



TASTE case studies for PUS Services (TAPS)



Objectives:

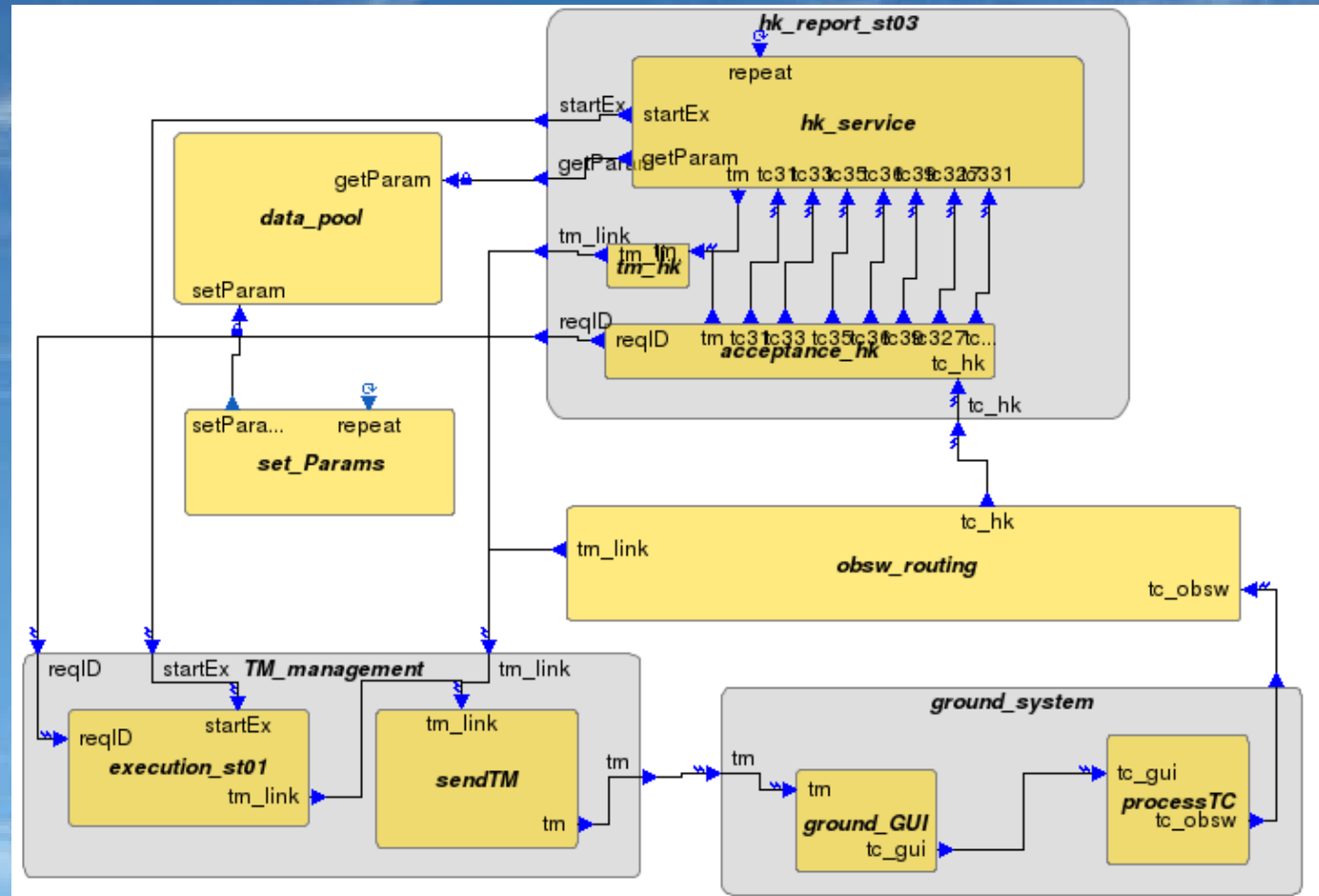
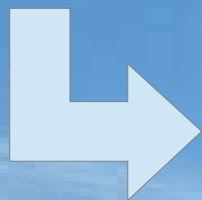
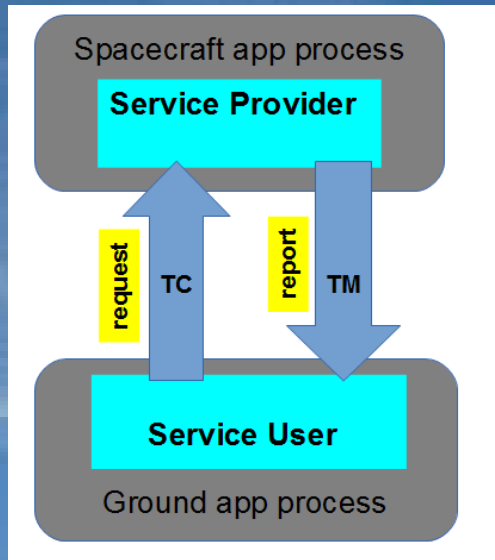
- › Validate and test the TASTE code generation and SDL editor

 - ➔ implement the PUS services

- › Generate code for PUS services

 - ➔ build a library of reusable PUS components

- Validate and test TASTE code generation and SDL editor



TAPS work flow

Development and validation of

➔ PUS services SW case studies

➔ PUS components library

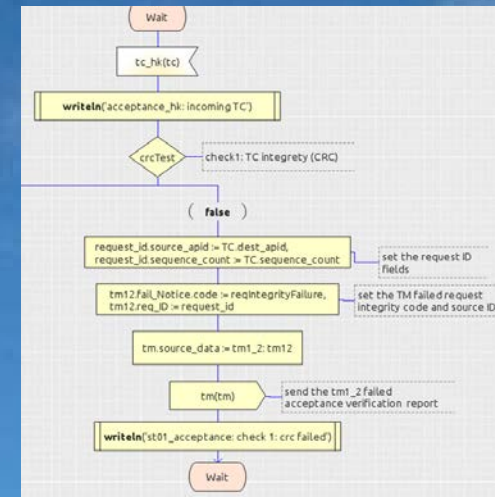
1) Define the consolidated requirements



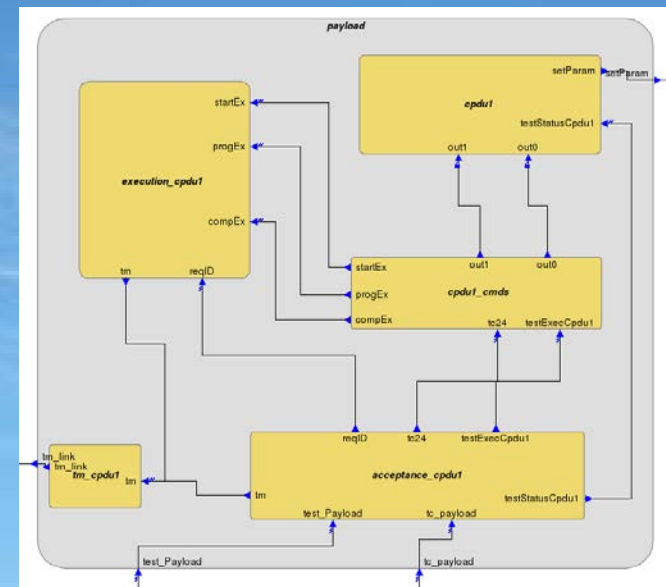
TAPS work flow

2) Build and validate the PUS services one by one

➤ Request verification ST[01]

