more than one channel is linked to one user request and the modes of the channels are different, the user shall get always the lowest mode indicated.
Rationale:	One user may request more than one communication Channel
Use Case:	--
Dependencies:	[BSW09084] API for querying the current communication mode [BSW09149] API for querying the requested communication mode.
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.3.2.1.11 [BSW09149] API for querying the requested communication mode

ID:	BSW09149
Initiator:	PSA
Date:	28.07.2005
Short Description:	API for querying the requested communication mode
Type:	--
Importance:	high
Description:	The Communication Manager shall provide an API which allows application to query the requested (target) communication mode.
Rationale:	SW-Cs shall have the possibility to query the real mode because it is unknown if and when a requested mode is reached (see BSW09084). The querying of the requested mode is added for completeness.
Use Case:	--
Dependencies:	[BSW09084] API for querying the current communication mode
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.3.2.1.12 [BSW09085] Indication of communication mode changes

ID:	BSW09085
Initiator:	BMW
Date:	01.12.2004
Short Description:	Indication of communication mode changes
Type:	--
Importance:	high
Description:	The Communication Manager shall provide an indication in order to notify application and DCM when the communication mode of the user has changed.
Rationale:	State changes shall be communicated to affected entities, constant polling would hinder performance.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.3.2.1.13 [BSW09168] Pseudo-channel for local communication

ID:	BSW09168
Initiator:	WP Mode Management
Date:	31.07.2006
Short Description:	Pseudo-channel for local communication
Type:	--
Importance:	high
Description:	The Communication Manager shall support users that are connected to no physical channel. If one or more SW-Cs request communication and are connected to the pseudo-channel for local communication, the channel state shall be Full communication and Run shall be requested from EcuM.

Rationale:	This requirement is necessary to support the AUTOSAR architecture. A SW-C description has to be independent of the mapping of SW-Cs to ECUs. Consequently, the service ports to the ComM have to be defined independent of the mapping of the SW-Cs connections being ECU internal or external.
Use Case:	--
Dependencies:	[BSW09090] User-to-channel relationship.
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.3.2.1.14 BSW09071] Limit Communication Manager Modes

ID:	BSW09071
Initiator:	DAI
Date:	27.09.2004
Short Description:	Limit Communication Manager modes
Type:	--
Importance:	high
Description:	It shall be possible to limit communication modes independently for each physical channel. An API shall be provided to set the highest available mode. Limiting communication modes takes effect immediately: • If the current mode of a physical channel exceeds the selected limit, the Communication Manager shall take action to set down the communication mode to such limit, as soon as possible. This mode reduction will propagate to all users linked to the affected physical channel. • Requests for modes exceeding the selected limit are not satisfied until the limitation is revoked. Communication mode ordering is described in BSW09083. Passive wake-up shall always be possible, disregarding of the limitation.
Rationale:	This feature is mainly to be used under error conditions, in order to force communication capabilities limitations.
Use Case:	• Prevention of physical channel wake-up. • Forcing into no communication mode. • Force ECU to sleep because it is assumed that this ECU keeps without reason, the physical channel awake or floods it with junk messages.
Dependencies:	[BSW09083] Support of different communication modes. [BSW09157] Revoke Communication Manager mode limitation.
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.3.2.1.15 [BSW09157] Revoke Communication Manager mode limitation

ID:	BSW09157
Initiator:	PSA
Date:	04.10.2005
Short Description:	Revoke Communication Manager mode limitation
Type:	--
Importance:	high

Description:	It shall be possible to revoke a communication mode limitation, independently for each physical channel. An API shall be provided.
Rationale:	This feature is necessary to cancel the effects of the mode limitation service described in BSW09071.
Use Case:	--
Dependencies:	[BSW09083] Support of different communication modes. [BSW09071] Limit Communication Manager modes.
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.3.2.1.16 [BSW09087] Minimum duration of communication request after wakeup

ID:	BSW09087
Initiator:	BMW
Date:	01.12.2004
Short Description:	Proxy communication request after wake-up
Type:	--
Importance:	high
Description:	The Communication Manager shall maintain the state Full com of a communication channel after communication request The minimum duration of the FullCom shall be configurable at pre build Time for each channel
Rationale:	Safeguard against possible irregularities caused by short time span with no communication request e.g. due to delayed functions.
Use Case:	• *Passive wake-up*: keeping the communication stack awake until the application is able to forward the first user request. • *Active wake-up*: keeping the communication stack awake while waiting for NM messages from other ECUs.
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.3.2.1.17 [BSW09089] Preventing waking up physical channels

ID:	BSW09089
Initiator:	BMW
Date:	01.12.2004
Short Description:	Preventing waking up physical channels
Type:	--
Importance:	high
Description:	The Communication Manager shall be able to prevent the host ECU from waking up physical channels. An API to set the wake-up prevention shall be provided.
Rationale:	Prevent constant wake-up from an ECU which is triggered by an error.
Use Case:	--
Dependencies:	[BSW09141] Configuration of physical channel wake-up prevention.
Conflicts:	--
Supporting Material:	--

