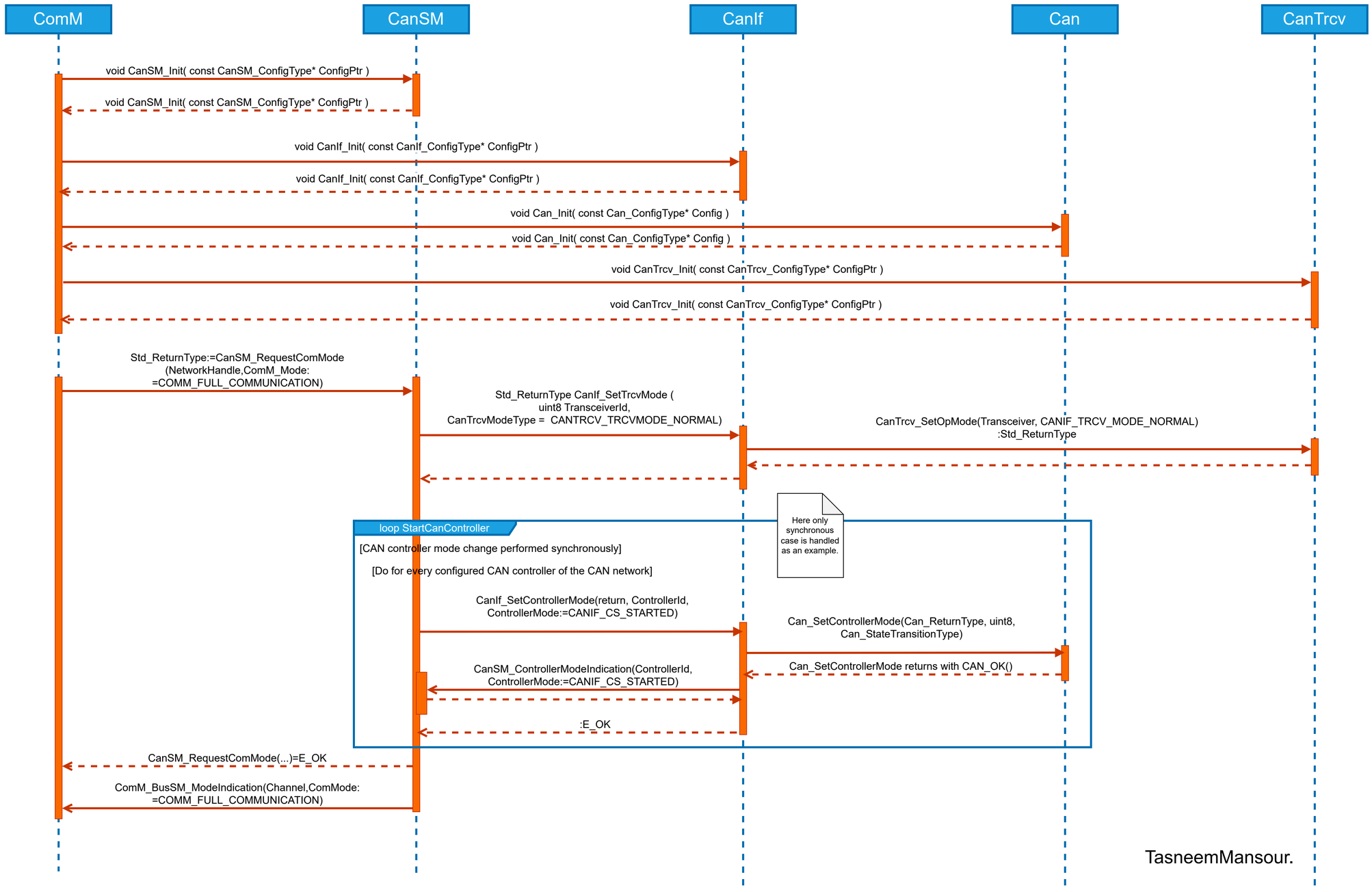