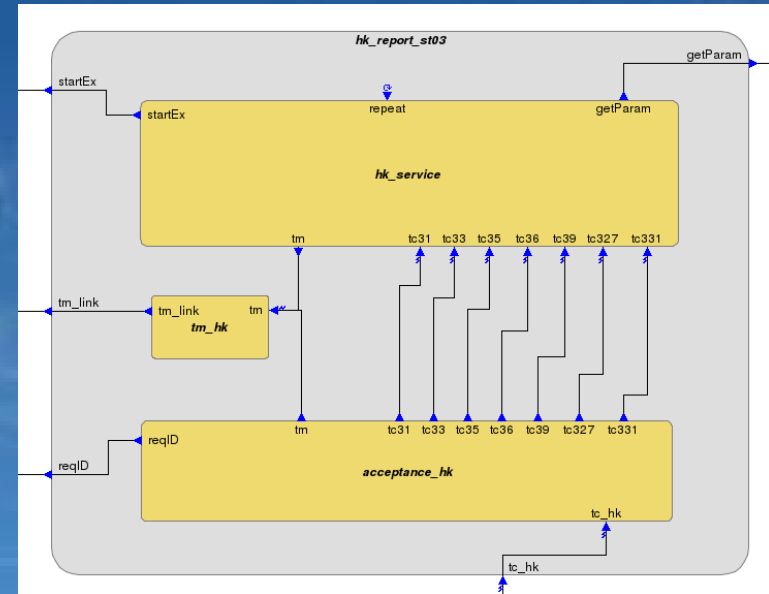


➤ Device access ST[02]

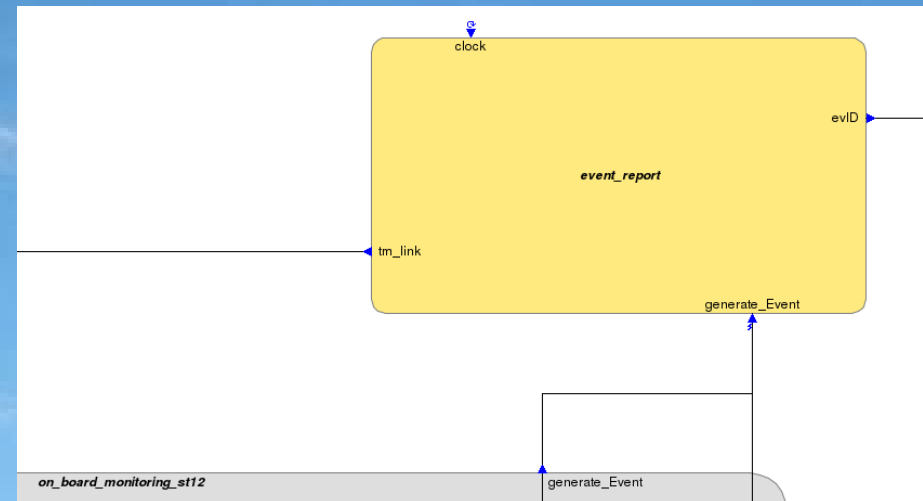


2) Build and validate the PUS services one by one

➤ Housekeeping ST[03]

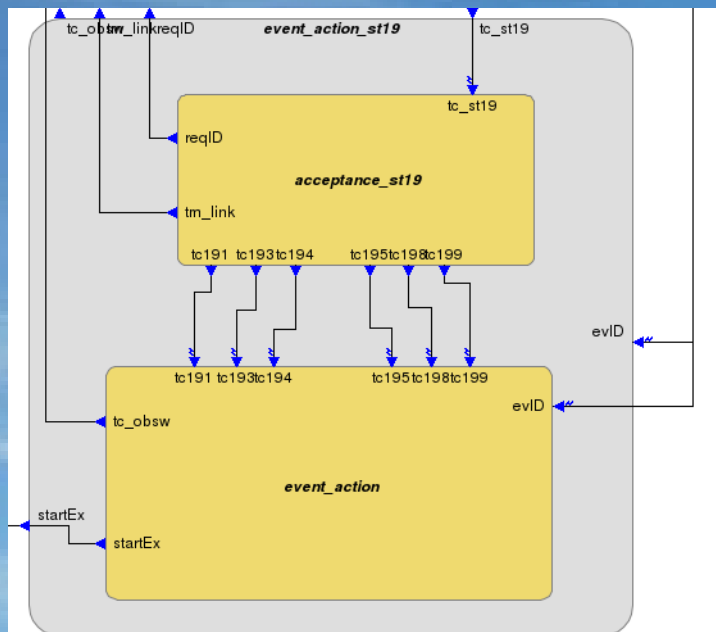
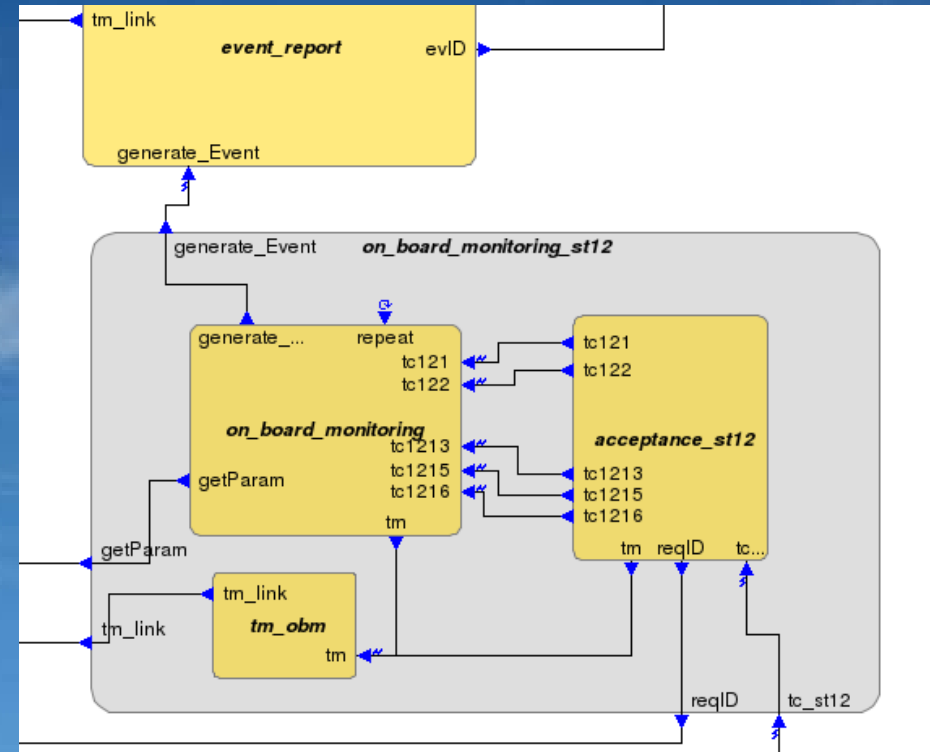