Contributes to:	--

4.3.2.1.18 [BSW09155] Counting of inhibited communication requests

ID:	BSW09155
Initiator:	DÁI, PSA
Date:	21.09.2005
Short Description	Counting of inhibited communication requests
Type:	--
Importance:	high
Description:	The Communication Manager shall provide a counter for all rejected "Full Communication" mode requests, due to communication mode limitations. The counter shall be stored in non-volatile memory.
Rationale:	The counter is used for detecting latent software problems relating to unmotivated communication bus wake-up.
Use Case:	--
Dependencies:	[BSW09071] Limit Communication Manager modes. [BSW09089] Preventing waking up physical channels.
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.3.2.1.19 [BSW09156] API to retrieve the number of inhibited "Full Communication" mode requests

ID:	BSW09156
Initiator:	DAI, PSA
Date:	21.09.2005
Short Description	API to retrieve the number of inhibited "Full Communication" mode requests
Type:	--
Importance:	high
Description:	It shall be possible to read out (and reset) the number of "Full Communication" mode requests that were inhibited due to communication mode limitations. This value shall be accessible via a Communication Manager API.
Rationale:	--
Use Case:	It shall be possible to read out and reset the current status of the counter by a diagnostic service.
Dependencies:	[BSW09155] Counting of inhibited communication requests.
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.3.2.1.20 [BSW09249] PNC gateway and coordination functionality

ID:	BSW09249
Initiator:	WP1.1.1-EEM
Date:	17.03.2011
Short Description	PNC gateway and coordination functionality
Type:	--

Importance:	high
Description:	Communication Manager shall be aware of the distribution of each PNC over different buses and shall take care to forward the activation requests from one channel to the other.
Rationale:	A PNC can span over different buses which are not necessarily all connected to each involved ECU.
Use Case:	ECU A is connected to CAN A and CAN B, ECU B is connected to CAN B and Flexray A, ECU C is connected to Flexray A; as PNC 1 involves SWCs on all these ECUs, the communication manager on each ECU must know where and how to forward the activation requests.
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.3.2.1.21 [BSW09250] PNC activation requests shall be exchanged with the Network Management by means of User Data

ID:	BSW09250
Initiator:	WP1.1.1-EEM
Date:	17.03.2011
Short Description	PNC activation requests shall be exchanged with the Network Management by means of User Data
Type:	--
Importance:	high
Description:	Communication Manager shall read the User Data in order to get external PNC activation requests and fill User Data in order to forward PNC activation requests according to the specified bit codes. These data are provided/required by the Network Management.
Rationale:	Activation requests can be coded in the Network Management User Data, using in this way already available facilities of the communication stack.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.3.2.1.22 [BSW09251] PNC communication state shall be forwarded to the BswM

ID:	BSW09251
Initiator:	WP1.1.1-EEM
Date:	17.03.2011
Short Description	PNC communication state shall be forwarded to the BswM
Type:	--
Importance:	high
Description:	BswM shall be notified of each change in the communication state of each configured PNC, meaning with this communication mode and relative substate.
Rationale:	This is needed in order to let BswM operate the necessary actions to allow an according operation mode change in the other BSM modules involved.
Use Case:	PNC A is deactivated, I-PDU group A, which contains the I-PDUs corresponding to this PNC, is disabled by a corresponding action in the BswM configuration.
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.3.2.2 Configuration

4.3.2.2.1 [BSW09090] User-to-channel relationship

ID:	BSW09090
Initiator:	BMW
Date:	01.12.2004
Short Description:	User-to-channel relationship
Type:	--
Importance:	high
Description:	Relationship between users and physical channels (which user communicates on which physical channel) are configurable at pre compile time. The usage of additional configuration classes is not restricted.
Rationale:	Necessary for physical channel independency; communication shall be only activated if communication is needed on this physical channel.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	Note: A possible solution is a 2D matrix with users as rows and physical channels as columns; an entry in this matrix links the user (row) with the corresponding physical channel (column).
Contributes to:	--

4.3.2.2.2 [BSW09133] Assigning physical channels to the Communication Manager

ID:	BSW09133
Initiator:	WP Mode Management
Date:	23.02.2005
Short Description	Assigning physical channels to the Communication Manager
Type:	--
Importance:	high
Description:	Physical channels, which should be handled by the Communication Manager, shall be assigned to the module by pre compile time configuration. The usage of additional configuration classes is not restricted.
Rationale:	The ECU specific implementations of the Communication Manager shall have knowledge about how many and which physical channels have to be treated by the Communication Manager on the dedicated ECU. This knowledge is required to enhance the independency of the physical channel management within the Communication Manager. Additionally, it gives flexibility to tailor required data-resources for the specific implementations.
Use Case:	The Communication Manager shall cope with individual communication resources of different ECUs e.g.: ECU A : two FlexRay busses, one CAN bus ECU B : one CAN bus, one LIN bus
Dependencies:	--
Conflicts:	--

Supporting Material:	--
Contributes to:	--

4.3.2.2.3 [BSW09132] Assigning Network Management to physical channels

ID:	BSW09132
Initiator:	WP Mode Management
Date:	23.02.2005
Short Description	Assigning Network Management to physical channels
Type:	--
Importance:	high
Description:	The usage of Network Management shall be assigned to the physical channels by pre compile time configuration. This option shall be provided independently for all physical channels, which are assigned to the Communication Manager. The usage of additional configuration classes is not restricted.
Rationale:	The Communication Manager needs the knowledge of the Network Management usage on each physical channel in order to provide each channel with the suitable synchronization-mechanism to the Network Management (handling of additional indications and acknowledges from/to Network Management). For physical channels without Network Management, the Communication Manager has to be aware of the simplified handling (no indication or acknowledges needed to/from Network Management).
Use Case:	An ECU with two physical channels (A, B): A is related to powertrain domains, B is related to comfort-components. - physical channel A does not use Network Management - physical channel B must use Network Management
Dependencies:	Assigning physical channels to the Communication Manager. This issue is only relevant for assigned physical channels.
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.3.2.2.4 [BSW09141] Configuration of physical channel wake-up prevention