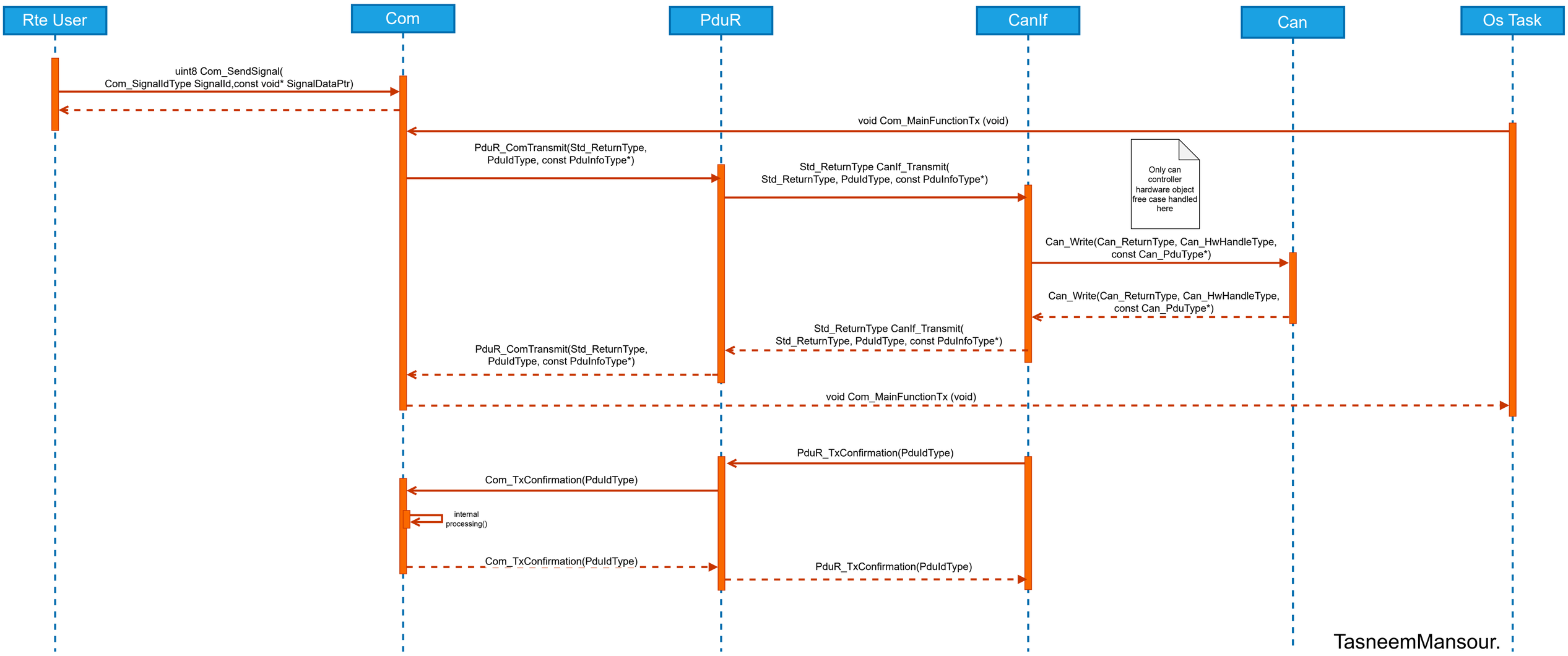
Communication Startup