➤ Event Reporting ST[05]



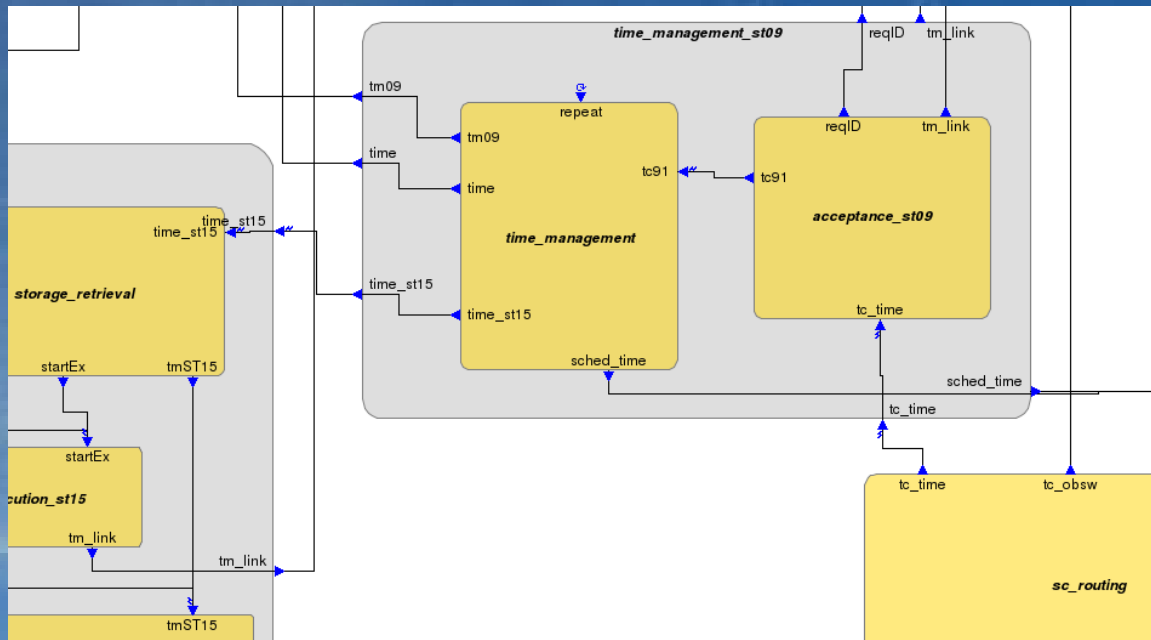
2) Build and validate the PUS services one by one

> On-board monitoring ST[12]



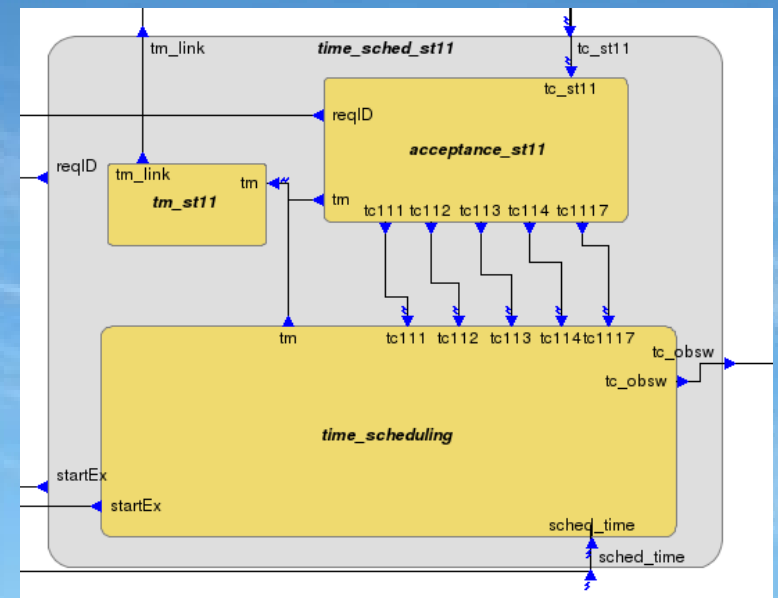
> Event-action ST[19]

2) Build and validate the PUS services one by one



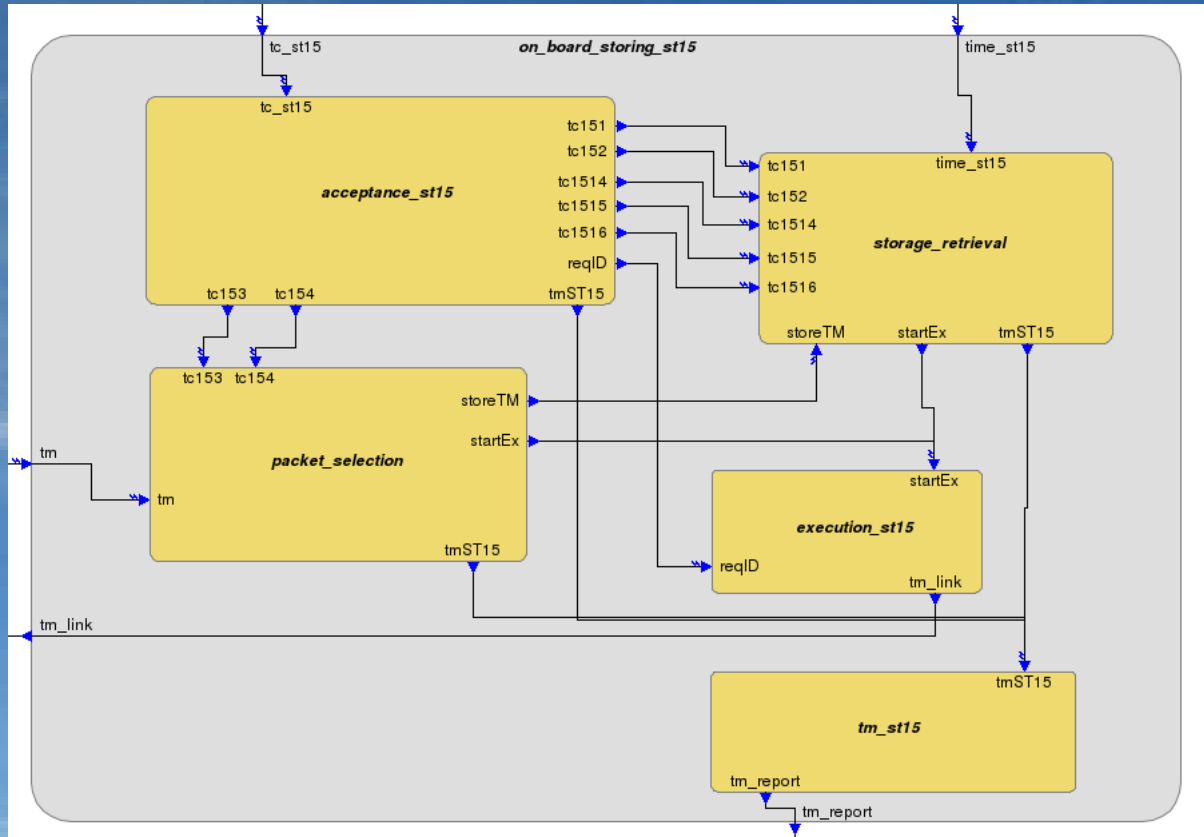
➤ Time management ST[09]