ID:	BSW09141
Initiator:	WP Mode Management
Date:	18.05.2005
Short Description:	Configuration of physical channel wake-up prevention
Type:	--
Importance:	high
Description:	The Communication Manager shall be able to prevent wake-up at the level of physical channels (BSW09089). This feature shall be configurable at pre compile time or post build time. The usage of additional configuration classes is not restricted.
Rationale:	This feature will not appear in all ECUs.
Use Case:	--
Dependencies:	[BSW09089] Preventing waking up physical channels.
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.3.2.2.5 [BSW09243] Handling of Partial Networks on Flexray and CAN

ID:	BSW09243
Initiator:	WP1.1.1-EEM
Date:	16.03.2011
Short Description:	Handling of Partial Networks on Flexray and CAN
Type:	--
Importance:	high
Description:	The Communication Manager shall be able to handle Partial Networks spanning on Flexray and CAN physical channels
Rationale:	Partial Network implementation is needed at least on these bus types.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.3.2.2.6 [BSW09244] Configuration of the number of supported Partial Network Clusters

ID:	BSW09244
Initiator:	WP1.1.1-EEM
Date:	16.03.2011
Short Description:	Configuration of the number of supported Partial Network Clusters
Type:	--
Importance:	high
Description:	The number of supported PNCs shall be configurable strictly at pre-compile time.
Rationale:	Letting complete freedom in configuring the number of PNCs also at Post-Build time would increase complexity in comparison to the advantages.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.3.2.2.7 [BSW09245] Enabling or disabling the Partial Network Cluster management in ComM shall be post-build selectable.

ID:	BSW09245
Initiator:	WP1.1.1-EEM
Date:	16.03.2011
Short Description:	Enabling and disabling the PNC management in ComM shall be post-build selectable.
Type:	--
Importance:	high
Description:	The management of the PNC states shall be post-build activatable.
Rationale:	It has to be possible to disable or enable the management without having to recompile the code.
Use Case:	Activation and deactivation via diagnostic command
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.3.2.2.8 [BSW09207] Configurable Assignment of Bus State Managers

ID:	BSW09207
Initiator:	WP Architecture
Date:	04.06.2008
Short Description:	Configurable Assignment of Bus State Managers
Type:	--
Importance:	high
Description:	ComM shall allow for additional bus specific state managers
Rationale:	Allow writing of CDDs as defined in the AUTOSAR architecture
Use Case:	Use cases for CDDs need not to be given. However, to state some current problems: • CDDs accessing MCAL (e.g. PWM, but call back routines of MCAL only call IOHWA, not a CDD) • CDDs accessing PduR (e.g. Debugging; but PduR only interfaces to Com or Dcm), CDDs accessing CanIf (e.g. OSEK NM or XCP, but there exists a parameter to only select PduR, CanTp or CanNm in CanIf) • new I/O busses besides SPI like USB etc.
Dependencies:	--
Conflicts:	--
Supporting Material:	Covered feature request: o BRF00225 (Enabling CDDs in the BSW architecture)
Contributes to:	--

4.4 Basic Software Mode Manager

4.4.1 Functional Overview

The VFB defines and the RTE implements the concept of Mode Declaration Groups.
There can be multiple Mode Declaration Groups that are independent of each other. E.g. in Figure 1 the Mode Declaration Group is “AMM_KeySwitch” and the Modes in this group are “Off”, “Accessory”, “Ignition”.
A Mode Declaration Group can not contain parallel Modes. Parallel Modes must be implemented by separate Mode Declaration Groups. E.g. diagnostic modes are independent of key switch modes, i.e., they go into a new Mode Declaration Group “AMM_Diagnostics”.
There is no hierarchy of Modes within a Mode Group. E.g. “Diagnostics_On” and “Diagnostics_Off” cannot have sub-modes.

