COM SOFTWARE ARCHITECTURAL UNDERSTANDING

Embedded Architectural FUNCTIONALITY

The functionality of Com can be broadly divided into three, Transmission, Reception, Signal Gateway and these three can be further divided into their components.

Com

Transmission

Reception

Signal Gateway

State Management

Figure 2: Major Components of Com Module

### Reception

The **reception** sub-module is responsible for unpacking one or more signals received by the Com Module from an I-PDU and generating notification to RTE after successful reception of each signal based on configuration. This module is also responsible to process the signal data for various configuration features of Com like Reception deadline monitoring, checking for invalidation, filtering, replication.

Reception

Rx-Replication

Filtering

Reception

Invalidation

Rx-Deadline Monitoring

TP-Reception

Rx-Counters

Figure 3: Sub Components of Reception

### Transmission

The **transmission** module is responsible for the packing of one or more signals sent by the RTE into corresponding I-PDU and transmission of the I-PDUs to the PduR. This module is also responsible for various features of Com during transmission, like transmission deadline monitoring, selecting I-PDU transmission mode in case of TMS, Considering the Minimum Delay Timer, invalidating signal, and updating I-PDU counters before transmission.

Transmission

Tx-Invalidation

TxCounters

TMC Evaluation

Synchronous Transmission

Asynchronous Transmission

TMS

MDT

Tx-Deadline Monitoring

Timeout Notification

Periodic

N Times Transmission

TP-Transmission

Figure 4: Sub Components of Transmission

|  |  |  |
| --- | --- | --- |
| Component Id | Component Name | Component Description |
| COM\_COMP\_01 | Transmission | A major component of Com module which handles transmission of I-PDUs |
| COM\_COMP\_02 | Synchronous Transmission | The component of ‘Transmission’ responsible for immediate transmission |
| COM\_COMP\_03 | Asynchronous Transmission | The Component of ‘Transmission’ responsible for scheduled transmission. |
| COM\_COMP\_04 | Periodic Transmission | The Component of ‘Asynchronous Transmission’ responsible for transmission at a fixed rate. This implements the ‘PERIODIC’ mode of transmission. |
| COM\_COMP\_05 | N-Times Transmission | The Component of ‘Asynchronous Transmission’ responsible for N-times, burst transmission. This implements the ‘DIRECT N-TIMES mode of transmission. |
| COM\_COMP\_06 | Transmission Mode Selection (TMS) | This Component is responsible for switching between the two modes of transmission for IPDUs. |
| COM\_COMP\_07 | TMC Evaluation | Transmission Mode Condition can be evaluated from filter algorithms that are implemented as part of TMS and filtering at receiver side. |
| COM\_COMP\_08 | Minimum Delay Timer(MDT) | Minimum Delay Timer is responsible for maintaining minimum time period between transmissions of same I-PDU. |
| COM\_COMP\_09 | Tx-Deadline Monitoring | Tx- Deadline Monitoring is responsible for Deadline monitoring mechanism of Com at transmission side. |
| COM\_COMP\_10 | Invalidation | The component of Transmission is responsible for Invalidating signals and signal groups. |
| COM\_COMP\_11 | Tp-Transmission | This Component is responsible to transmit TP I-PDU. |
| COM\_COMP\_12 | Tx-Counter | In transmission side Counter component is responsible for increasing the I-PDU Counter by one before transmitting I-PDU. |
| COM\_COMP\_13 | Reception | A major component of Com module which handles reception of I-PDUs |
| COM\_COMP\_14 | Filtering | This is sub component in Reception which is responsible for filter out the signal based on configured filter algorithm |
| COM\_COMP\_15 | Reception Invalidation | This is responsible to determine whether received value of signal is a valid value or invalid value. If it is invalid value it should perform actions depends on configuration. |
| COM\_COMP\_16 | Reception Deadline monitoring | Reception deadline monitoring is responsible to check the frames are received within the allowed time frame or not. |
| COM\_COMP\_17 | TP-Reception | This component is responsible for receiving larger I-PDUs(TP I-PDUs) |
| COM\_COMP\_18 | Replication | This component is responsible to ensure that received I-PDU data is not lost or corrupted. |
| COM\_COMP\_19 | Rx-Counters | In Reception side Rx-Counter module is responsible for detecting the out of sequence I-PDUs. |
| COM\_COMP\_20 | Signal Gateway | It is responsible to route signals from received I-PDUs to transmit I-PDUs. |
| COM\_COMP\_21 | State Management | It is the component which depicts the state of the Com Module. |

Table 2: Component Description for Com

Concepts Of Execution

### Synchronous Transmission

Synchronous transmission is sub module of Transmission, which is responsible for transmission of I-PDU whenever there is an update in signal value. It is responsible for evaluation of Transmission Mode Selection. In this transmission process various analysis on configuration parameters like transfer property of signal, transmission mode of I-PDU will be carried out.

The Synchronous Transmission component includes

* DET Checks: Any errors classified as development errors will be logged to DET by this part of the routine
* Pack Routine: This part is responsible for packing the signal buffers with the latest signal data. The type of routine executed is determined by the particulars of the signal: endianness, length, position etc.
* Filter Routine: This is responsible for evaluation of configured Filter algorithm with the updated signal data. The execution of this routine shall depend on the configuration.
* Transmission Routine: This routine is responsible for evaluating TMS of an I-PDU, analyzing signal transfer property, checking whether particular I-PDU is enabled or not, and analysis of transmission mode of I-PDU, and it will initiate transmission of I-PDU by invoking PduR\_ComTransmit().

**Transmission of Signal**

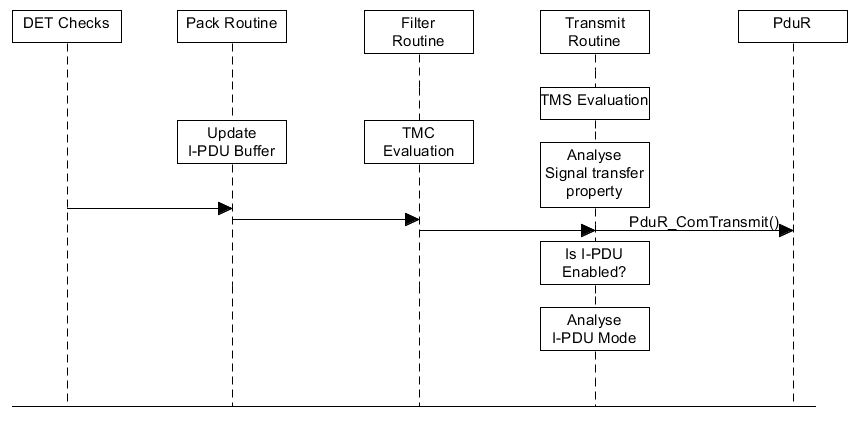
****

Figure 5: Sequence of Events for Transmission of Signal in Synchronous Transmission

**Transmission of Signal Group:**

For transmission of signal groups for which ComSignalGroupArrayAccess is configured as False, Com will provide Shadow Buffer mechanism in order to achieve consistency.

Update of Group Signal includes

* DET Checks: Any errors classified as development errors will be logged to DET by this part of the routine
* Pack Routine: This part is responsible for packing the shadow buffers with the latest group signal data. The type of routine executed is determined by the particulars of the group signal: endianness, length, position etc.
* Filter Routine: This is responsible for evaluation of configured Filter algorithm with the updated group signal data. The execution of this routine shall depend on the configuration

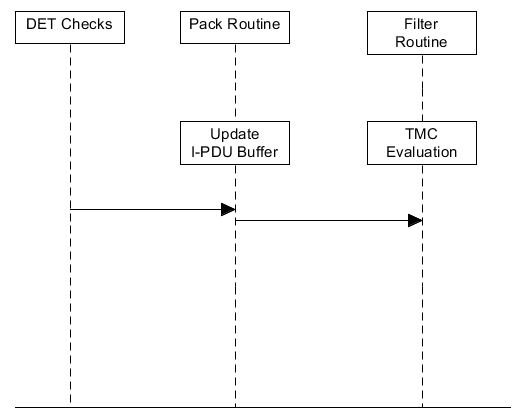


Figure 6: Sequence of Events for Updating Shadow Buffer

After updating group signals RTE will invoke Com\_SendSignalGroup explicitly to trigger the transmission of signal group. The transmission of group signals includes

* DET Checks: Any errors classified as development errors will be logged to DET by this part of the routine
* Pack Routine: In transmission of signal group Pack routine is responsible for packing data from shadow buffer to I-PDU buffer.
* Transmission Routine: This routine is responsible for evaluating TMS of an I-PDU, analyzing signal group transfer property, checking whether particular I-PDU is enabled or not, and analysis of transmission mode of I-PDU, and it will initiate transmission of I-PDU by invoking PduR\_ComTransmit().

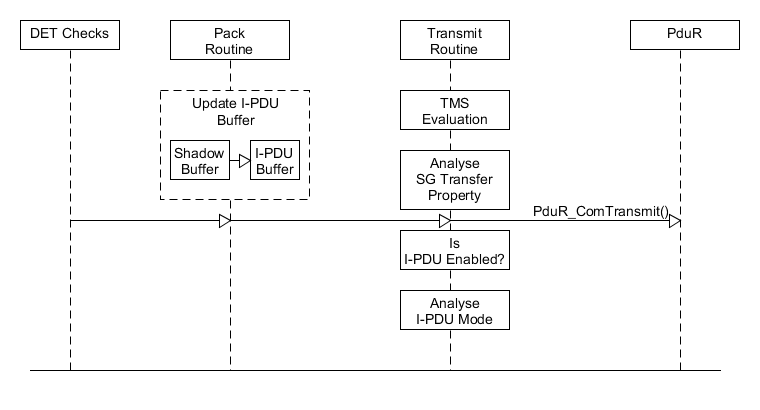


Figure 7: Sequence of Events in Transmission of Signal Group in Synchronous Transmission

### Asynchronous Transmission

Asynchronous transmission is another sub module in transmission which is responsible for transmitting I-PDU in a scheduled manner. The timer period in which transmission takes place is configurable per I-PDU. Before transmission of I-PDU this module should evaluate Transmission Mode Selection based on current value of signals, in addition to that it should consider Minimum Delay Timer, and Deadline Monitoring mechanisms also.

In Com Periodic and Direct N Times transmissions are asynchronous transmission. Updating signal value includes the same routines used in Synchronous transmission. But Transmission routine need not to analyze signal transfer property.

For Periodic Transmission in Com\_MainFunctionTx()

* DET Checks need to be performed for logging any development errors in DET.
* It should check whether particular I-PDU is enabled or not
* I-PDU mode needs to be changed based on TMS evaluation done in Transmit routine
* If Periodic timer of particular timer expires then Com will transmit that particular I-PDU

For Direct N Times Transmission in Com\_MainFunctionTx() in addition to above mentioned points Repetition count will be checked before checking periodic timer, and transmission will takes place as long as repetition count is zero.

**Periodic Transmission**

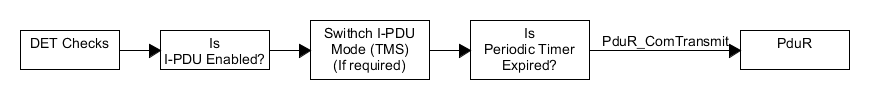


Figure 8: Sequence of Execution in Periodic Transmission

**Direct N Times Transmission**

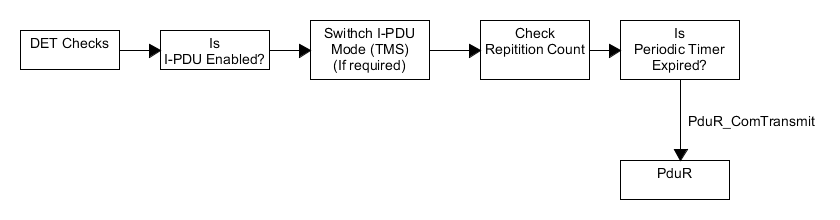


Figure 9: Sequence of Execution in Direct N Times Transmission

### Transmission Mode Selection (TMS)

Transmission Mode Selection (TMS) is a mechanism to derive I-PDU transmission mode from the state of signals that are contained in specific I-PDU. For transmission of I-PDU it is possible to configure two I-PDU modes statically, and selection of mode at specific point in time depends on the value of signals that are mapped to I-PDU. Signals that are contributing for TMS contains configured Filter Algorithm (Transmission Mode Condition), I-PDU mode will be decided based on evaluation of Filter Algorithm.

Transmission Mode Selection includes:

* Pack Routine: The pack routine will be executed during transmission of signal, it will update I-PDU buffer with the latest signal value.
* Filter Routine: This routine evaluates filter condition based on current value of signal and it returns particular filter condition is true or false
* Transmit Routine: The role of transmit routine in TMS is selection of transmission mode of I-PDU, it will select based on result from filter condition evaluation.

Formula for calculating overall TMS of I-PDU is

TMS = Filter1 || Filter2 || Filter3

If all the Filter conditions are false then I-PDU mode will be switched from True mode to false mode and vice versa.