➤ Time-based scheduling ST[11]

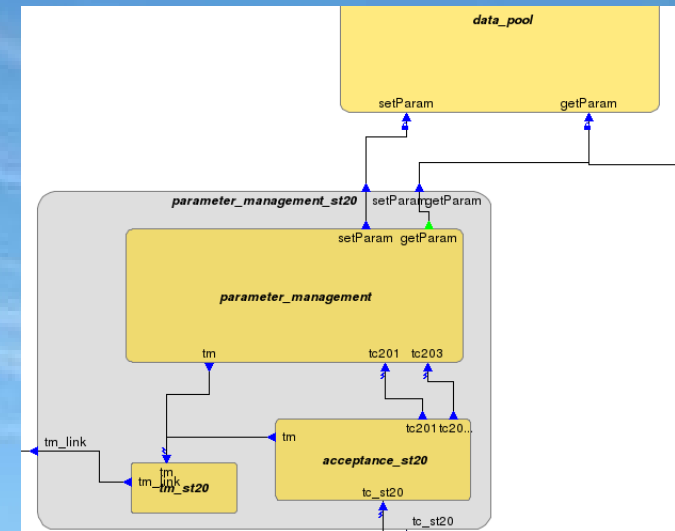


2) Build and validate the PUS services one by one

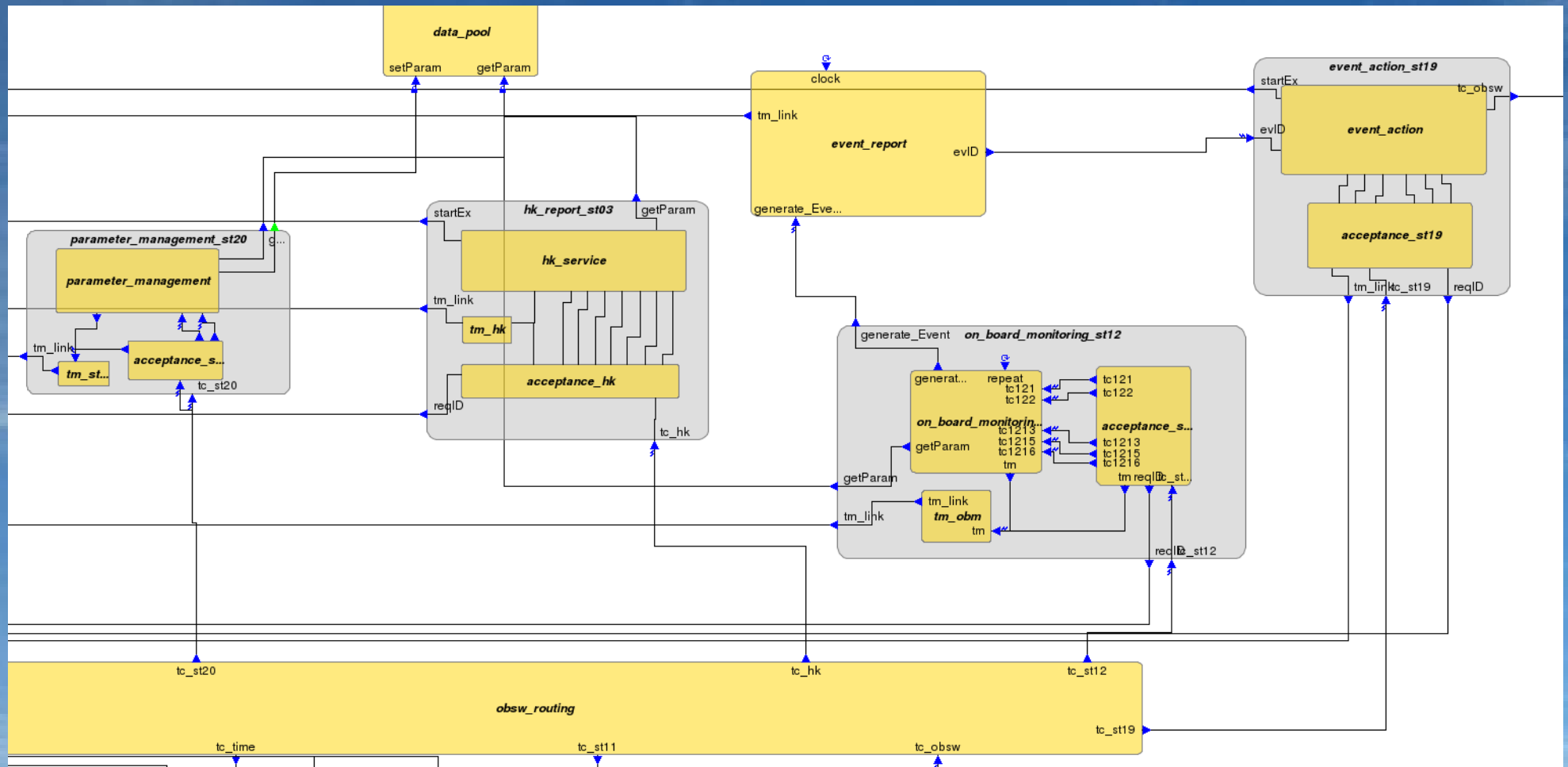
- On-board storage and retrieval ST[15]