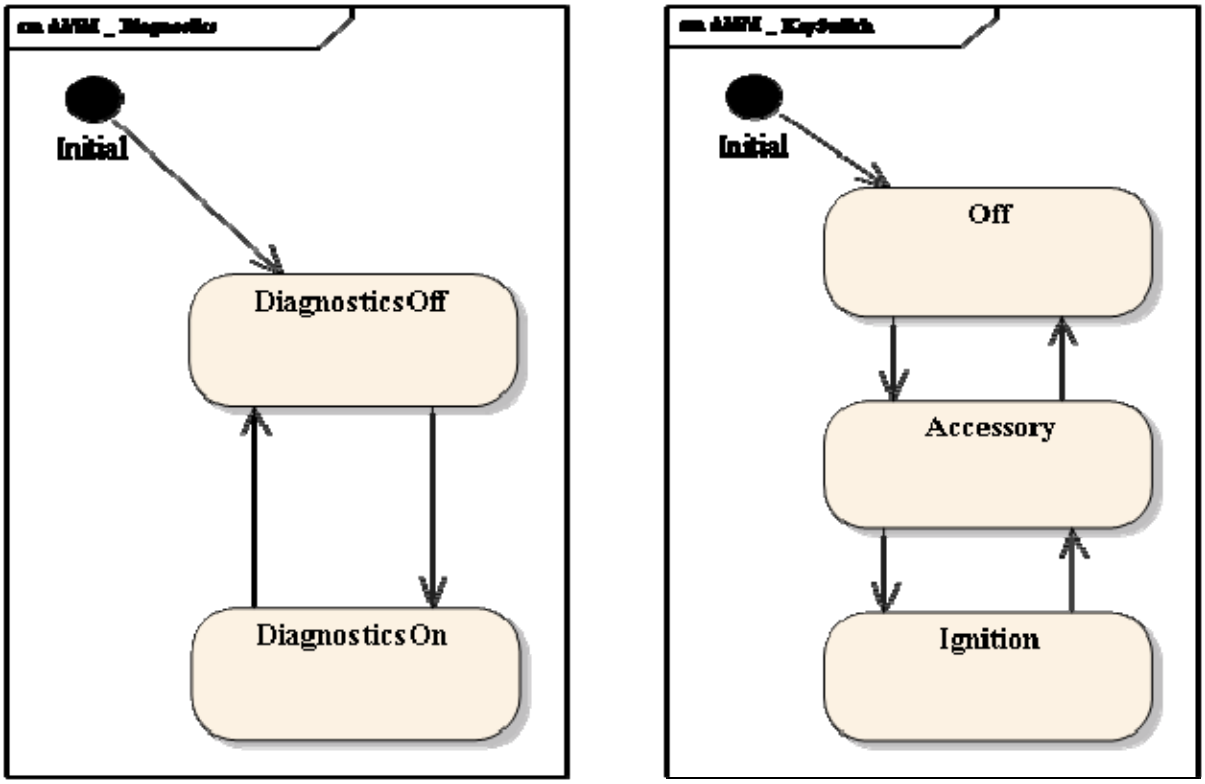

Figure 1 Mode Declaration Groups

4.4.1.1 Interfaces between Mode -Requester, -Manager and -User

As defined above there are three roles that an SW-C or a BSW module can take: Mode Requester, Mode Manager, and Mode User.

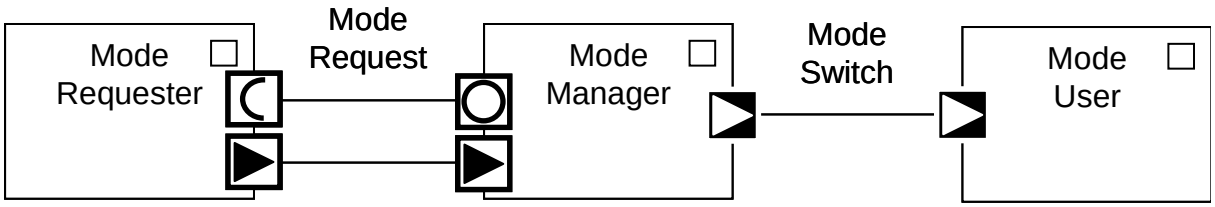

Figure 2 Mode Management Roles and Interfaces

The Mode Requester requests a Mode from a Mode Manager by sending some data via a port with a Mode Request interface. As shown in Figure 2, this port can be a Client or a Sender port, in case of a Mode Manager in the BSW even a C function call. This interface is not standardized.
The Mode Manager receives the incoming information via its Server or Receiver ports or C functions with Mode Request Interfaces, arbitrates the requests and decides upon a resulting mode. It uses a local Sender port with a Mode Switch Interface[2] to switch a Mode Declaration Group into the resulting mode.
To react upon mode changes, Mode Users have a local Receiver port where it receives Mode Switch Notifications. It can either read and evaluate the information directly or via its Software Component Description instruct the RTE to start and stop some of its runnables.
Note that in all cases there may be multiple ports with Mode Request and Mode Switch Interfaces attached to the corresponding roles. E.g., one Mode Requester may request Modes from multiple Mode Managers (This requires the use of a sender-receiver-interface). One Mode Manager may receive requests from multiple Mode Requesters. One Mode Manager may switch multiple Mode Declaration Groups each with multiple Mode Users. And one Mode User may receive Mode Switch Notifications from multiple Mode Managers, where one mode machine instance is switched by only one mode manager.

4.4.1.2 Relation of Modes

Every system contains modes at different levels of granularity. As shown in Figure 3, there are Vehicle Modes and several Applications with modes and ECUs with local BSW Modes.

[2] A Mode Switch Interface is a special kind of Sender-Receiver Interface that is tagged as a local Service and contains a Mode Declaration Group Prototype as a data element.