**Transmission Mode Selection**

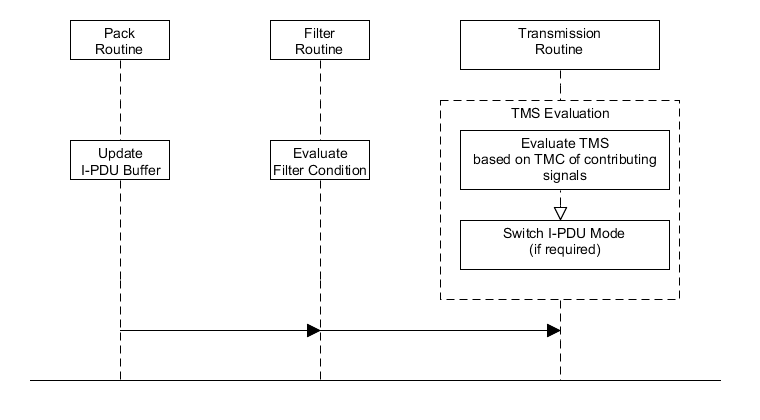
****

Figure 10: Sequence of Execution in Transmission Mode Selection

### Minimum Delay Time (MDT)

A minimum delay time is the delay between two consecutive transmit requests on the network. A minimum delay time between transmissions is configured per I-PDU. The Com module shall load and start the minimum delay time counter upon transmission of that I-PDU to the PDU Router via PduR\_ComTransmit. If a transmission is requested before MDT expires the transmission get postponed until the MDT expires.

Transmit routine is responsible for loading MDT timer after transmission of I-PDU. In the successive calls of Com\_MainFunctionTx MDT timer will be decremented.

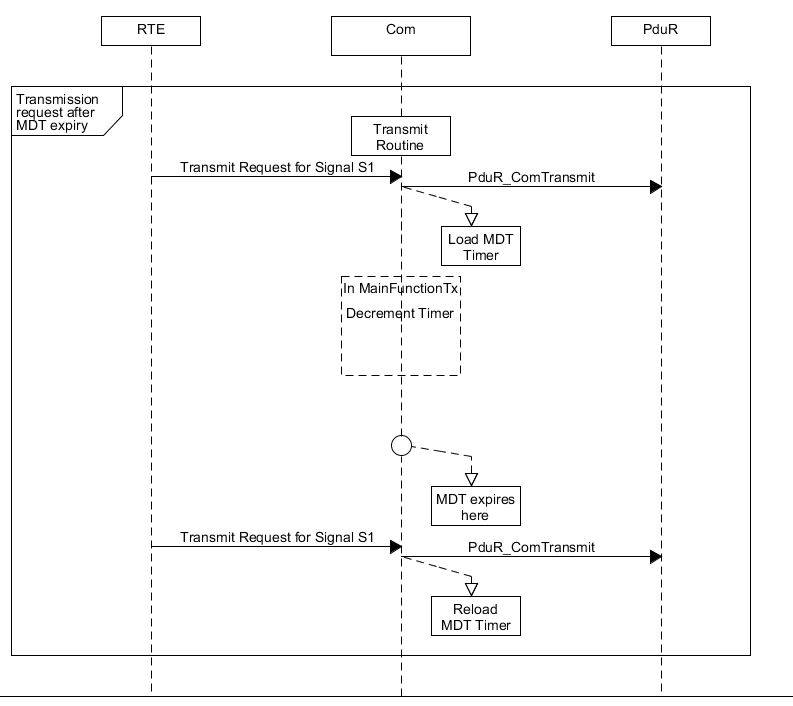
****

Figure 11: Transmission request after MDT expires

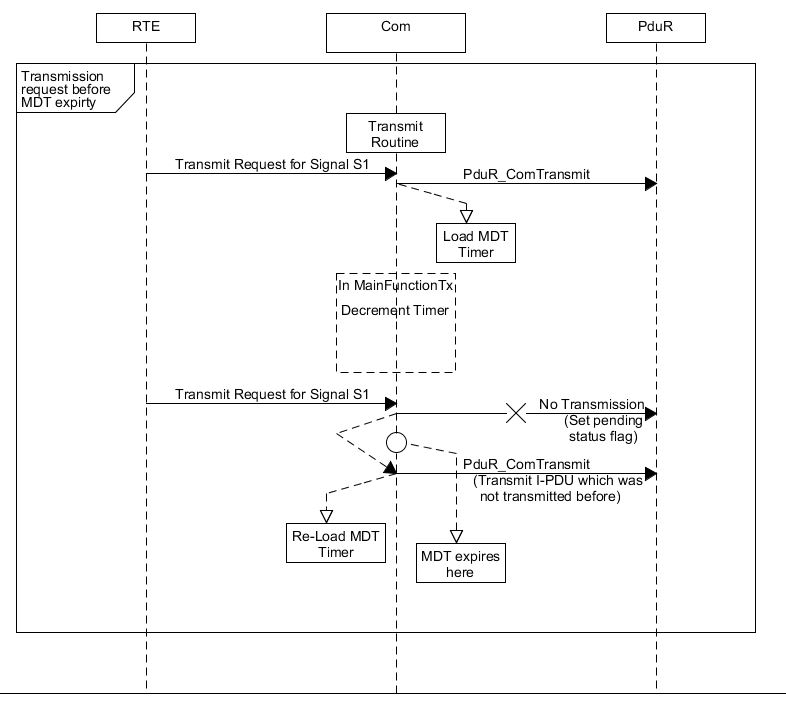


Figure 12: Transmission request before MDT expires

### Transmission Deadline Monitoring

The deadline monitoring mechanism monitors an I-PDU that is transmitted within the given time interval. If the I-PDU is configured for deadline monitoring and transmit confirmation is not obtained within the given period from underlying layer, signals configured for timeout notification in that I-PDU will be notified. If the transmit confirmation is received from underlying layer, respective deadline monitoring timers will be cancelled for that I-PDU.

Transmit routine is responsible for loading the Deadline Monitoring Timer upon transmission of I-PDU. And timer will be decremented in successive calls of Com\_MainFunctionTx.

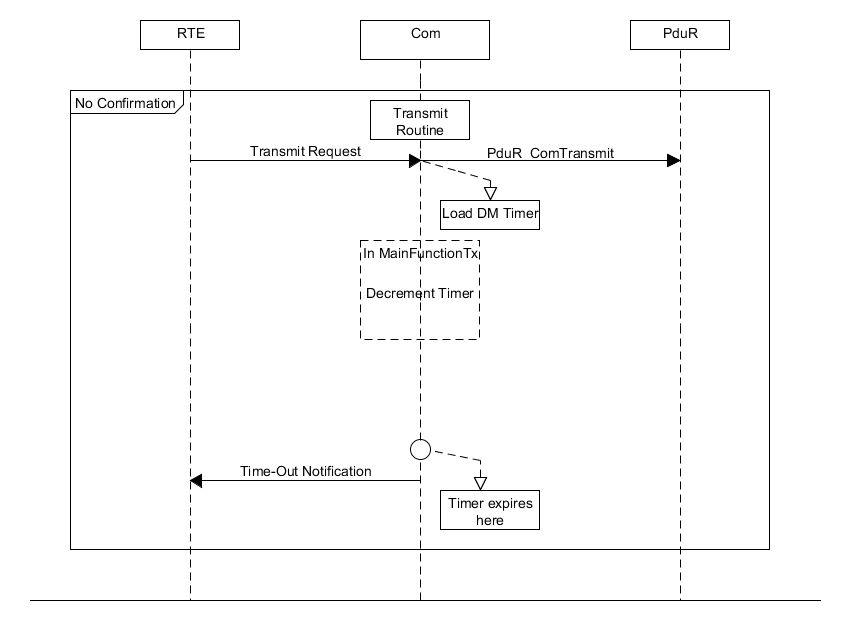


Figure 13: Time out Notification

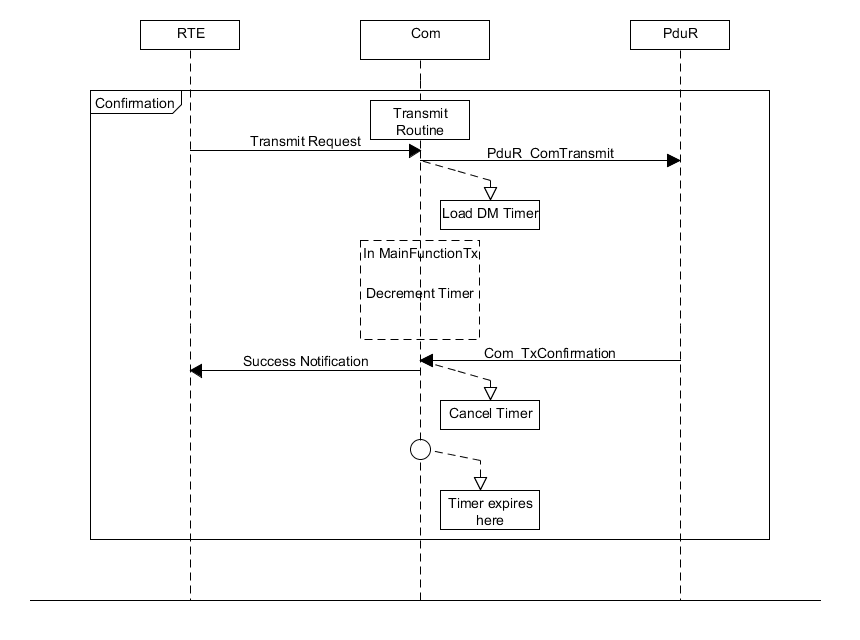


Figure 14: Success Notification

* + 1. Tx-Invalidation

It is possible for the sender side to indicate that it is not able to provide a valid value for a corresponding signal via Com\_InvalidateSignal(), Com\_InvalidateShadowSignal() and Com\_InvalidateSignalGroup() APIs for signal, signal in group and signal groups respectively. Data invalid value can be configured for each signal during configuration. The Com Module replaces the current value of the signal with the data invalid value if the upper layer indicates that it is not able to provide a valid value. Com Module internally calls Com\_SendSignal() or Com\_UpdateShadowSignal() APIs with the configured data invalid value to update the signal value with the invalid value in I-PDU/Shadow buffer.

Invalidation process involves

* DET Checks: Any errors classified as development errors will be logged to DET by this part of the routine
* Invalidation routine: This routine is responsible for fetching configured invalid value of signal or group signal.
* Pack Routine: Pack routine packs the data into I-PDU buffer or shadow buffer with invalid value, the type of routine executed is determined by the particulars of the signal: endianness, length, position etc.
* Filter Routine: Filter routine will evaluate configured filter condition with invalid value of signal or group signal.
* Transmit Routine: The role transmit routine here is same as synchronous/asynchronous transmission.

**Invalidation**

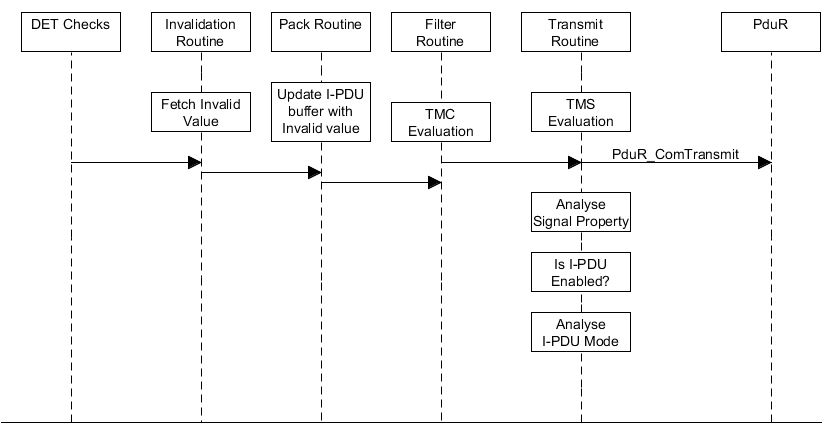
****

Figure 15: Sequence of Events in Invalidation of Signal

* + 1. TP-Transmission

The AUTOSAR COM module sends large I-PDUs via the generic PduR APIs for upper layer modules that use transport protocol.

The sequence to transmit large I-PDUs is

1. Initiation of Send request: PduR\_ComTransmit

2. One or more calls to copy data: Com\_CopyTxData

3. Transmission confirmation: Com\_TpTxConfirmation

TP transmission includes

Transmission Routine: Transmit routine is responsible for initiating transmission request by invoking PduR\_ComTransmit, after initiation it will lock TP I-PDU buffer.

CopyTxData: PduR will initiate CopyTxData to copy data from I-PDU buffer to TP module buffer, and it is responsible for providing information on remaining number of bytes yet to copy.

Com\_TpTxConfirmation: After successful transmission of data, PduR will give confirmation, then locked Tp I-PDU buffer will be released.

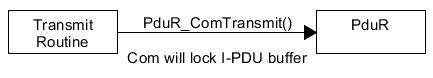


Figure 16: Initiation of Send Request during Tp Transmission

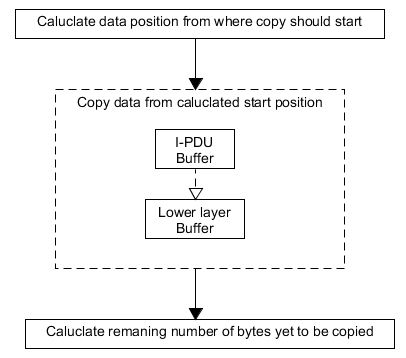


Figure 17: Copy Tx Data in TP- Transmission

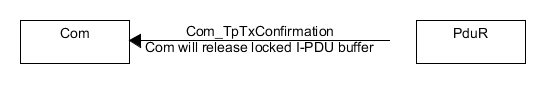
****

Figure 18: Transmission Confirmation

* + 1. Filtering

The **‘Filtering’** mechanism is responsible to filter out signals on the reception side based on the configured Filter condition (Filter Algorithm). For each signal a different filtering condition can be configured through a Filter algorithm.The received signal data will be evaluated as per filter algorithm and the signal will be filtered out if the filter result is fail.