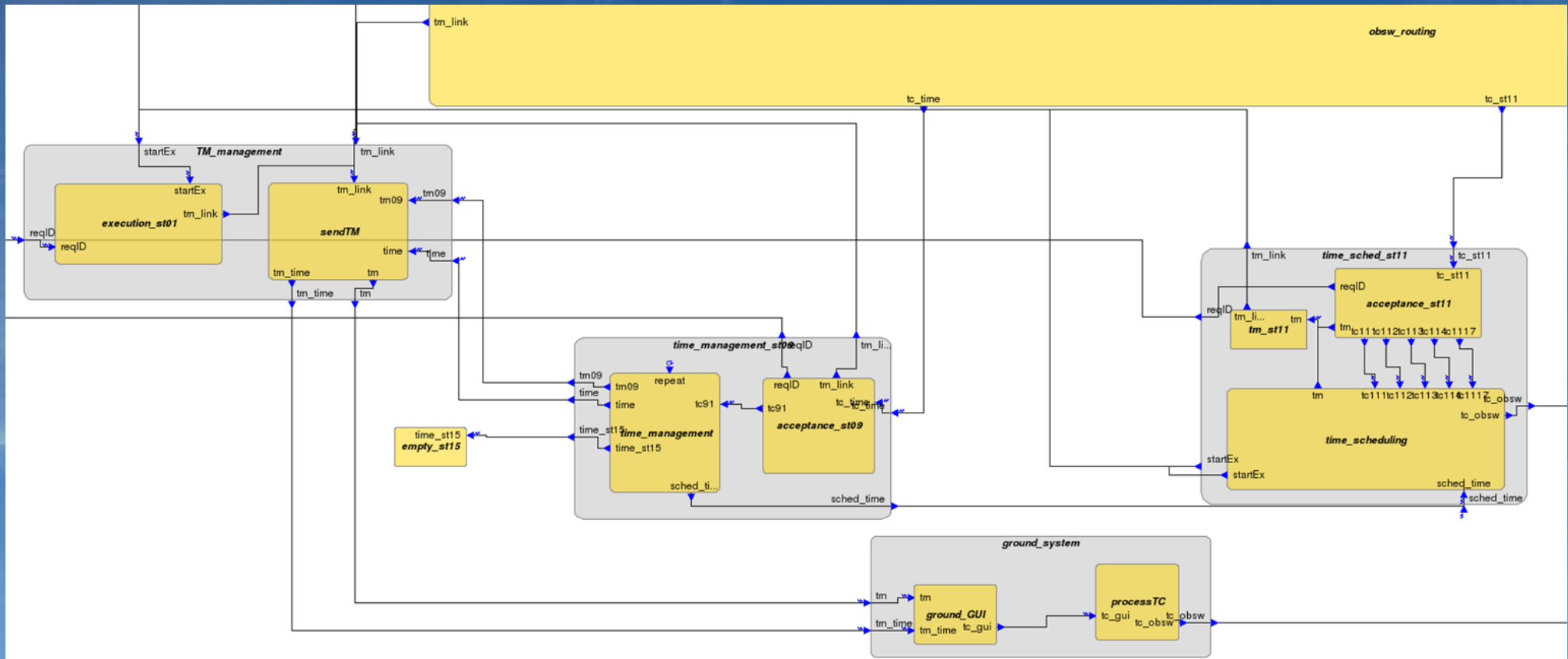
- Parameter management ST[20]



3) Make a library of PUS components



3) Make a library of PUS components



➤ PUS library

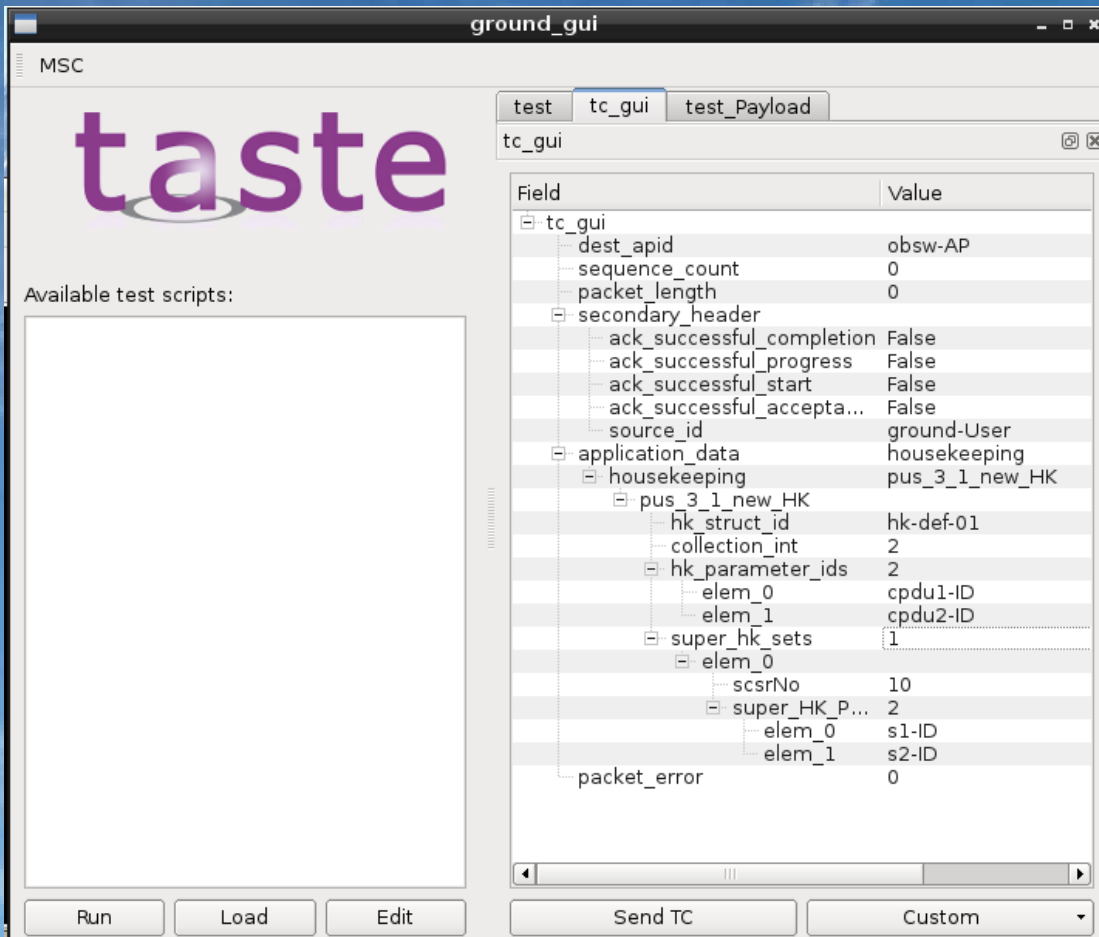
Local Function Types

- ▷ **FU** ground_system
- ▷ **FU** obsw_routing
- ▷ **FU** parameter_management_st20
- ▷ **FU** TM_management
- ▷ **FU** event_report
- ▷ **FU** on_board_monitoring_st12
- ▷ **FU** event_action_st19
- ▷ **FU** time_management_st09
- ▷ **FU** time_scheduled_st11
- ▷ **FU** empty_st15
- ▷ **FU** hk_report_st03
- ▷ **FU** data_pool