TasneemMansour.

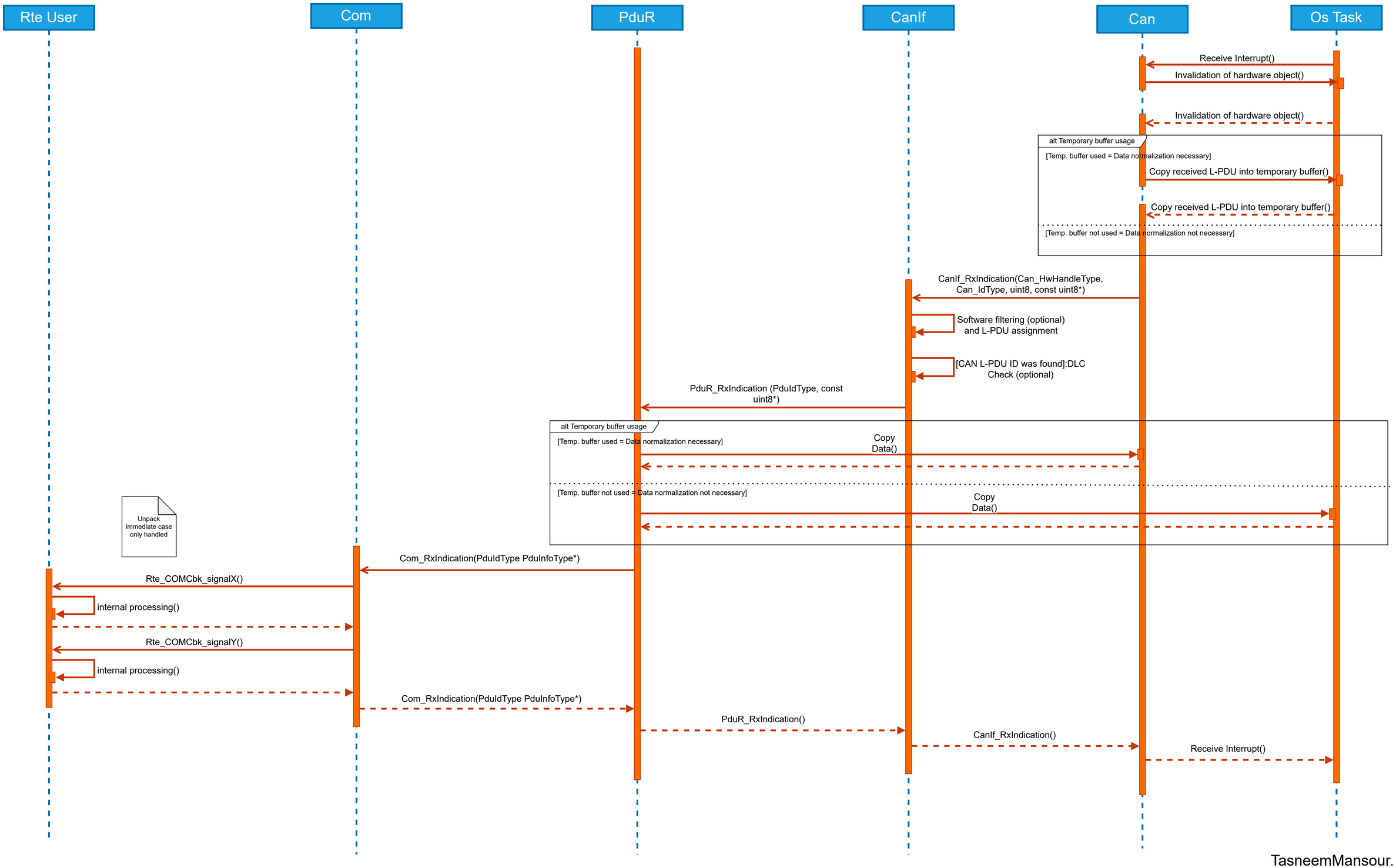
Activity	Description
Modules initialization	The ComM module starts by initialization of all other modules; CanSM, CanIf, Can and CanTrcv.
Full Communication mode requested	Full communication mode is requested starting from ComM module and this request is then passed through all other modules to start communication for all CAN drivers connected on the system.
CanTrcv Normal Mode	Setting the CanTrcv to normal mode before can controllers start.
Start CanController	All can controllers on the network are started one by one.
E_OK(FULL_COMMUNICATION MODE)	Return of corresponding communication startup request.
Indication of a successful startup sequence	The CanSM module notifies ComM module that the startup and mode request FULL_COMMUNICATION is successful.

Communication Tx Sequence



Activity	Description
OS Task starts transmission	Through Com_MainFunctionTx() the entire transmission sequence is started as PDUs are sent asynchronously on the bus. A cyclic task receives those PDUs.
Transmission request	The upper layer initiates a transmit request via the service CanIf_Transmit(). The CanTxPduId identifies the requested L-PDU. The service performs following steps: - validation of the input parameter - definition of the CAN controller to be used The second parameter *PduInfoPtr is a pointer on the structure with transmit L-PDU related data such as CanSduLength and *CanSduPtr
Start transmission	CanIf_Transmit() requests a transmission and calls the CanDrv service Can_Write() with corresponding processing of the HTH.
E_OK from Can_Write service	Can_Write() returns E_OK to CanIf_Transmit().
E_OK from CAN Interface	CanIf_Transmit() returns E_OK to the upper layer.
Tx Confirmation	Tx confirmation is sent from CanIf to all upper layers.

Communication Rx Sequence



TasneemMansour.

Activity	Description
Receive Interrupt	The CAN controller signals a successful reception and triggers a receive interrupt.
Buffering, normalizing	The L-SDU is normalized an temporary buffer located in the CAN Driver. Each d is buffered in the CAN Driver owns a temporary buffer for every physical channel only if normalizing of the data is necessary.
Indication to CAN Interface	The reception is indicated to the CAN Interface by calling of CanIf_RxIndication(). The HRH specifies the CAN RAM hardware object and the corresponding CAN controller, which contains the received L-PDU. The temporary buffer is referenced to the CAN Interface by *CanSduPtr.
Software Filtering	The Software Filtering checks, whether the received LPDU will be processed on a local ECU. If not, the received L-PDU is not indicated to upper layers.
DLC check	If the L-PDU is found the DLC of the received L-PDU is compared with the expected, statically configured one for the received L-PDU.
Receive Indication to the upper layer	The corresponding receive indication service of the upper layer is called. This signals a successful reception to the target upper layer. The parameter CanPduId specifies the L-PDU, the second parameter is the reference on the temporary buffer within the L-SDU. During is execution of this service the CAN hardware buffers must be unlocked for CPU access/locked for CAN controller access.