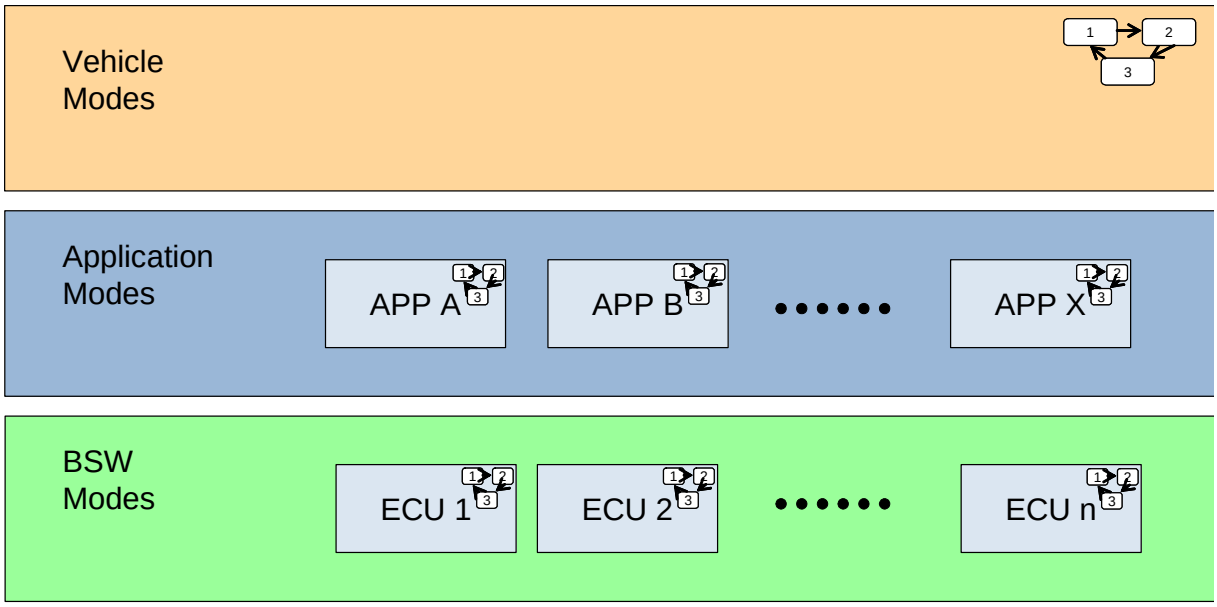

Figure 3 Levels of Modes

But all these modes are not really independent: In reality, Modes at all levels influence each other.

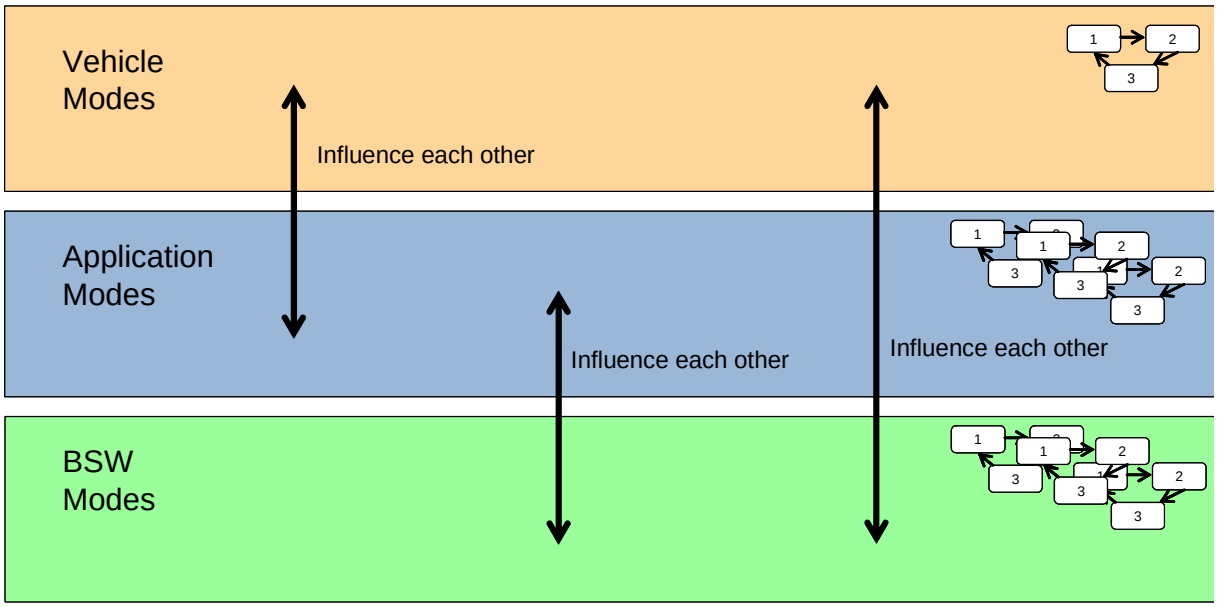

Figure 4 Influences between Mode Levels

Figure 4 shows the following relationships:

- Depending on Vehicle Modes, Applications may be active or inactive and thus be in different Application Modes.
- Vice versa, the operational state of certain Applications may cause Vehicle Mode changes.

- Depending on Vehicle and Application Modes, the BSW modes may change, e.g. the communication needs of an Application may cause a change in the BSW Mode of a communication network.
- Vice versa, BSW Modes may influence the Modes of Applications and even the whole vehicle, e.g. when a communication network is unavailable, Applications that depend on it may change into a Limp-Home Mode.

There are also cross-dependencies within all levels of Modes through mode relations. These get even more complicated because Applications can be distributed over several ECUs.