▷ **DeploymentView**

TAPS components and data flow

- › Set the request fields



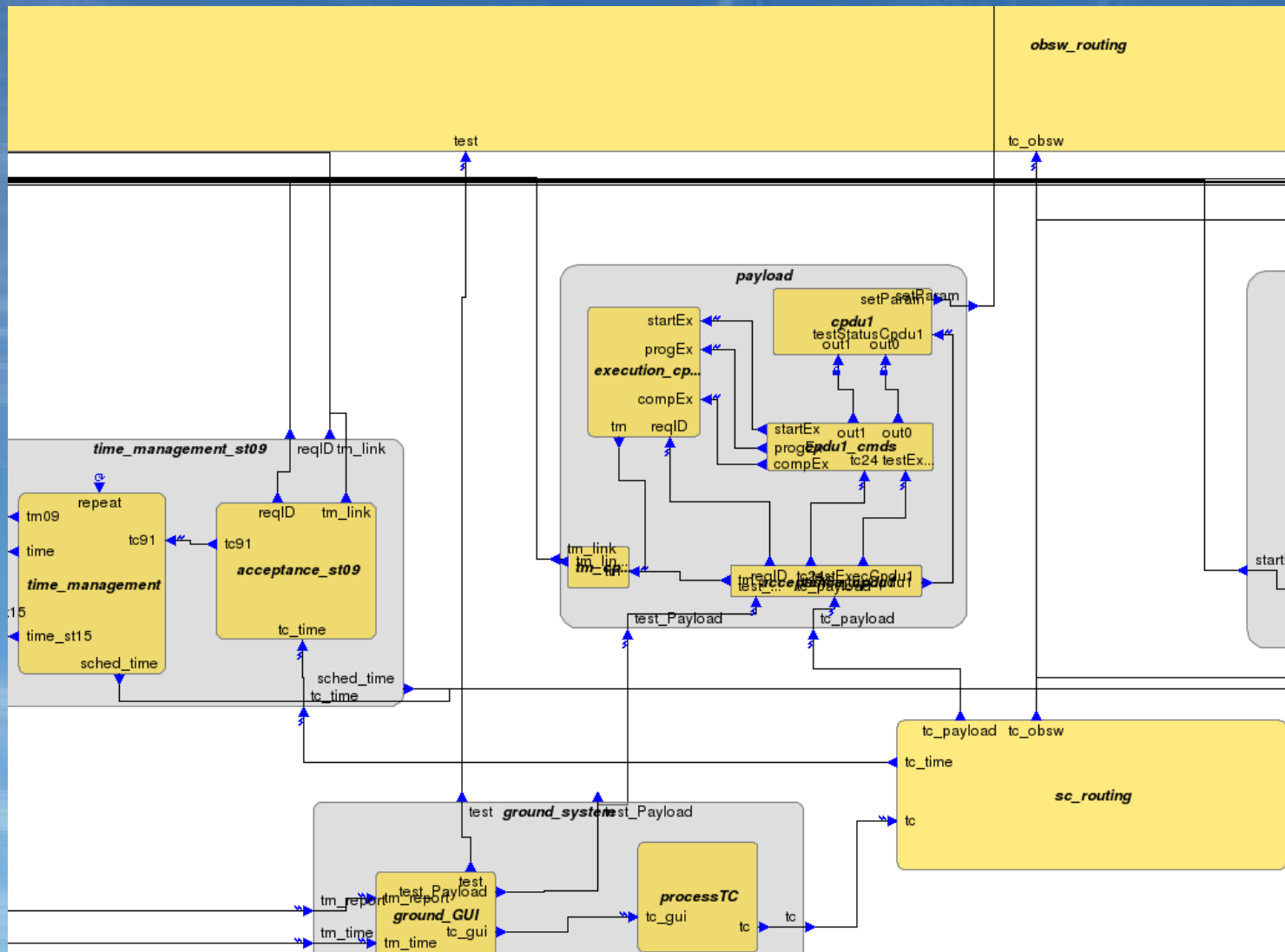
The screenshot shows the TASTE ground_gui interface. On the left, the TASTE logo is displayed above a section titled "Available test scripts:" which is currently empty. On the right, a configuration panel for the "tc_gui" test script is shown. This panel contains a table of fields and their values, organized in a tree structure. The fields include destination APID, sequence count, packet length, secondary header (with sub-fields for acknowledgment and source ID), application data, housekeeping (with sub-fields for structure ID, collection interval, parameter IDs, super HK sets, and super HK parameters), and packet error.

Field	Value
tc_gui	
dest_apid	obsw-AP
sequence_count	0
packet_length	0
secondary_header	
ack_successful_completion	False
ack_successful_progress	False
ack_successful_start	False
ack_successful_accepta...	False
source_id	ground-User
application_data	housekeeping
housekeeping	pus_3_1_new_HK
pus_3_1_new_HK	
hk_struct_id	hk-def-01
collection_int	2
hk_parameter_ids	
elem_0	cpdu1-ID
elem_1	cpdu2-ID
super_hk_sets	1
elem_0	
scsrNo	10
super_HK_P...	2
elem_0	s1-ID
elem_1	s2-ID
packet_error	0

At the bottom of the interface, there are buttons for "Run", "Load", "Edit", "Send TC", and a "Custom" dropdown menu.

TAPS components and data flow

- Route the request



TAPS components and data flow

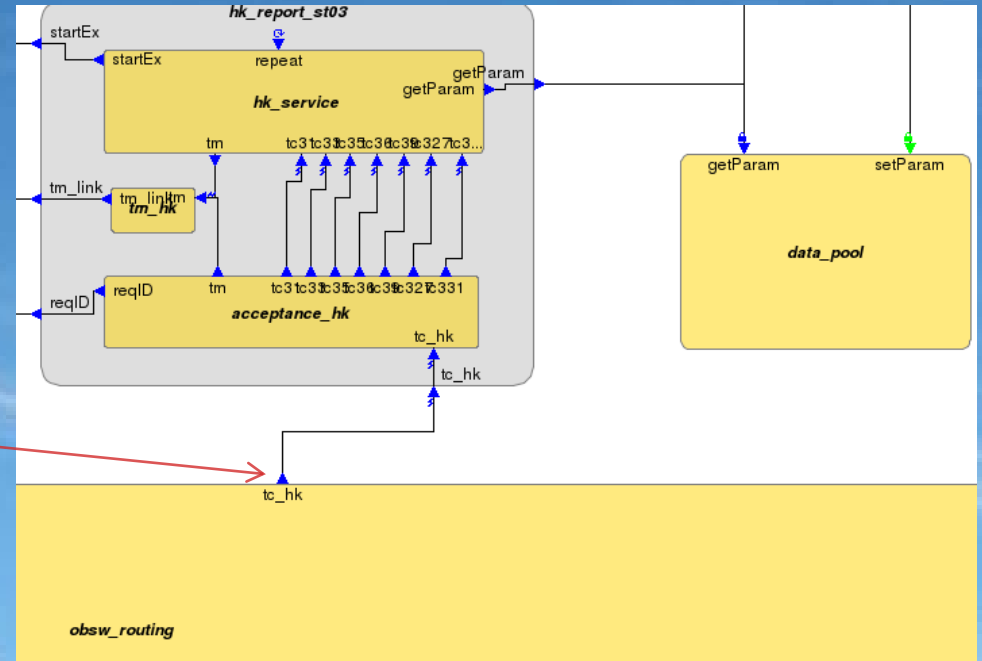
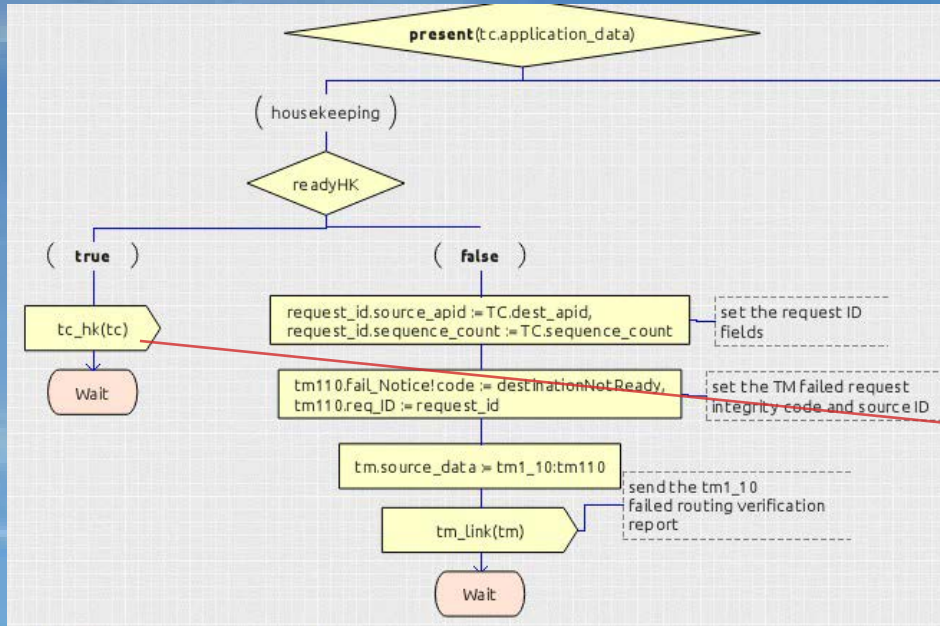
Route the request

```

-- choose between one of the PUS services
PUS ::= CHOICE {
  device-access           PUS-2,
  housekeeping            PUS-3,
  memory-management      PUS-6,
  time-management        PUS-9,
  time-scheduling         PUS-11,
  on-board-monitoring    PUS-12,
  storage-and-retrieval  PUS-15,
  event-action           PUS-19,
  param-management       PUS-20
}
-- Instantiate the generic TC-type
TC ::= TC-type(APID, Seq-count, APUserID, PUS)
    
```