Com Module provides the following filter algorithms

* ALWAYS
* NEVER
* MASKED\_NEW\_EQUALS\_X
* MASKED\_NEW\_DIFFERS\_X
* MASKED\_NEW\_DIFFERS\_MASKED\_OLD
* NEW\_IS\_WITHIN
* NEW\_IS\_OUTSIDE
* ONE\_EVERY\_N

Filtering process involves:

DET Checks: Any errors classified as development errors will be logged to DET.

I-PDU Group Status check: To ensure that received I-PDU is enabled

Copy Routine: Com will update its internal I-PDU buffers with the received data.

If for received I-PDU ComIPduSignalProcessing is configured as IMMEDIATE then signal will be processed immediately, if it is DEFERRED it will be processed in next MainFunction Rx.

Signal Processing: Data of the signal will be extracted from received I-PDU.

Filter Routine: The received value of the signal will be evaluated as per the Filter Algorithm

If Filter condition is true signal data will be copied into signal buffer.

If Filter condition is false signal will be filtered out.

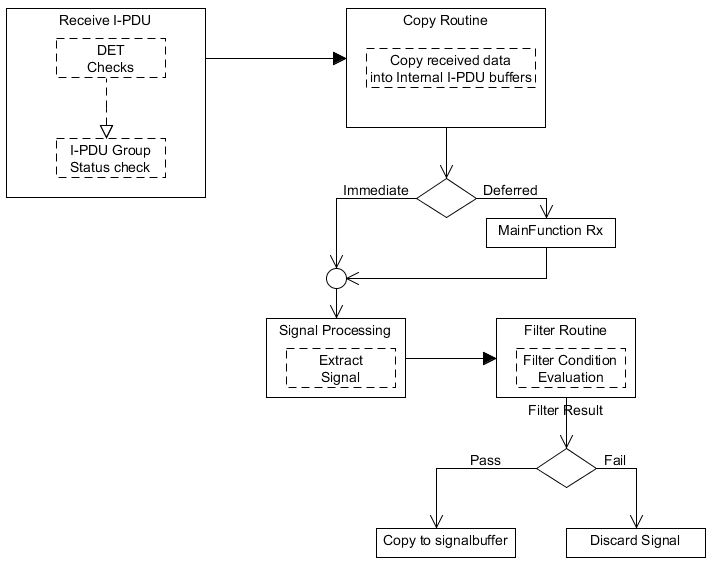


Figure 19: Filtering of Rx Signal

* + 1. Reception validaton

The reception validation component is responsible to determine whether the received signal or group signal data is valid value or invalid value.

Process to determine this is as follows:

DET Checks: Any errors classified as development errors will be logged to DET.

I-PDU Group Status check: To ensure that received I-PDU is enabled

Copy Routine: Com will update its internal I-PDU buffers with the received data.

If for received I-PDU ComIPduSignalProcessing is configured as IMMEDIATE then signal or group signal will be processed immediately, if it is DEFERRED it will be processed in next MainFunction Rx.

Signal Processing: Data of the signal or group signal will be extracted from received I-PDU.

Validate Routine: This routine will check whether the received signal or group signal data is valid or invalid. If the data is valid then signal or group signal data will be copied into signal buffer. If the data is invalid and the configured invalid action is REPLACE Com will update signal buffer with initial value. If the Configured invalid action is NOTIFY Com will notify RTE by invoking invalid notification.

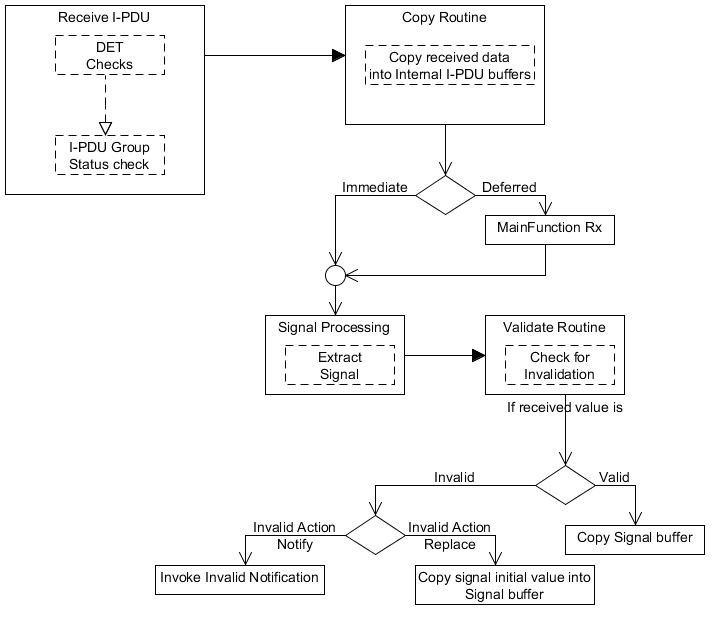


Figure 20: Reception of Invalid Signal or Group Signal

* + 1. Reception Deadline Monitoring

Reception deadline monitoring is used to verify that I-PDU is received within the allowed time frame or not. This mechanism can be configured per signal and is performed by monitoring the reception of the I-PDU that contains the signal. In case where deadline monitoring is configured for signals with update bits, separate deadline monitoring for each signal with an update bit will be performed. The deadline monitoring mechanism monitors that I-PDU or signal with an update bit is received within a given time interval. The monitoring timer will be cancelled and restarted upon each new successful reception from the underlying layer of the I-PDU that contains the signal. If there is no indication of the I-PDU reception by the underlying layer, the timeout occurs and the appropriate action will be taken immediately as per configuration of parameter ComRxDataTimeoutAction. Deadline monitoring timer will be re-started after the timeout.

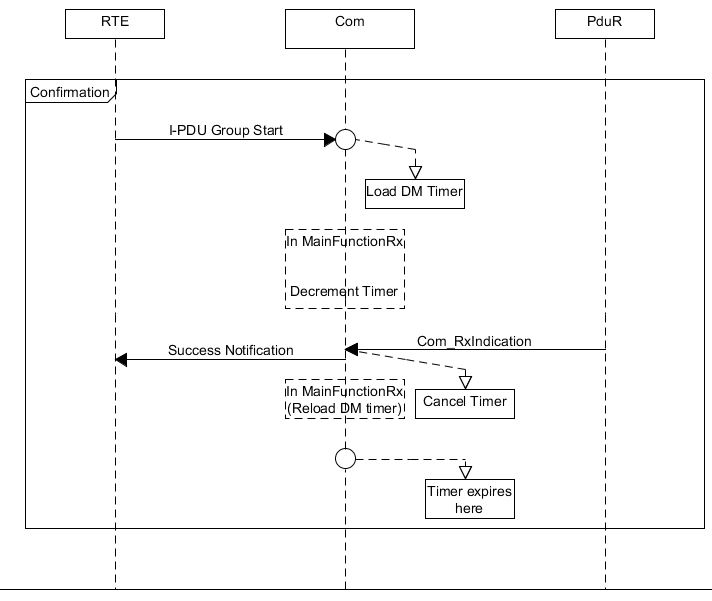
****

Figure 21: I-PDU based Deadline Monitoring

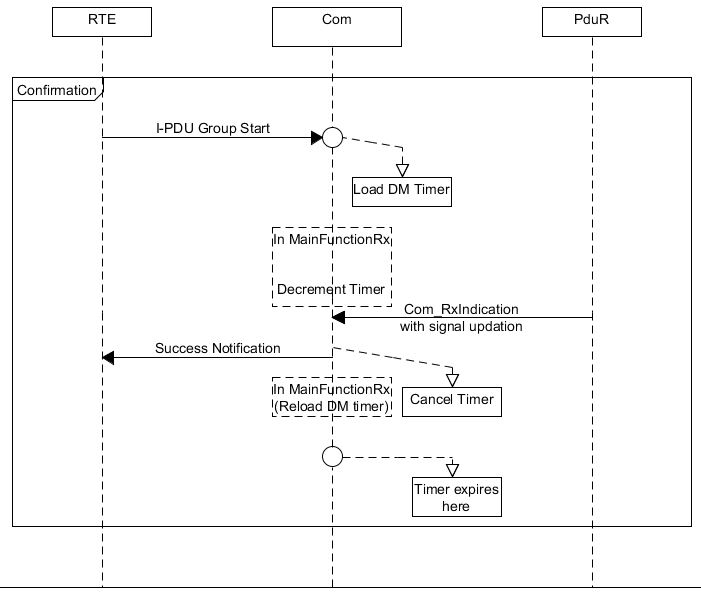
****

Figure 22: Signal level Deadline Monitoring

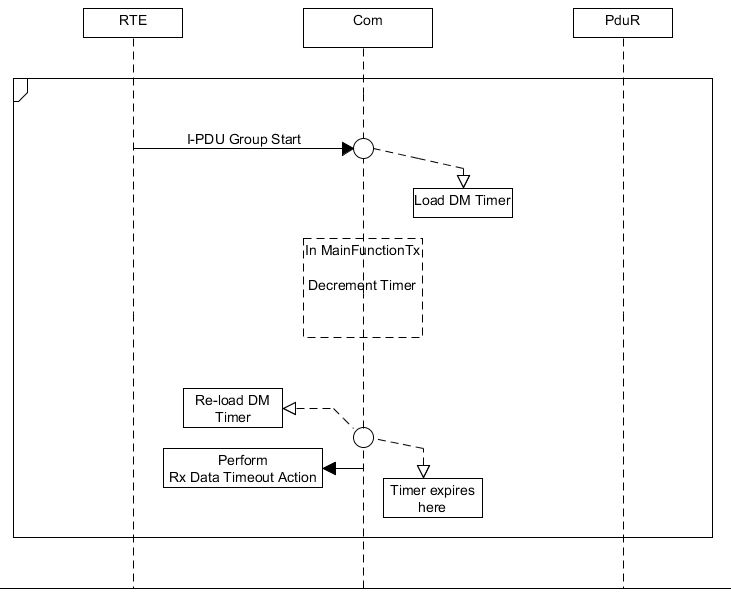
****

Figure 23: Reception Deadline Monitoring Expiry

* + 1. TP-Reception

The AUTOSAR COM module receives large I-PDUs via the generic PDU Router’s APIs for upper layer modules that use transport protocol

The sequence to receive large I-PDU is

1. Initiate RX request: Com\_StartofReception

2. One or more call backs to copy received data from lower layer buffer: Com\_CopyRxData

3. Indication of correct or incorrect received data and end of reception process: Com\_TpRxIndication

The TP-Reception process includes

TP reception routine: It is responsible to acknowledge the reception request and locking of TP I-PDU buffer. If first frame or metadata is received in this call data will be copied into internal I-PDU buffer.

CopyRxData: PduR will invoke multiple calls of CopyRxData to copy data from lower layer to Com internal I-PDU buffers. It is also responsible to provide available receive buffer after data has been copied.

Com\_TpRxIndicaiton: PduR will invoke Com\_TpRxIndication indicating the end of reception process, and correctness of data. Here locked I-PDU buffer will be released, and data will be processed further depends the correctness of the data.

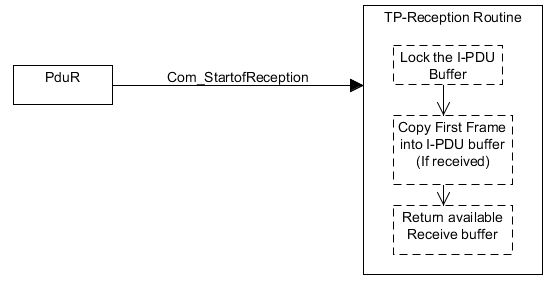
****

Figure 24: Initiation of Reception Request in TP- Reception

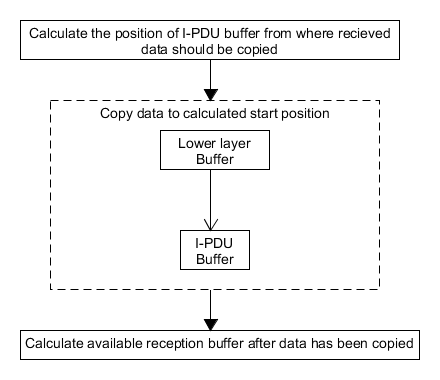


Figure 25: Copy Rx Data in TP-Reception

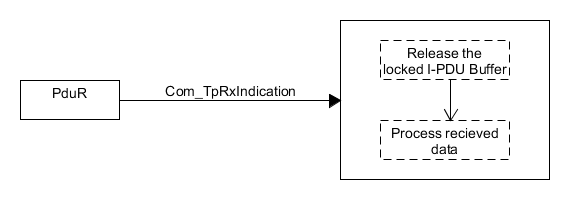


Figure 26: End of Reception Process in TP-Reception

* + 1. Rx-Replication

The replication of I-PDUs in multiple L-PDUs is supported in order to prevent corruption and loss of data. The comparison and voting of replicated I-PDUs is performed in COM Module. COM Module allows configuring two additional replicated I-PDUs and the number of successfully equally received I-PDUs needed for a qualified reception. At receiver side, COM Module stores the replicated I-PDUs and performs a voting about the multiple replicated I-PDUs. Only when the identical copies of a replicated I-PDU are more than the Quorum, COM Module shall provide the signals and/or signal groups out of this I-PDU to the RTE.