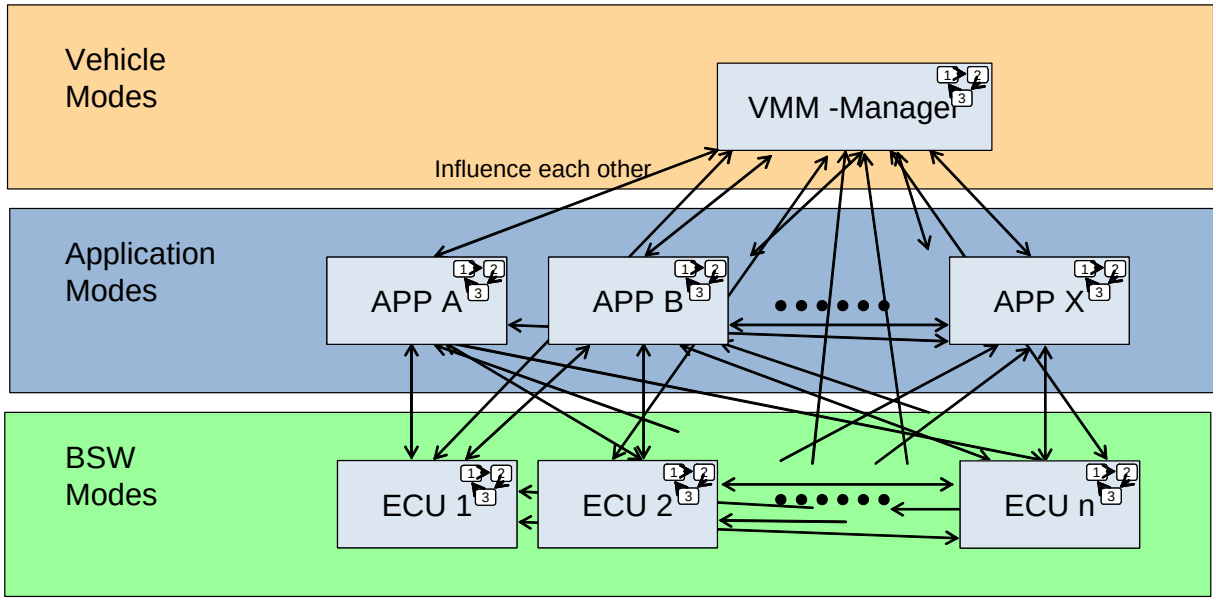

Figure 5 Influences between Modes

Figure 5 gives an impression of the resulting network of complex dependencies.

4.4.2 Functional Requirements

4.4.2.1 Normal Operation

4.4.2.1.1 [BSW09174] Support of 'Disable normal Communication'

ID:	BSW09174
Initiator:	WP Mode Management
Date:	05.06.2008
Short Description:	Support of 'Disable normal Communication'
Importance:	high
Description:	The BSW Mode Manager shall provide the possibility to request modes, where all IPDU groups are disabled except a configurable set of IPDU groups.
Rationale:	It shall be possible to disable normal communication and at the same time keep a predefined set of PDU groups active including diagnostic communication.
Use Case:	Disable normal com, but probably some 'normal' IPDUs shall be still enabled.
Dependencies:	BSW09175, BSW09177
Conflicts:	--
Supporting Material:	Covered feature request: o BRF00045 (The enabling/disabling of IPDU groups shall be done by calling an appropriate interface of COM) o BRF00168 (Support of SAE J1939 Protocol Features)
Contributes to:	--

4.4.2.1.2 [BSW09179] Provision of an Interface to allow Mode Requests of SW-C's

ID:	BSW09179
Initiator:	WP Mode Management
Date:	05.06.2008
Short Description:	Provision of an interface to allow mode requests of SW-C's
Importance:	high
Description:	The BSW Mode Manager shall provide an interface to the RTE. This interface will be used to tell the BSW Mode Manager mode requests which have been issued by a SW-C.
Rationale:	SW-C's shall be able to request mode changes
Use Case:	A SW-C tries to initiate a mode change by telling the request
Dependencies:	BSW09177
Conflicts:	--
Supporting Material:	Covered feature request: o BRF00189 (Enable SW-Cs to request dedicated Modes) This interface will be accessible by SW-C's of other ECU's (see concept requirements for system template and RTE)
Contributes to:	--

4.4.2.1.3 [BSW09228] Provision of an Interface to allow Mode Requests of BSW Modules

ID:	BSW09228
Initiator:	WP Mode Management
Date:	05.06.2008
Short Description:	Provision of an interface to allow mode requests of BSW Modules
Importance:	High
Description:	The BSW Mode Manager shall provide an interface to the BSW Modules and CDD's. This interface will be used to tell the BSW Mode Manager mode requests.
Rationale:	BSW modules shall be able to request mode changes.
Use Case:	A LIN State Manager tells the switch of its schedule table.
Dependencies:	BSW09177
Conflicts:	--
Supporting Material:	Covered feature request: o BRF00168 (Support of SAE J1939 Protocol Features) o BRF00060 (Control Runtime Changeable LIN Schedule Tables)
Contributes to:	--

4.4.2.1.4 [BSW09180] Arbitration of Mode Requests

ID:	BSW09180
Initiator:	WP Mode Management
Date:	05.06.2008
Short Description:	Arbitration of mode requests
Importance:	High
Description:	The BSW Mode Manager shall arbitrate (evaluate) the current mode requests and switch the modes dependant on the result.
Rationale:	Each OEM applies specific rules for mode changes. The resulting state is hold in a state manager like the EcuM, in the RTE or in a SW-C (the BSW Mode Manager is stateless)
Use Case:	--
Dependencies:	BSW09177
Conflicts:	--
Supporting Material:	Covered feature request: o BRF00103 (The enabling of IPDU groups shall be done by calling an appropriate interface of COM) o BRF00060 (The control of the LIN Schedule tables shall be done by calling an appropriate interface of the LIN interface) o BRF00190 (Configurable BSW internal Evaluation of Mode Requests)
Contributes to:	--