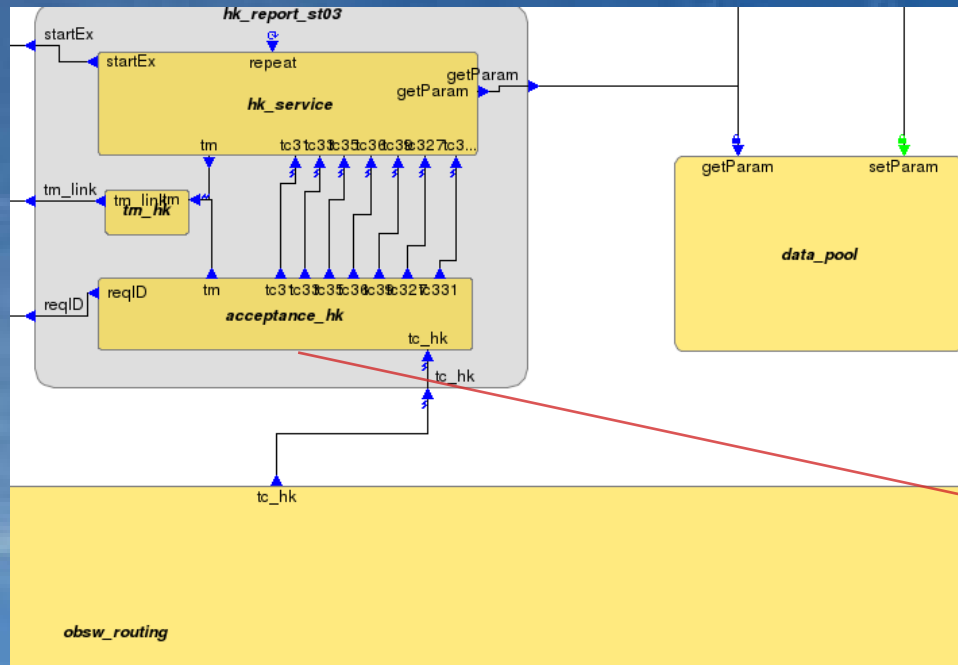
```

PUS-3 ::= CHOICE {
  pus-3-1-new-HK           Create-HK-Struct,
  pus-3-3-del-HK           Manage-HK-Struct,
  pus-3-5-periodic-HK-on  Manage-HK-Struct,
  pus-3-6-periodic-HK-off Manage-HK-Struct,
  pus-3-9-report-HK       Manage-HK-Struct,
  pus-3-27-one-shot-HK    Manage-HK-Struct,
  pus-3-31-new-col-int    Change-CollectionInterval
}
    
```