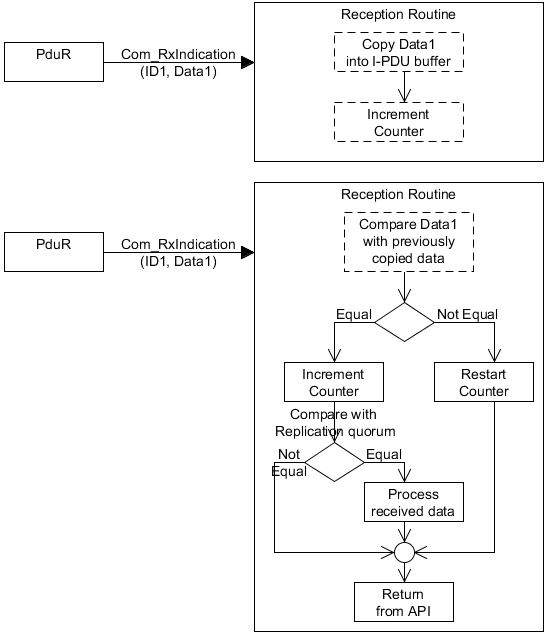


Figure 27: Rx Replication

* + 1. Tx Counters

The Com module provides mechanisms of data sequence control in the form of I-PDU counters. During transmission Com will increment the counter value in I-PDU before transmission.

Transmit

Request

Increment

Counter

Perform

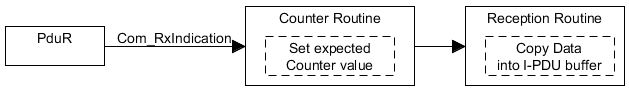
Transmission

Figure 28: Counter functionality in Transmission Side

* + 1. Rx-Counters

In Reception side Rx-Counter module is responsible for detecting and discarding out of sequenc*e* I-PDUs, those are repeated I-PDUs or I-PDUs received after missing I-PDUs.

**First Reception**

****

**Successive Receptions**

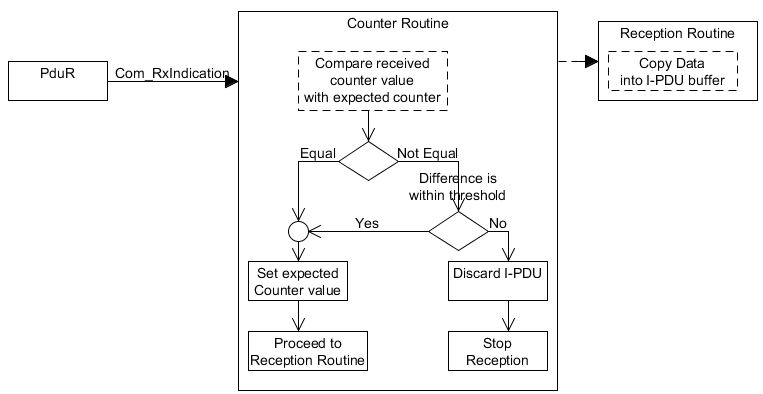
****

Figure 29: Counter functionality in Reception Side

* + 1. Signal Gateway

The **‘Signal Gateway’** component of Com is responsible for the said functionality. It takes data of signals from received I-PDUs which are configured as gateway source, and it updates data of destination signals that are mapped to Transmission I-PDUs.

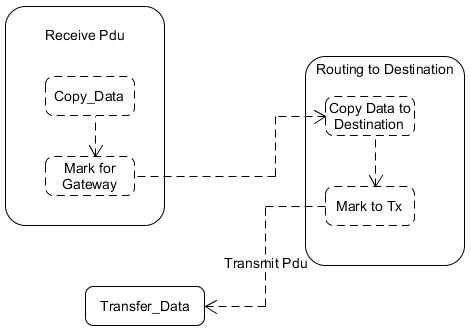


Figure 30: Com Signal Gateway

* + 1. State Management

This module provides the functionality of initialization and deinitialization. This module provides Com\_Init(), Com\_DeInit() and Com\_GetStatus() APIs.

* Com\_Init() service initializes internal buffers and network interfaces of the Com Module for the further processing. After Com\_Init(), the inter-ECU communication is still disabled. Inter-ECU communication requires separate start.
* Com\_DeInit() service provides the functionality of de-initializing the Com Module. This means that communication between the ECUs is not possible, after de-initialization of the Com Module, all I-PDU groups, which have been started earlier, are stopped.
* Com\_GetStatus() service provides the status of the Com Module whether it is initialized or uninitialized.

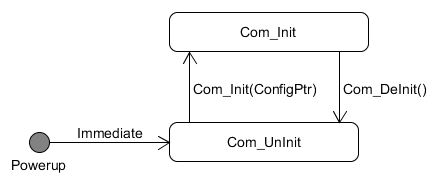
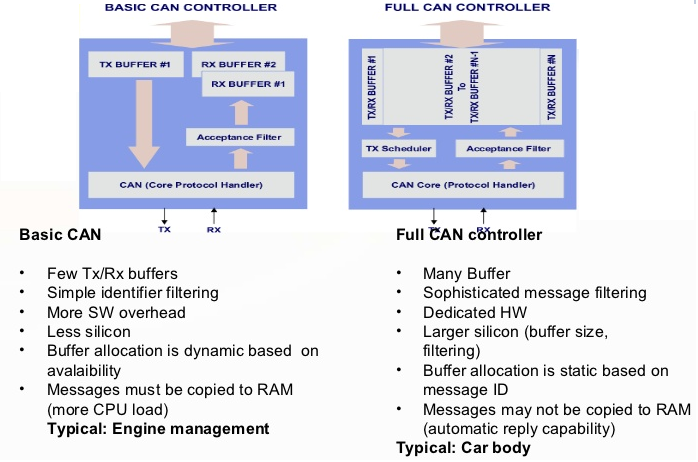


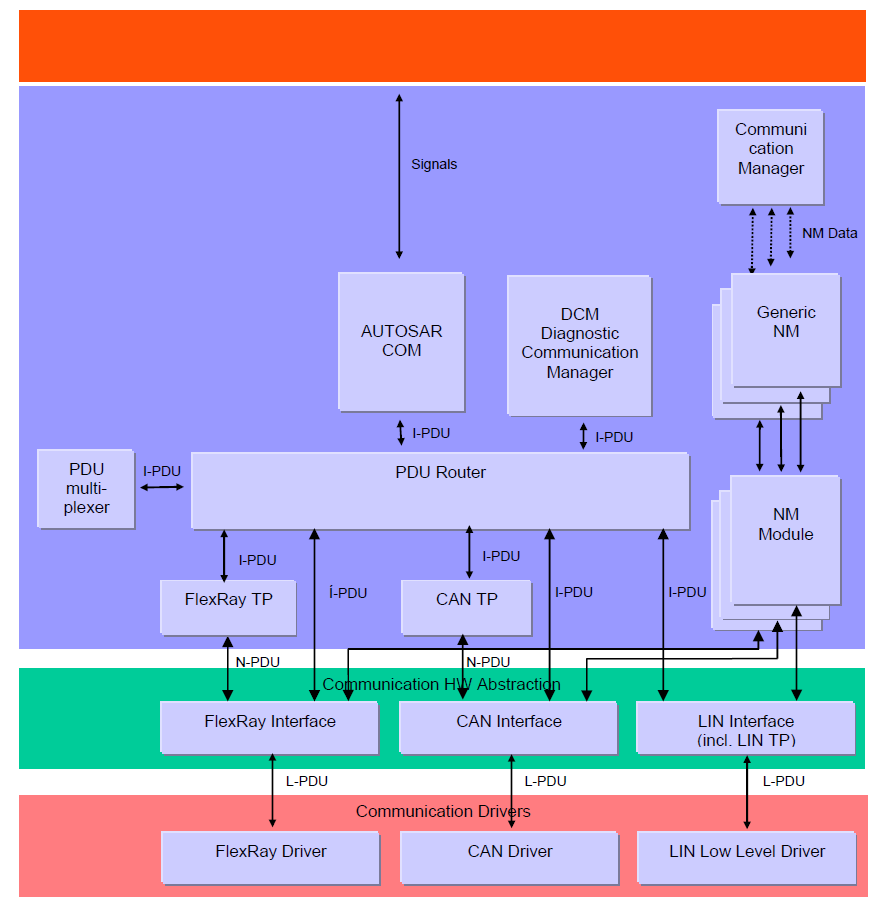
Figure 31: Com State Management

**Difference between Full CAN and Basic CAN?**

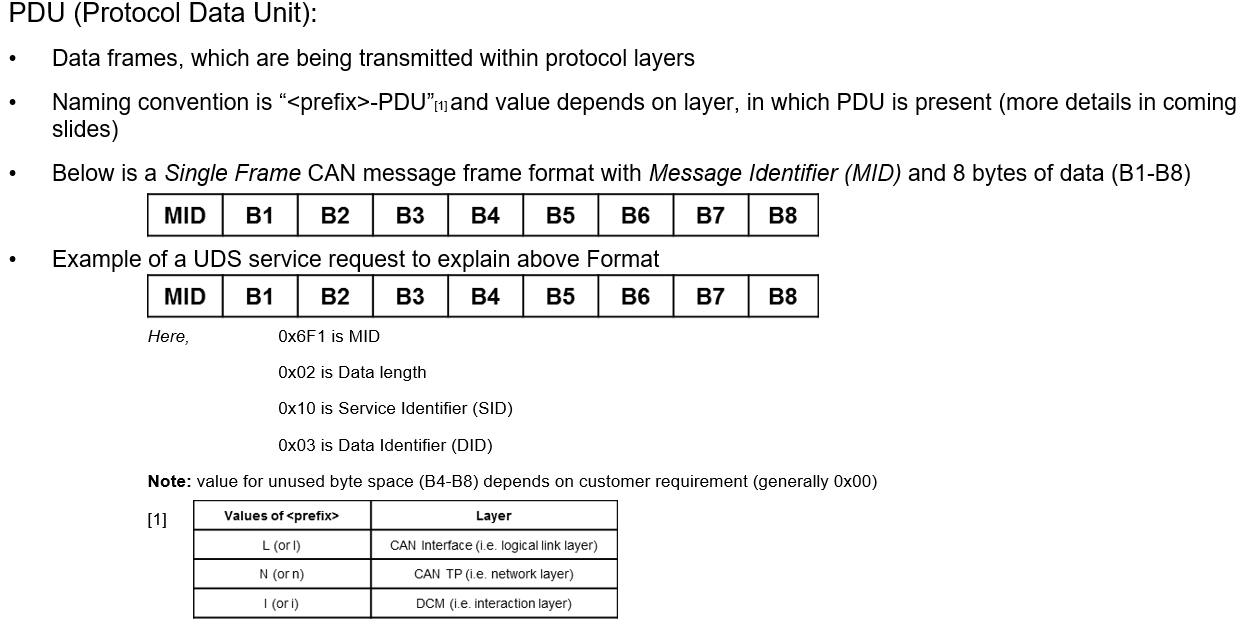
****

**Introduction to AUTOSAR COM Stack**

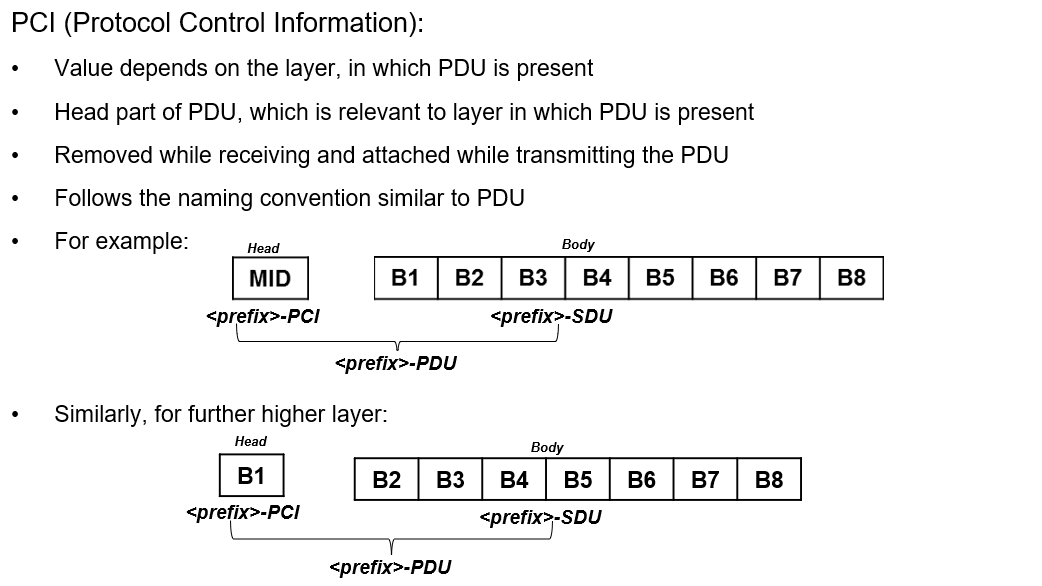
* AUTOSAR COM Stack is a software stack that provides communication to the basic software modules and application layer.
* Depending on the bus type of the in-vehicle network (CAN, LIN, Flex-ray, MOST), implementation of the communication stack is executed.
* A generic communication stack in AUTOSAR layered architecture will have the following parts :
  1. AUTOSAR COM
  2. Bus specific interface modules
  3. External Bus Drivers
  4. Internal Bus Drivers