4.4.2.1.5 [BSW09182] Local propagation of mode change information

ID:	BSW09182
Initiator:	WP Mode Management
Date:	05.06.2008
Short Description:	Local propagation of mode change information
Importance:	High

Description:	The BSW Mode Manager shall propagate a performed mode change to all local SW-Cs.
Rationale:	All local SW-C shall get the possibility to react on mode changes independent from its role be a dedicated mode change (it is the requester, it has requested an other mode, it has no mode change requested).
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	Covered feature request: o BRF00191 (not really in its current version!)
Contributes to:	--

4.4.2.1.6 [BSW09184] Mode dependent activation and deactivation of IPDU groups

ID:	BSW09184
Initiator:	WP Mode Management
Date:	05.06.2008
Short Description:	Mode dependent activation and deactivation of IPDU groups
Importance:	high
Description:	Depending on arbitration the mode manager shall be able to use a COM interface to activate respectively deactivate I-PDU groups.
Rationale:	To save resources, like network bandwidth and calculation power, only the minimal needed subset of I-PDU groups shall be active (transmitted and received).That means: there shall be no transmission (reception) of I-PDU groups which are not mandatory in the active mode.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	Covered feature request: o BRF00103 (Control of Mode dependent IPDU Groups)
Contributes to:	--

4.4.2.1.7 [BSW09229] Mode dependent callout

ID:	BSW09229
Initiator:	WP Mode Management
Date:	05.06.2008
Short Description:	Mode dependent callout
Importance:	high
Description:	Depending on arbitration the mode manager shall be able to make generic, configured callouts of void functions to other BSW modules.
Rationale:	Dependent on OEM specific modes specific behavior in the BSW will be initiated.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	Covered feature request: o BRF00045 (Support Disable ‘normal’ Communication) o BRF00060 (The control of the LIN Schedule tables shall be done by calling an appropriate interface of the LIN interface) o BRF00103 (Control of Mode dependent IPDU Groups)

	o BRF00104 (Mode Dependent Reset of Initial Values)
Contributes to:	--

4.4.2.1.8 [BSW09230] All actions shall only be performed on mode change

ID:	BSW09230
Initiator:	WP Mode Management
Date:	05.06.2008
Short Description:	All actions shall only be performed on mode change
Importance:	high
Description:	There shall be a mechanism which inhibits that the mode manager performs the configured actions repeatedly without any real change in the reconditions.
Rationale:	A repeated execution of the same actions without a real mode change shall be avoided.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	Technical reason to request that
Contributes to:	--

4.4.2.1.9 [BSW09240] Notification of PNC communication mode change from ComM

ID:	BSW09240
Initiator:	WP1.1.1-EEM
Date:	21.03.2011
Short Description:	Notification of PNC communication mode change from ComM
Importance:	high
Description:	ComM shall notify BswM, if the last is accordingly configured, of any PNC communication state change.
Rationale:	BswM needs to know communication modes of the PNC in order to de-/activate I-PDU groups, issue mode switches etc.
Use Case:	An SWC requests activation of a VFC which is mapped on a PNC spanning over three ECUs. The ECU not hosting the activator SWC receive a request and must activate locally the required I-PDUs, Runnables etc.
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.4.2.1.10 [BSW09241] Request of Communication Mode for a given user

ID:	BSW09241
Initiator:	WP1.1.1-EEM
Date:	21.03.2011
Short Description:	Request of Communication Mode for a given user
Importance:	high
Description:	BswM shall be able to request communication modes for existing CommUsers (i.e. configured in ComM).
Rationale:	BswM needs be able to modify communication modes of channels or PNCs (through their mapped users).
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.4.2.1.11 [BSW09252]Request of Communication Mode for a given Partial Network

ID:	BSW09252
Initiator:	WP1.1.1-MMS
Date:	14.06.2011
Short Description:	Request of Communication Mode for a given Partial Network
Importance:	high
Description:	BswM shall be able to directly request communication modes for the available Partial Networks.
Rationale:	BswM needs to be able to modify communication modes of channels or PNCs in connection to mode requests even without using a Comm User defined for a Partial Network Cluster but directly interacting with the PNC through the ComM.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.4.2.1.12 [BSW09242] Notification of PNC communication mode change from ComM

ID:	BSW09242
Initiator:	WP1.1.1-EEM
Date:	21.03.2011
Short Description:	Notification of PNC communication mode change from ComM
Importance:	high
Description:	ComM shall notify BswM, if the last is accordingly configured, of any PNC communication state change.
Rationale:	BswM needs to know communication modes of the PNC in order to de-/activate I-PDU groups, issue mode switches etc.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--
Contributes to:	--

4.4.2.2 Configuration

4.4.2.2.1 [BSW09177] Support of a configurable mode arbitration

ID:	BSW09177
Initiator:	WP Mode Management
Date:	05.06.2008
Short Description:	Support of a configurable mode arbitration
Importance:	high
Description:	The rules of the mode arbitration shall be pre-compile configurable.
Rationale:	The BSW Mode Manager shall arbitrate mode requests and grant the request only if it is desired and allowed in the current situation.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	Covered feature request: o BRF00103 (Control of Mode dependent IPDU Groups) o BRF00190 (Configurable BSW internal Evaluation of Mode Requests)
Contributes to:	--