****

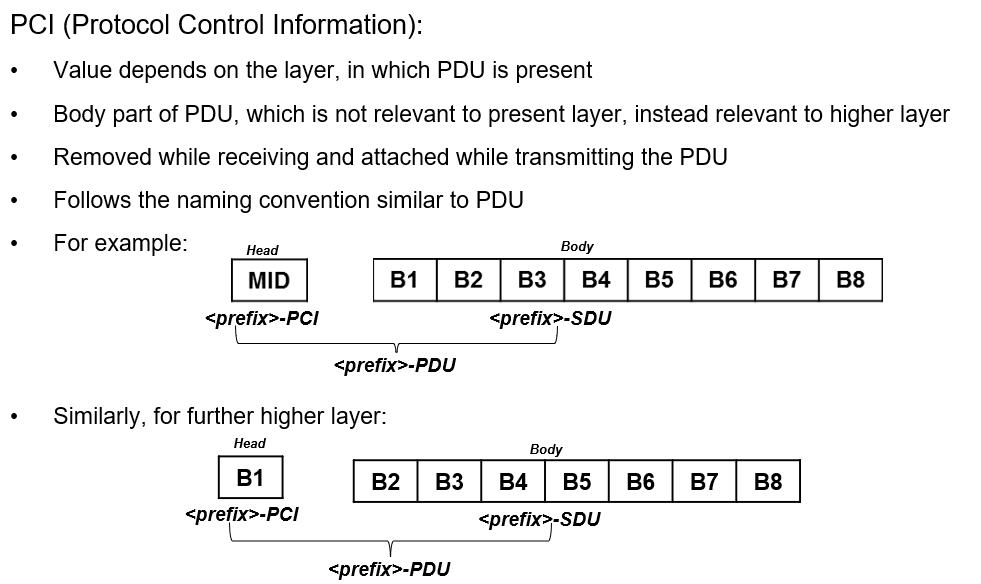
**PDU**



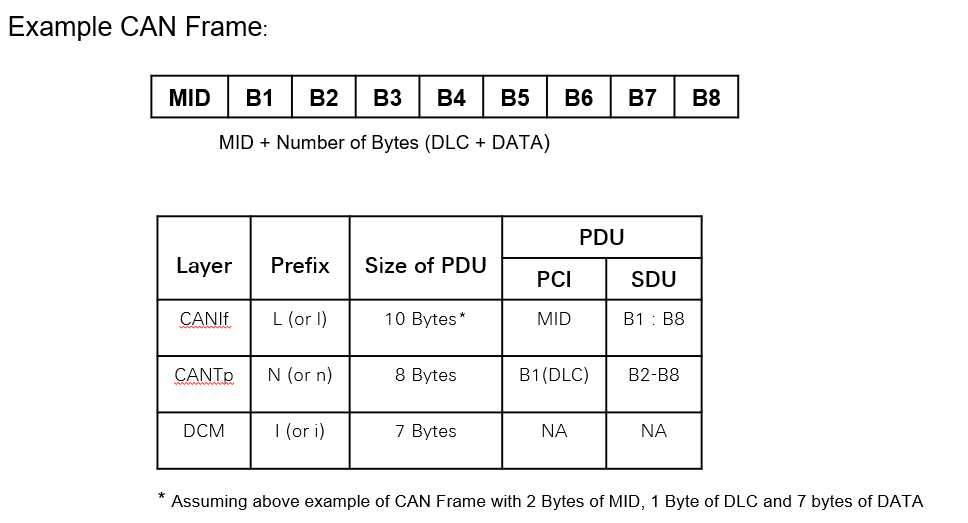
**PCI**



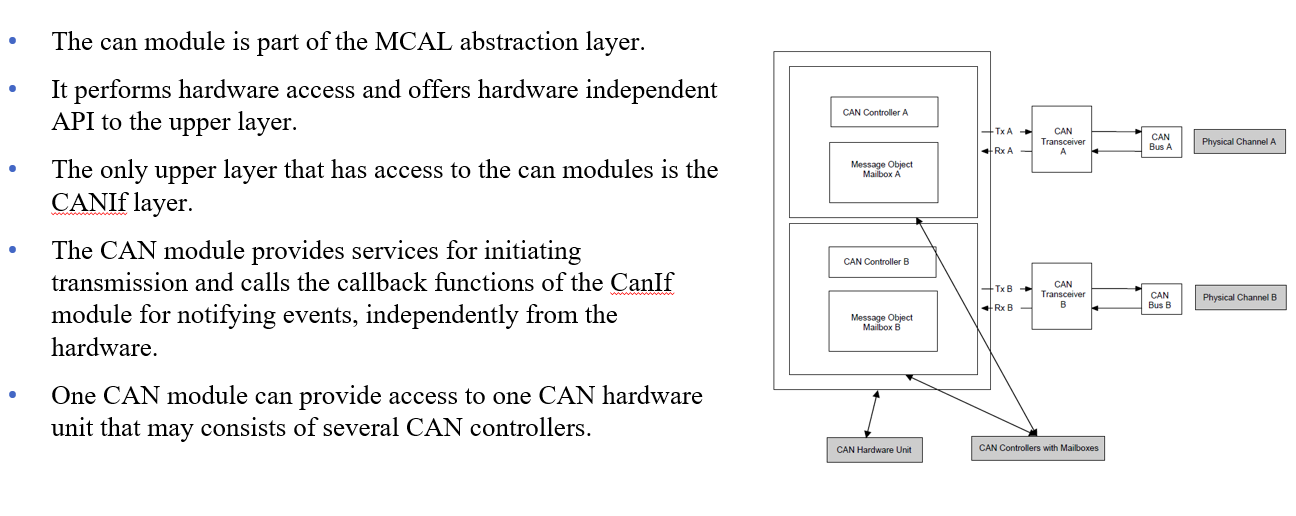
**SDU**

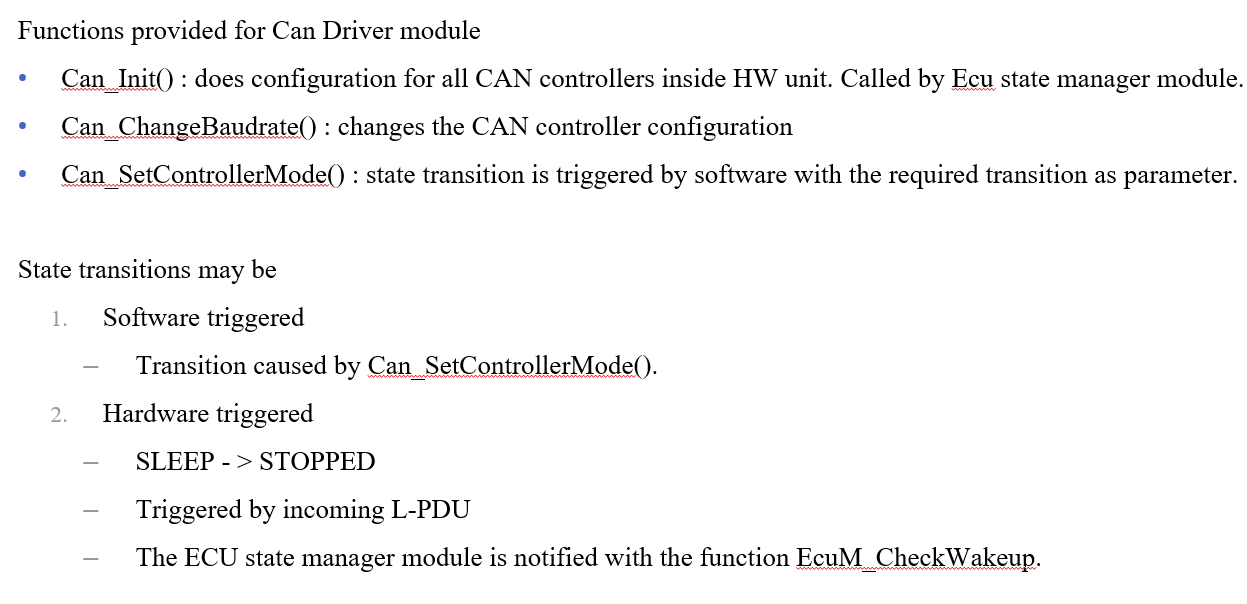


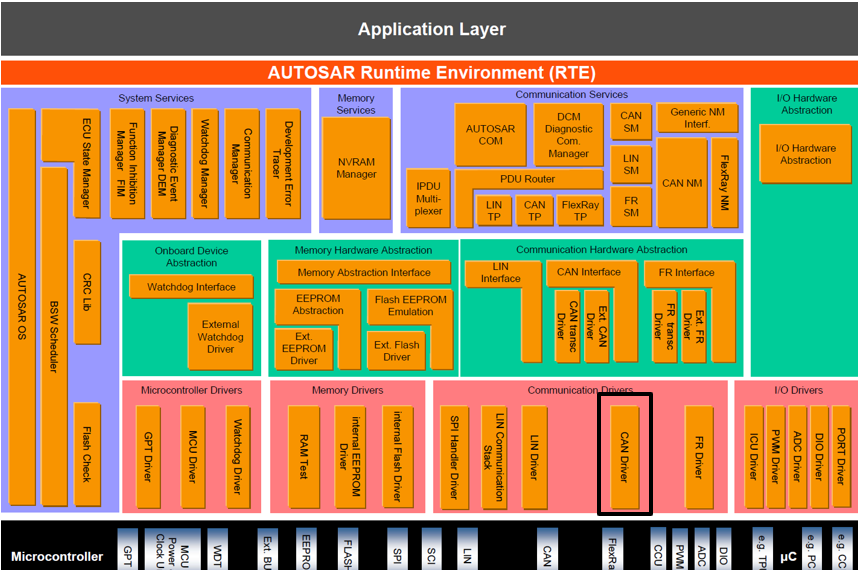
**PDU Breakout**



**CAN Driver**







**Priority Inversion in CAN**

Inner priority inversion

* This occurs when only single transmit buffer is used.
* Because of a low priority message stored in the buffer and huge traffic on the bus, a higher priority message wont be transmitted over the bus.

Outer Priority inversion

* It occur when the inter-frame space between two consecutive messages stored in different message buffers is longer than the minimum space defined by the CAN standard, a second node is able to transmit a lower priority message.

To prevent Priority inversion we have two mechanisms

* + Multiplexed transmission
  + Transmit cancellation

**Mechanisms to prevent Priority inversion in CAN**

1. Multiplexed Transmission
   * Several transmit hardware objects shall be assigned by one transmit handle to represent one transmit entity to the upper layer.
   * Multiplexed transmission mechanisms shall be supported for devices where either
     + Multiple transmit hardware objects, which are grouped to a single transmit entity can be filled over the same register set and the controller stores the L-PDU into a free buffer autonomously.
     + Or hardware provides registers or functions to identify a free transmit hardware object within a transmit entity.
2. Transmit Cancellation
   * Sometimes it is always required to transmit the newest data on the bus always.
   * The L-PDUs which are pending in the transmit buffer from the previous transmit cycle will be replaced with L-PDUs of the current transmit cycle.
   * This requirement is supported by cancellation of pending PDUs with identical priority.

**CAN Interface**

* The CAN interface module is a part of ECU abstraction layer.
* The CAN Interface module is located between the low level CAN device drivers and the upper communication layer services.
* It acts as an interface between the CAN driver (external or internal) and upper service layer.
* CanIf optionally filters out L-PDU and prohibits further software processing
* The basic functionality of the CanIf layer is
  1. Initialization
  2. Transmit request services
  3. Transmit confirmation services
  4. Reception indication services
  5. Controller mode control services
  6. PDU mode control services

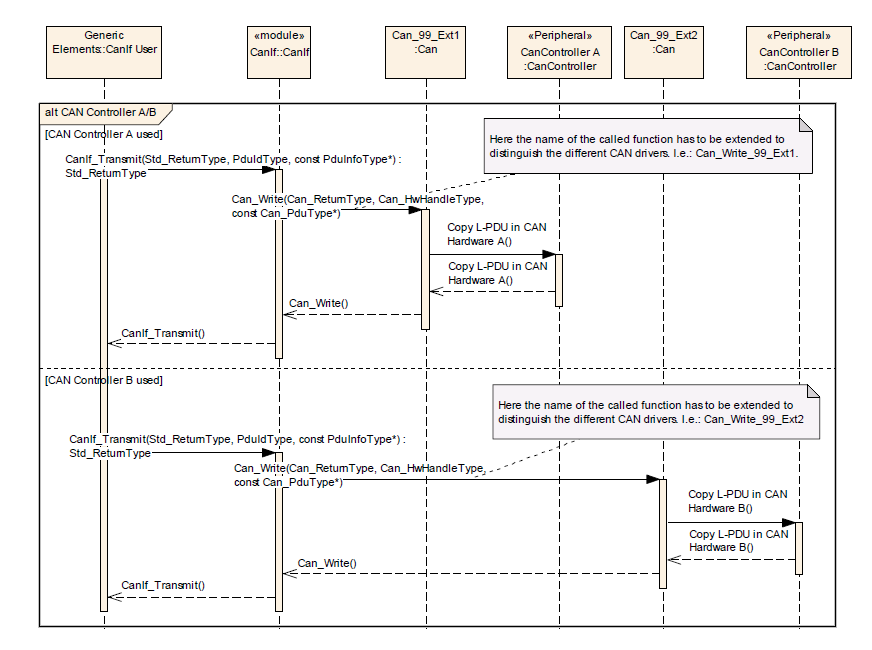
Possible applications of CanIf :

* 1. Interrupt mode
  2. Polling mode
  3. Mixed mode

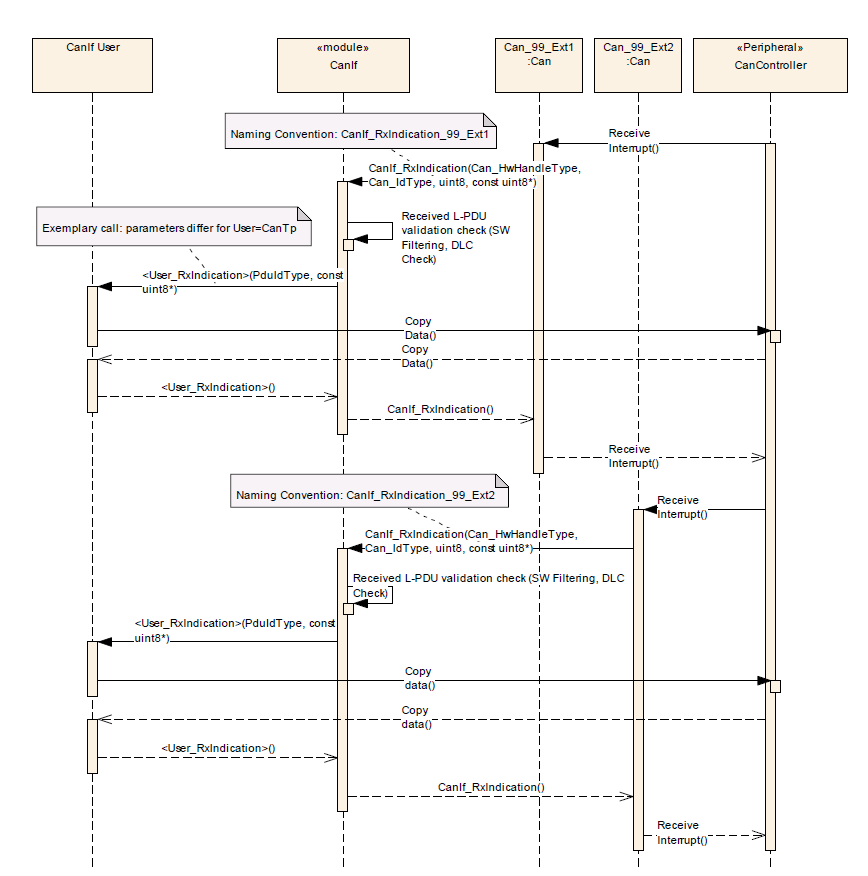
Functions provided by CanIf module :

* 1. CanIf\_Init() : for the initialization of the entire Interface (buffers).
  2. CanIf\_Transmit() : interface for the upper layer to transmit PDU on the can network.
  3. CanIf\_TxConfirmation() : callback from CanDrv to CanIf after successful transmission of data.
  4. CanIf\_RxIndication() : called when an L-PDU is received
  5. CanIf\_ReadRxPduData() : common interface for upper layer modules to read can L-PDU from can network.

**Use Case 1 : Transmission of message over CAN**

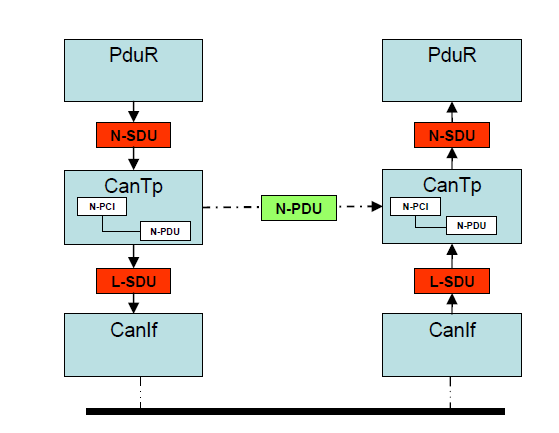
****

**Use Case 2 : Reception of message over CAN**



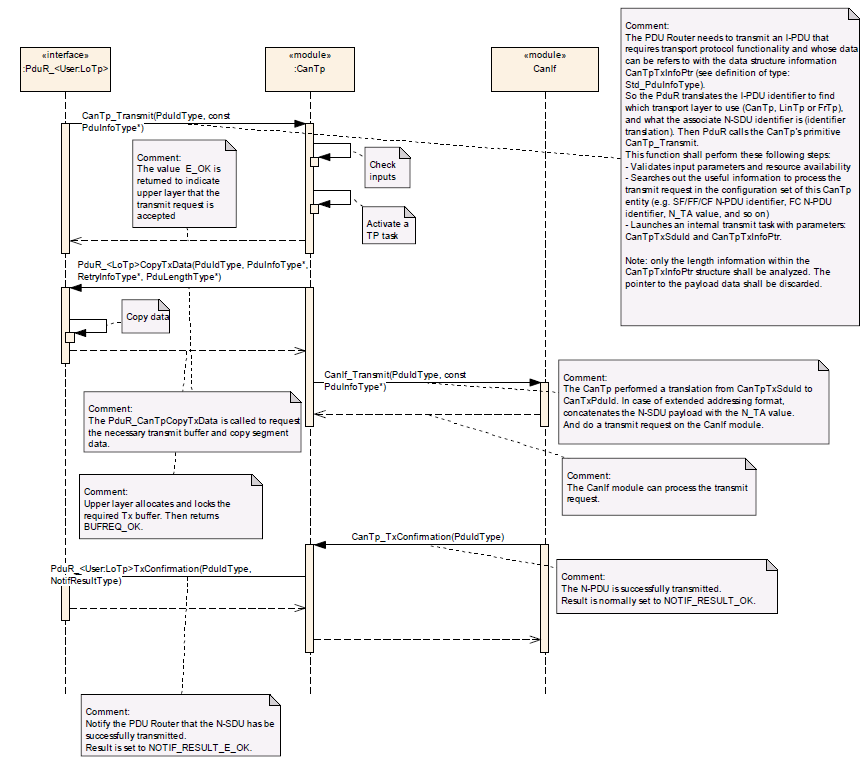
**Can Transport Layer**

* CanTp is the module present between PDU router and CanIf module
* CanTp is a part of communication stack service layer
* CanTp provides services for
  1. Segmentation of data in transmit direction
  2. Reassembling of data in receive direction
  3. Control of data flow
  4. Detection of errors in segmentation sessions
  5. Transmit cancellation
  6. Receive cancellation

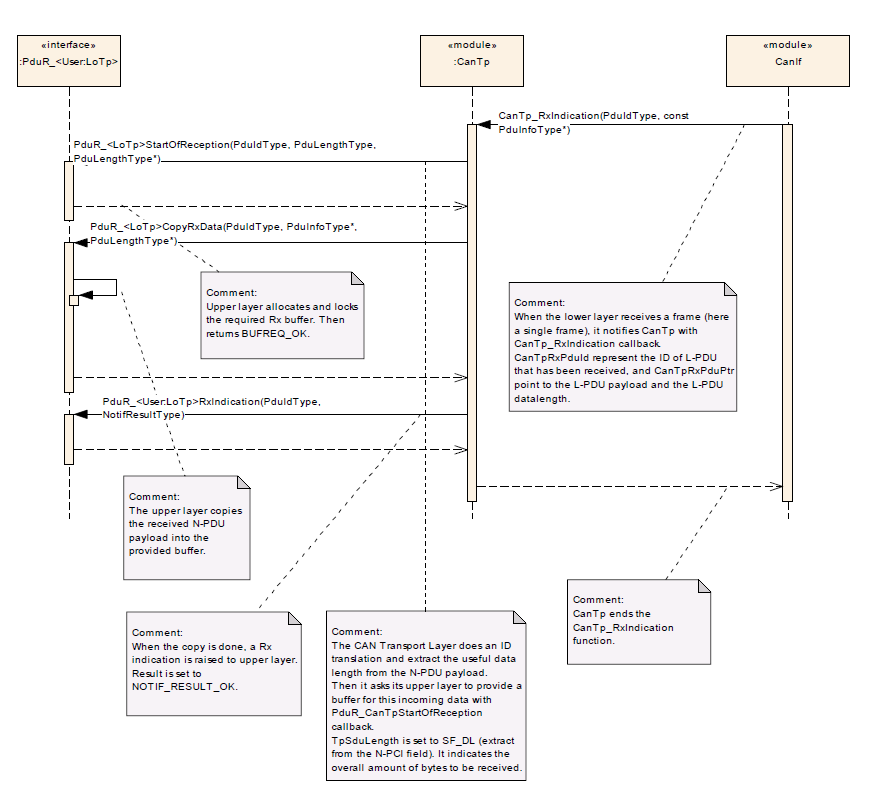
****

* CanTp features will be used by both diagnostic and AUTOSAR COM I-pdu
* CanTp upper interface provides PduR module global access to transmit and receive data
* Function provided by CanTp layer
  1. CanTp\_Init() : initialization of Tp layer buffers
  2. CanTp\_TxConfirmation() : callback function after successfully transmitting data to the bus.
  3. CanTp\_RxIndication() : when a new Can PDU frame is received, this function is called.

**CanTp Transmission**

****

**CanTp Reception**

****

**CAN Network management**

* It is a hardware independent protocol that is used only for CAN. It is a part of the services layer.
* Its main purpose is to co-ordinate the transition between normal operation and the bus-sleep mode.
* It also helps to detect all the present nodes or to detect if all other nodes are ready to sleep.
* Every network node in a cluster shall transmit periodic Network management PDUs as long as it requires bus-communication otherwise it shall not transmit any Network Management PDUs
* If bus communication a cluster is released and there are no network management PDUs on the bus for a configurable amount if time determined by

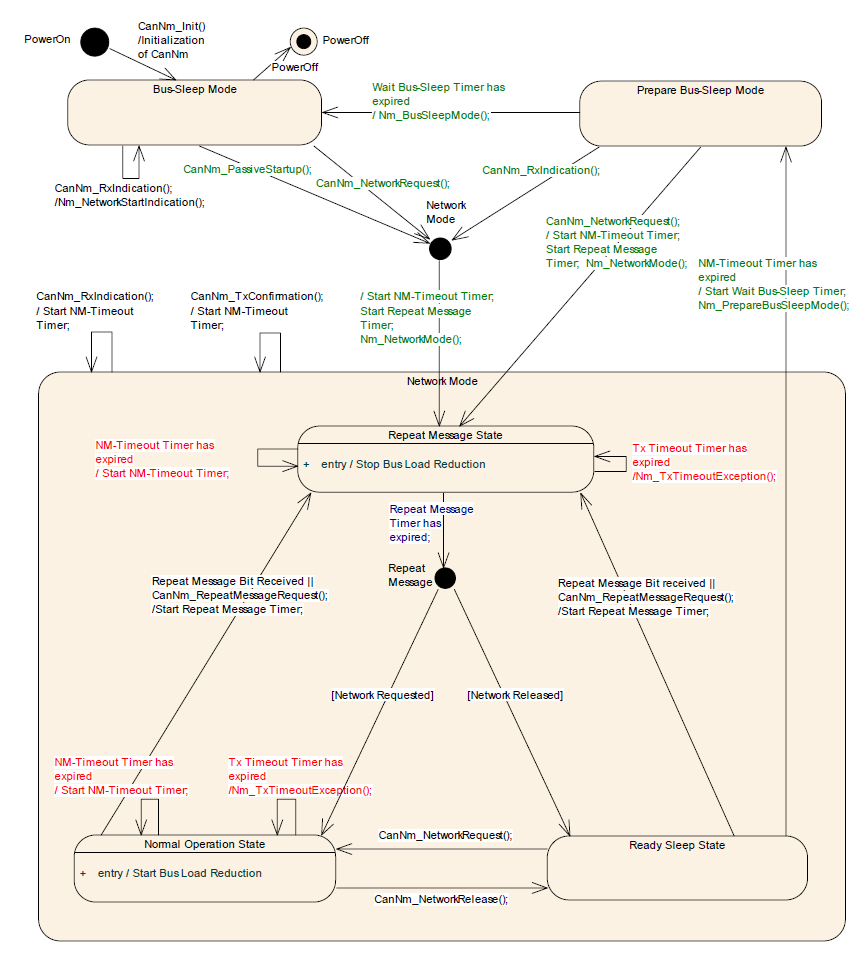
CANNM\_TIMEOUT\_TIME + CANNM\_WAIT\_BUS\_SLEEP\_TIME

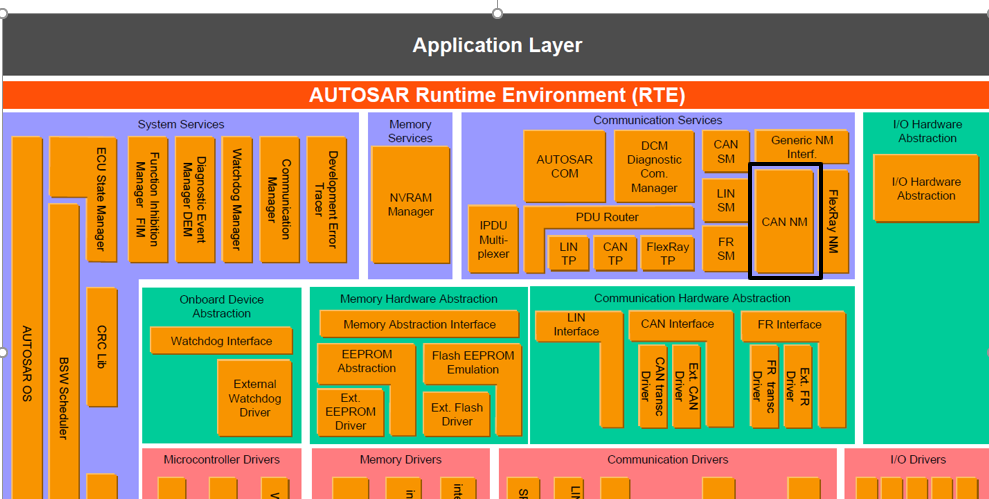
Transition into bus-sleep mode shall be performed