4.4.2.2.2 [BSW09178] Support of Lists of Mode Dependant Actions

ID:	BSW09178
Initiator:	WP Mode Management
Date:	05.06.2008
Short Description:	Support of lists of mode dependant actions
Importance:	high
Description:	The lists of mode transition specific actions (including their parameterizations) shall be pre-compile configurable. The order of the listed actions shall cause the sequence of their execution.
Rationale:	Specific for each transition between two modes a specific set of actions shall be performed.
Use Case:	Not until all the actions have been performed, a notification about the entered mode shall be sent (last function call in each transition specific list)
Dependencies:	--
Conflicts:	--
Supporting Material:	Covered feature request: o BRF00045 (Support Disable ‘normal’ Communication) o BRF00060 (The control of the LIN Schedule tables shall be done by calling an appropriate interface of the LIN interface) o BRF00103 (Control of Mode dependent IPDU Groups) o BRF00104 (Mode Dependent Reset of Initial Values) o BRF00191 (Propagation of Mode Information) Actions here are: mode switches, activation/deactivation of I-PDU’s, callouts
Contributes to:	--

4.4.2.2.3 [BSW09175] Support of a configurable Set of Mode dependent enabled and concomitant disabled IPDU groups

ID:	BSW09175
Initiator:	WP Mode Management
Date:	05.06.2008
Short Description:	--
Importance:	high
Description:	The configurable list of mode transition specific function calls shall support mode dependent enabled and concomitant disabled I-PDU groups. It shall be possible to pre-compile and link time configure a set of I-PDU groups to be enabled in a dedicated mode, while all other I-PDU groups will be disabled.
Rationale:	It shall be possible to disable normal communication and at the same time to keep a predefined set of I-PDUs active (including diagnostic communication).
Use Case:	Disable normal com, but probably some I-PDUs shall be still active (e.g. delivering the clamp state)
Dependencies:	BSW09178
Conflicts:	--
Supporting Material:	Covered feature request: o BRF00045 (Support Disable 'normal' Communication)
Contributes to:	--

4.4.2.2.4 [BSW09176] Support of configurable Sets of Mode dependent enabled I-PDU Groups

ID:	BSW09176
Initiator:	WP Mode Management
Date:	05.06.2008
Short Description:	Support of configurable sets of mode dependent enabled I-PDU groups
Importance:	High
Description:	The configurable list of mode transition specific function calls shall support mode dependent enabled I-PDU groups. It shall be possible to pre-compile and link time configure sets of I-PDU groups whose activation depends on the activation of dedicated modes.
Rationale:	It shall be possible to enable I-PDU groups dependant on the active mode.
Use Case:	• If the engine is on, a set of engine controlling I-PDU shall be active.
Dependencies:	BSW09178
Conflicts:	--
Supporting Material:	Covered feature request: o BRF00103 (Control of Mode dependent IPDU Groups)
Contributes to:	--

4.4.2.2.5 [BSW09183] Support of configurable Mode Activation initiated Reset of Signals to Initial Values

ID:	BSW09183
Initiator:	WP Mode Management
Date:	05.06.2008
Short Description:	Support of configurable mode activation initiated reset of signals to initial

	values
Importance:	High
Description:	It shall be possible to pre-compile and link time configure the reset of signal values to their initial values dependent on mode transitions.
Rationale:	In certain power modes, a signal will not be available. To recover quickly, when the signal is available again, it shall be possible to restore an initial value.
Use Case:	• A signal becomes available again after the transporting IPDU (group) has been disabled temporarily in a dedicated mode
Dependencies:	BSW09178
Conflicts:	--
Supporting Material:	Covered feature request: o BRF00104 (Mode Dependent Reset of Initial Values)
Contributes to:	--

5 References

[1] Specification of Communication Manager
AUTOSAR_SWS_COMManager.pdf

[2] Specification of ECU State Manager
AUTOSAR_SWS_ECUStateManager.pdf

[3] Specification of Watchdog Manager
AUTOSAR_SWS_WatchdogManager.pdf

[4] Specification of Communication Stack Types
AUTOSAR_SWS_CommunicationStackTypes.pdf

[5] General Requirements on Basic Software Modules
AUTOSAR_SRS_BSWGeneral.pdf

[6] Layered Software Architecture
AUTOSAR_EXP_LayeredSoftwareArchitecture.pdf

[7] Specification of the Virtual Functional Bus
AUTOSAR_EXP_VFB.pdf

[8] MetaModel
AUTOSAR_MMOD_MetaModel.eap

[9] Software Component Template
AUTOSAR_TPS_SoftwareComponentTemplate.pdf

[10] Specification of System Template
AUTOSAR_TPS_SystemTemplate.pdf

[11] Requirements on CAN
AUTOSAR_SWS_CANDriver.pdf

[12] Requirements on LIN
AUTOSAR_SRS_LIN.pdf

[13] Specification of Operating System
AUTOSAR_SWS_OS.pdf

[14] Feature Specification of the BSW Architecture and the RTE
AUTOSAR_TR_BSWAndRTEFeatures.pdf