CAN network management shall have three operational modes along with their internal states

1. Network Mode
   * Repeat message state
   * Normal operation state
   * Ready sleep state
2. Prepare bus-sleep Mode
3. Bus-sleep mode

Please refer the UML state chart diagram for the possible state transition scenarios

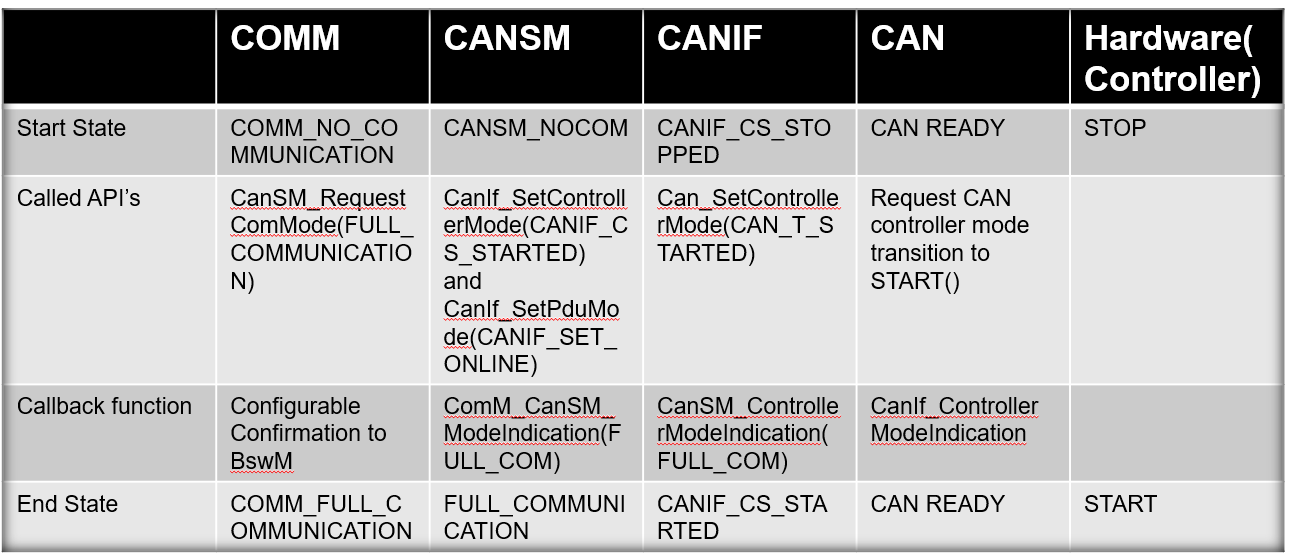
****



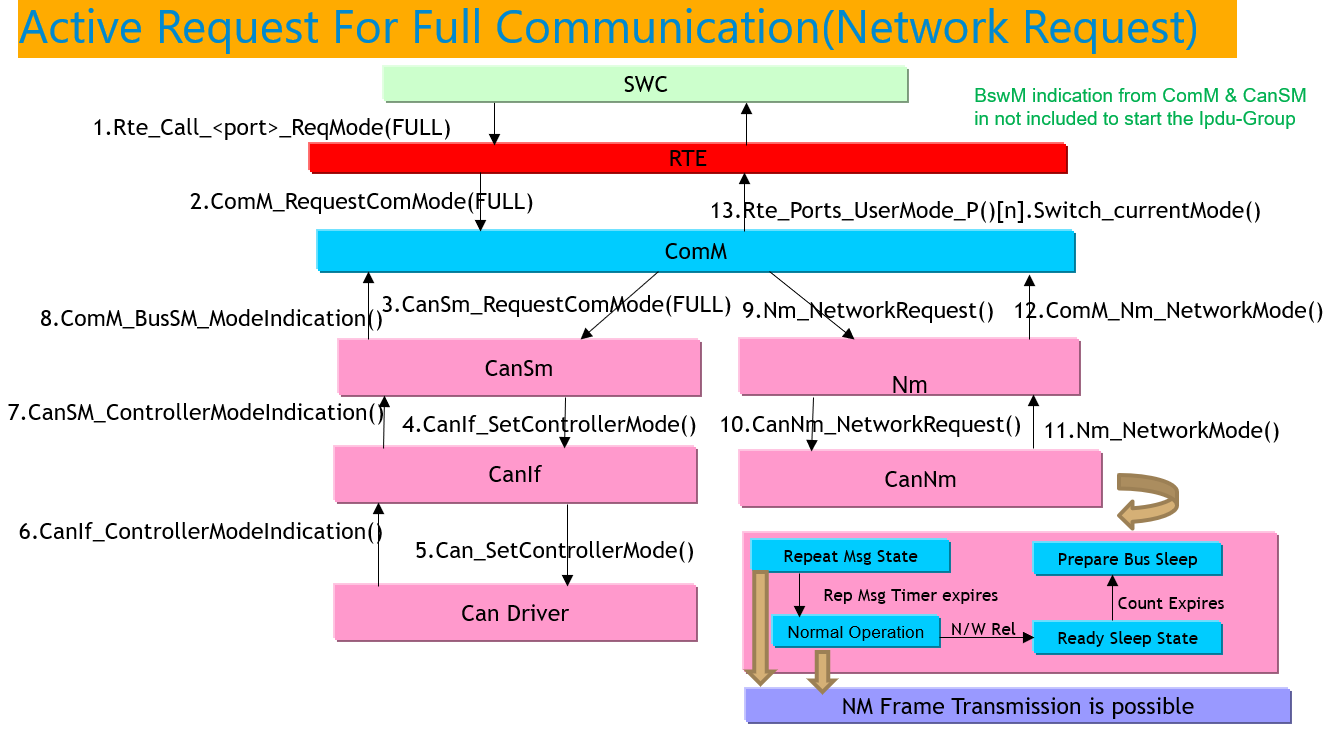
**CAN State Manager**

* The CAN state manager module is a member of the Communication service layer.
* The CanSM module is responsible for the control flow abstraction of CAN networks :
  + It changes the communication modes of the configured CAN networks depending on the mode requests from the ComM module.
* It uses the API of CanIf module.
* The CanIf module is responsible for the control flow abstraction of the configured CAN controllers and CAN transceivers.
* Any change of the CAN controller modes and CAN transceiver modes will be notified by the CanIF module to the CanSM modules.
* The possible network modes are
  + No Communication
  + Silent communication
  + Full communication

**No Communication to Full Communication**

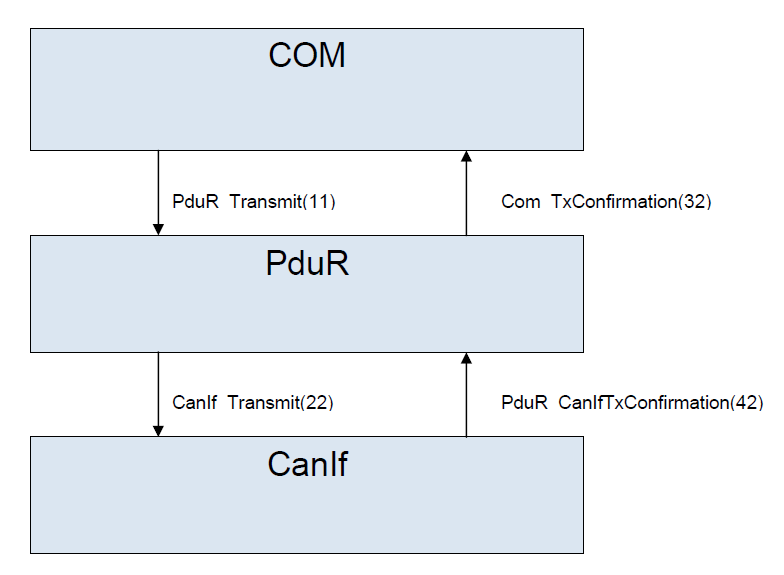


**Active Request For Full Communication(Network Request)**



**PDU Router**

* The PDU router module provides services for routing of I-PDU using
  1. Communication interface modules
  2. Transport protocol modules
* The routing of I-PDU are made statically by the I-PDU identifier, they are not routed dynamically on run-time or dependent on contents.
* PDU router is a part of Communication service layer
* PDU router module mainly consists of two parts :
  1. PDU routing tables : using static routing tables
  2. PDU routing engines : provides minimum routing capability configured at pre-compile time or link-time.[1]
* PDU router has three different functionalities
  1. PDU reception/ transmission/ gateway
  2. Single cast routing
  3. Multi-cast routing

****

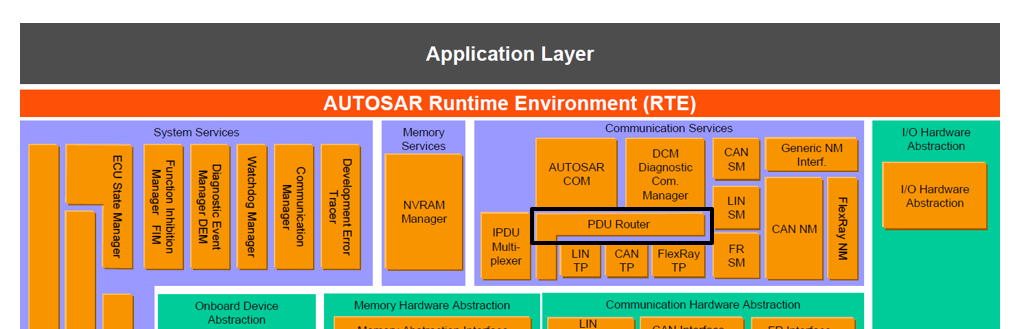
I-PDU Handling

* An I-PDU is identified by the PDU-ID
* Each BSW module that handles I-PDU and provides an API for I-PDUs must contain a list of PDU-ID.

Communication Interface

* Direct Data provision
* Trigger transmit data provision
* Data is not copied in transmit function instead data is requested later by trigger function

The beside image describes the sample interfacing between the COM module and CANIf module



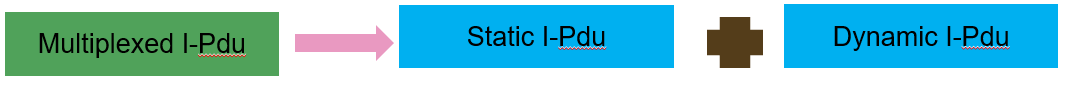
**I-PDU Multiplexer**

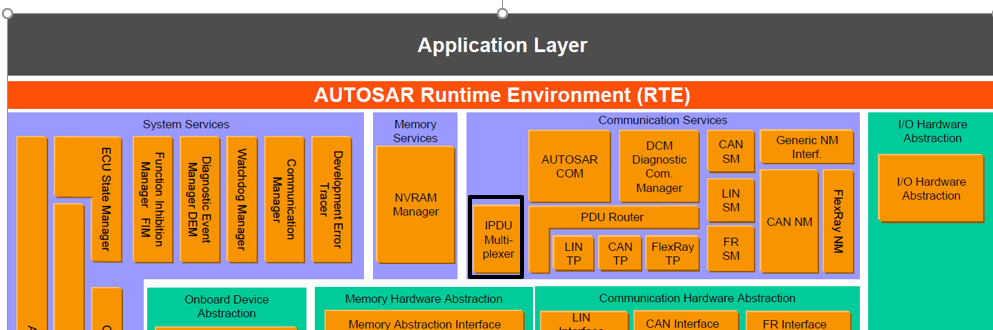
What is PDU multiplexing ?

* + PDU multiplexing means using the same PCI (Protocol control information) of a PDU (Protocol Data unit) with more than one unique layout of SDU(Service data unit).
  + A selector field is a piece of the SDU of the multiplexed PDU which is used to distinguish between the contents of the multiplexed PDUs from each other.

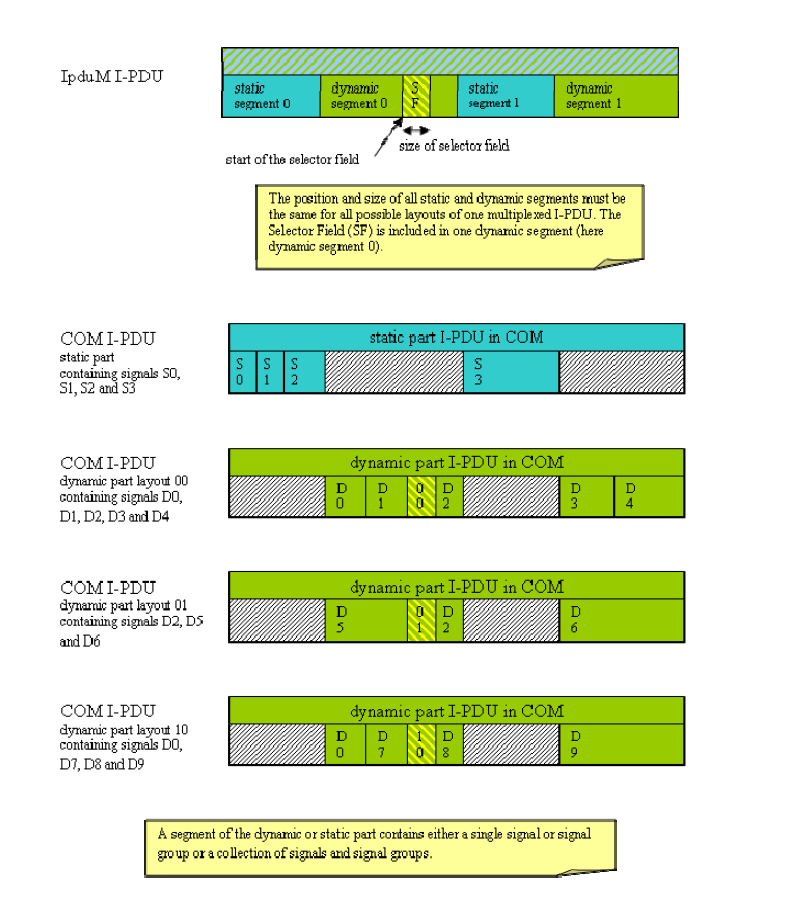
Responsibility of I-PDU multiplexer

* + On the sender side : it combines appropriate I-PDUs from COM to new, multiplexed I-PDU and send them back to the PDU-router
  + On the receiver side : it interprets the content of multiplexed I-PDU and provide COM with its appropriate separated I-PDU taking into account the value of the selector field

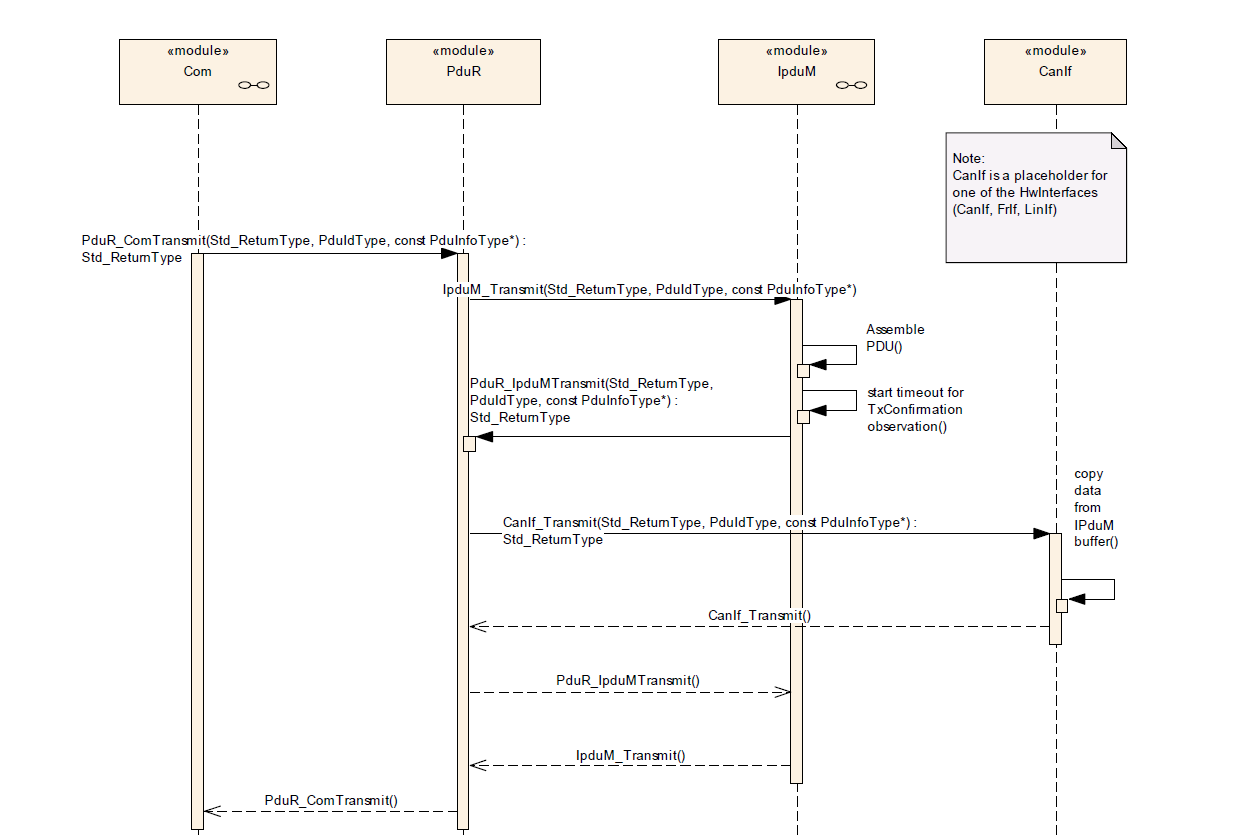




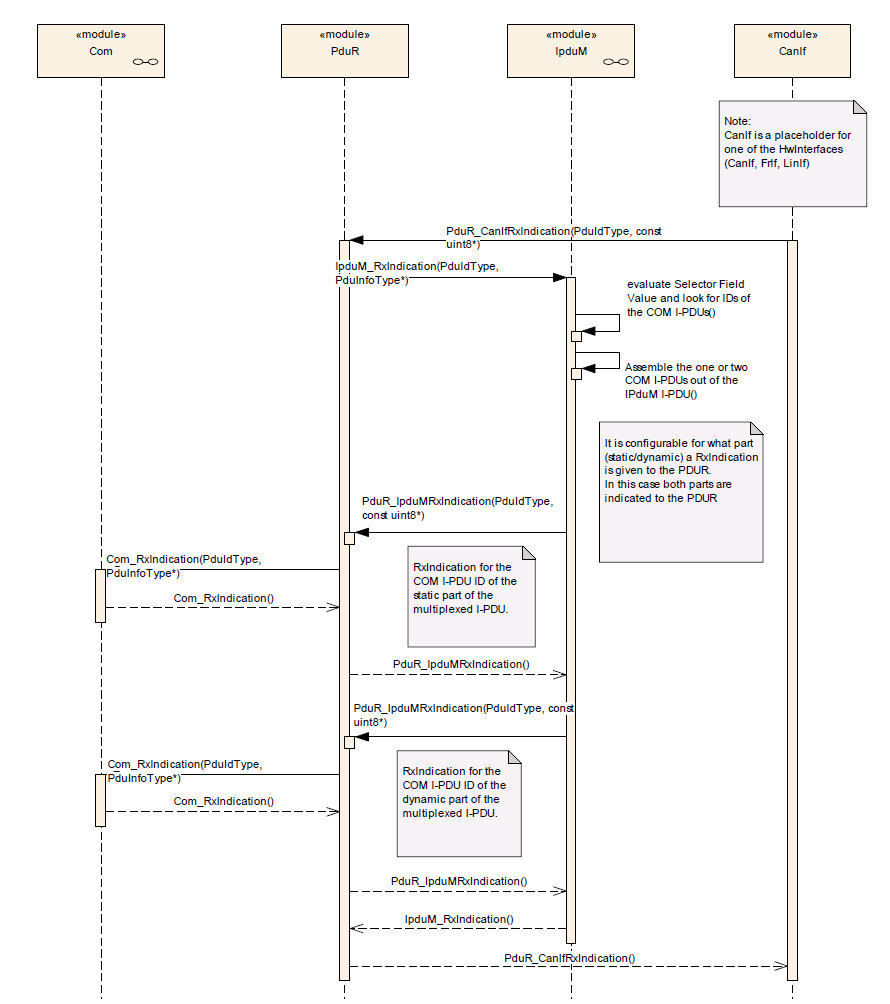
* A multiplexed I-PDU consists of a static part and a dynamic part. The static part consists of zero or more signals or signal groups. The dynamic part consists of the selector field and one or more signals or signal groups.
* The position of the static and the dynamic part of the multiplexer shall be arbitrary and has to be configurable as per I-PDU.
* There shall be only one selector filed within one multiplexed I-PDU. The value of the selector field defines how the content of the dynamic part of the I-PDU will be interpreted.



**Use Case 1: Transmit using I-PduM interface**



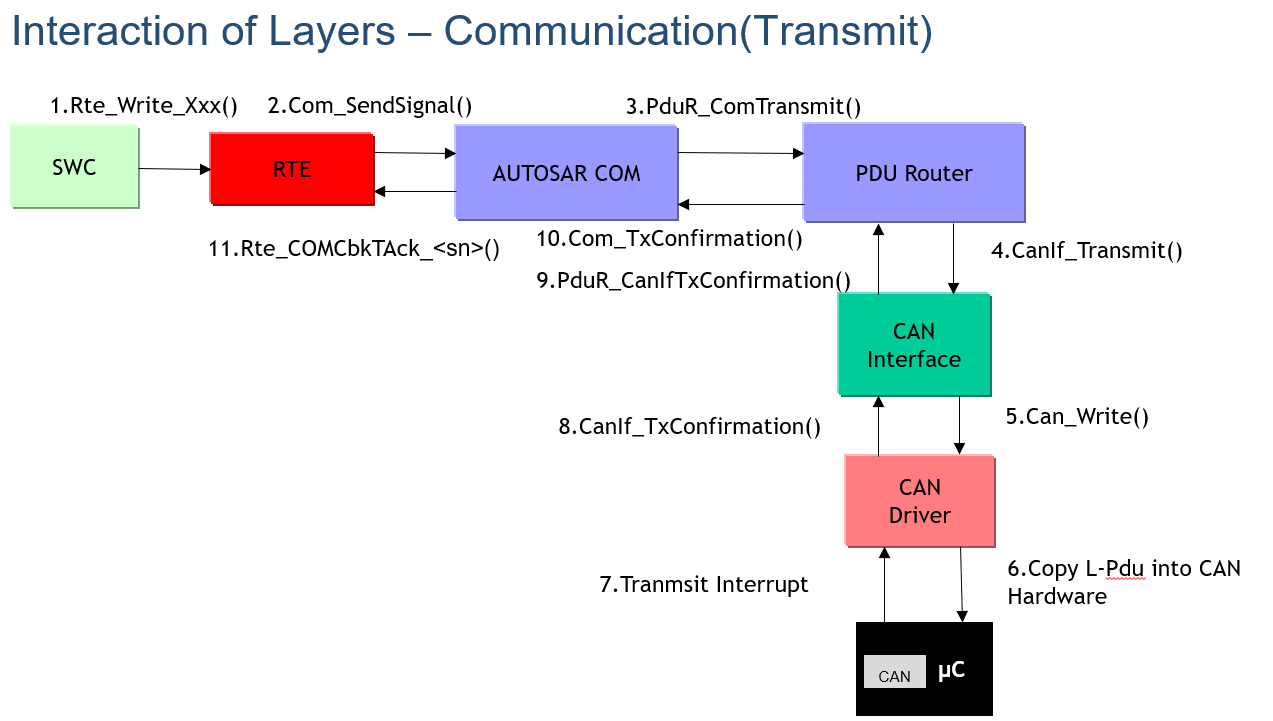
**Use Case 2: Receive using I-PduM interface**



**AUTOSAR COM Module**

* The AUTOSAR COM module is a part of the services layer
* It is placed between RTE and the PDU router.
* The main features of AUTOSAR COM module is
  + Provision of signal oriented data interface for the RTE
  + Packing of AUTOSAR signals to I-PDUs to be transmitted
  + Unpacking of received I-PDUs and provision of received signals to RTE
  + Routing of signals / signal groups from received I-PDUs into I-PDUs to become transmitted
  + Communication transmission control (start/ stop of I-PDU groups)
  + Replications of send requests
  + Guarantee of minimum distances between transmit I-PDUs
  + Monitoring of receive signals (signals timeout)
  + Filter mechanisms for incoming signals
  + Different notification mechanisms
  + Provision of init values and update indications
  + Byte order conversion
  + Sign extension
  + Support of two different transmission modes per I-PDU
  + Signal based gateway
  + Support of large and dynamic length data types
  + Support of I-PDU counters and I-PDU replication

**Interaction of Layers – Communication (Transmit)**



**Interaction of Layers – Communication (Reception)**