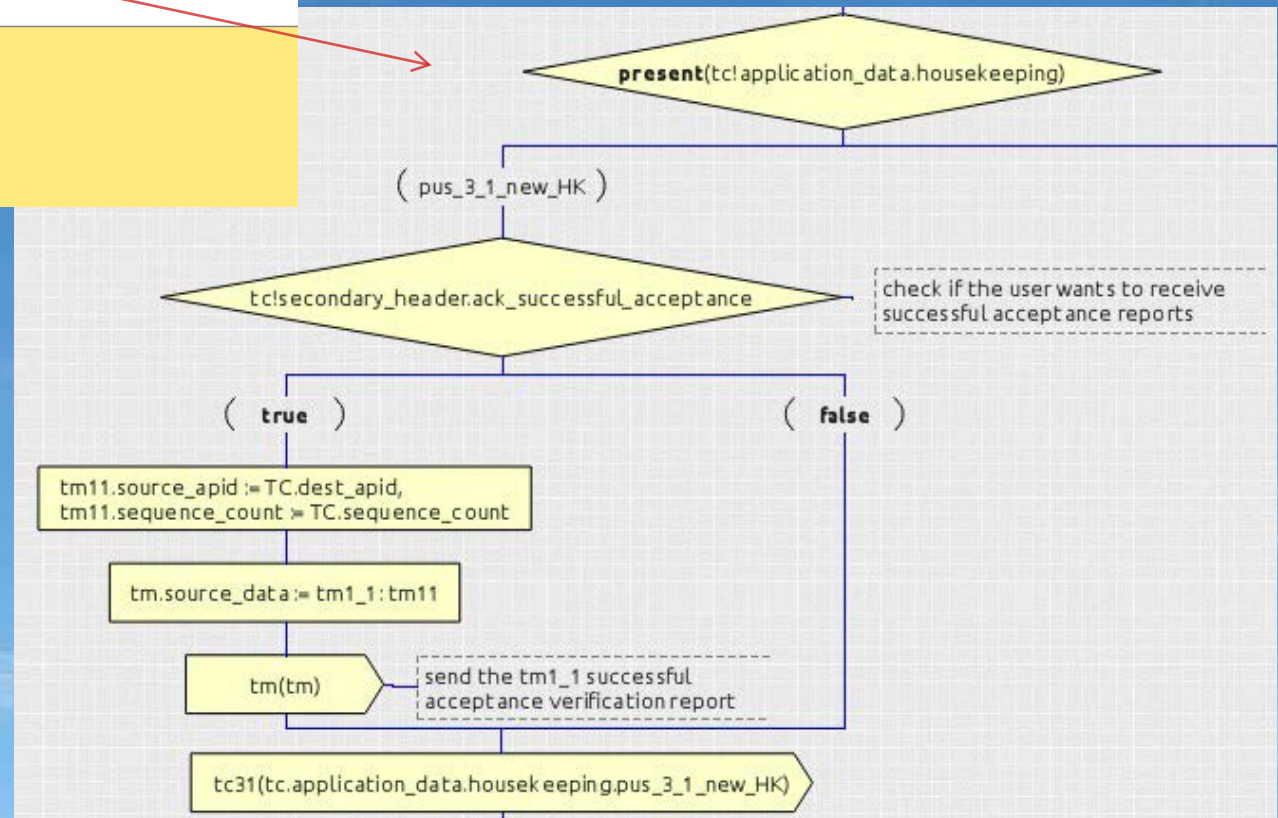


TAPS components and data flow

> Route the request

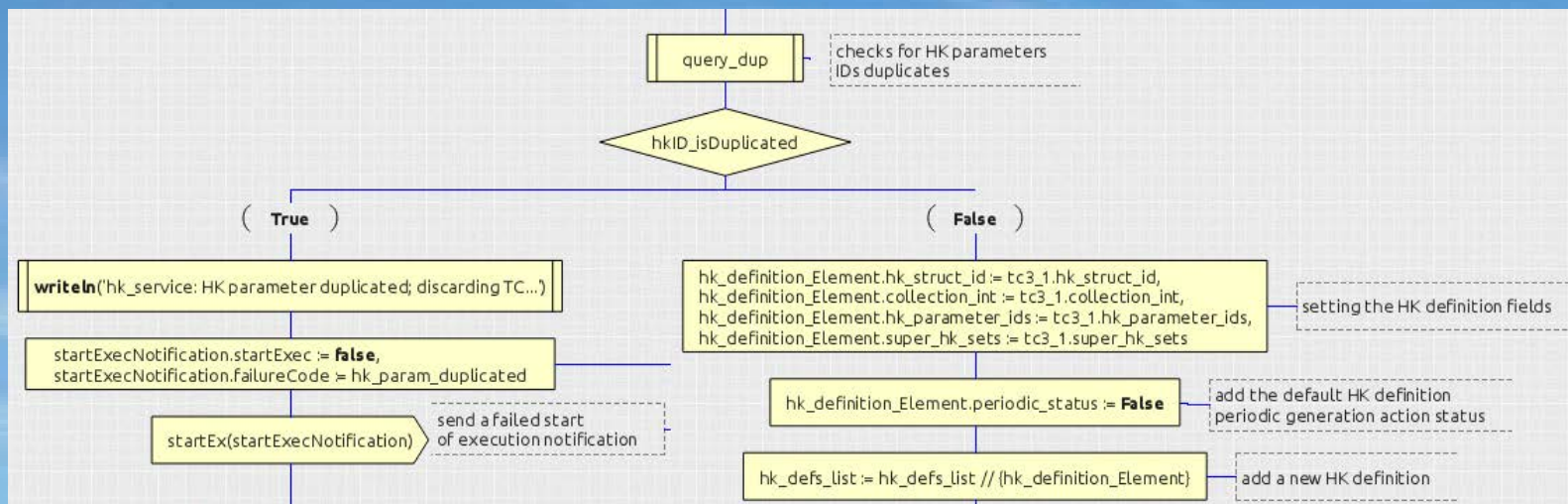
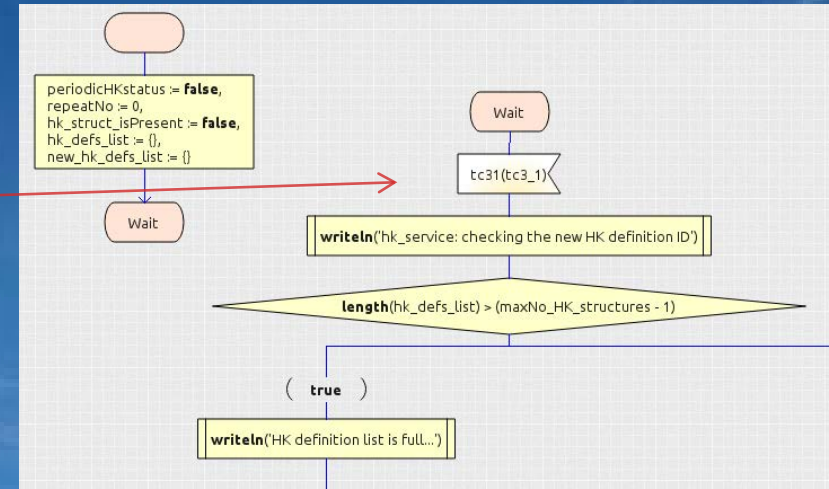
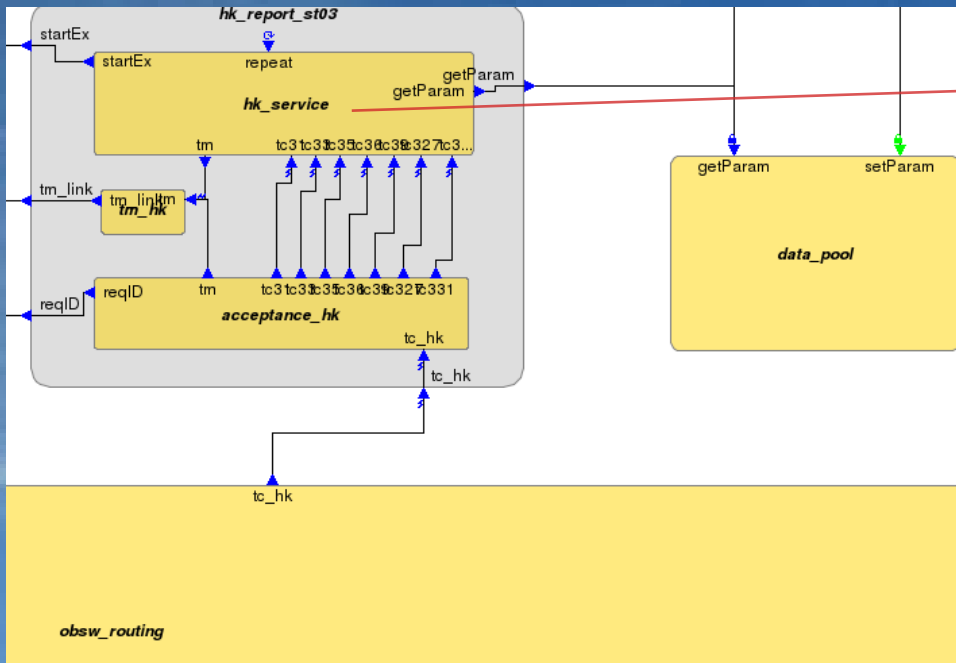


```
PUS-3 ::= CHOICE {
  pus-3-1-new-HK          Create-HK-Struct,
  pus-3-3-del-HK         Manage-HK-Struct,
  pus-3-5-periodic-HK-on Manage-HK-Struct,
  pus-3-6-periodic-HK-off Manage-HK-Struct,
  pus-3-9-report-HK      Manage-HK-Struct,
  pus-3-27-one-shot-HK   Manage-HK-Struct,
  pus-3-31-new-col-int   Change-CollectionInterval
}
```



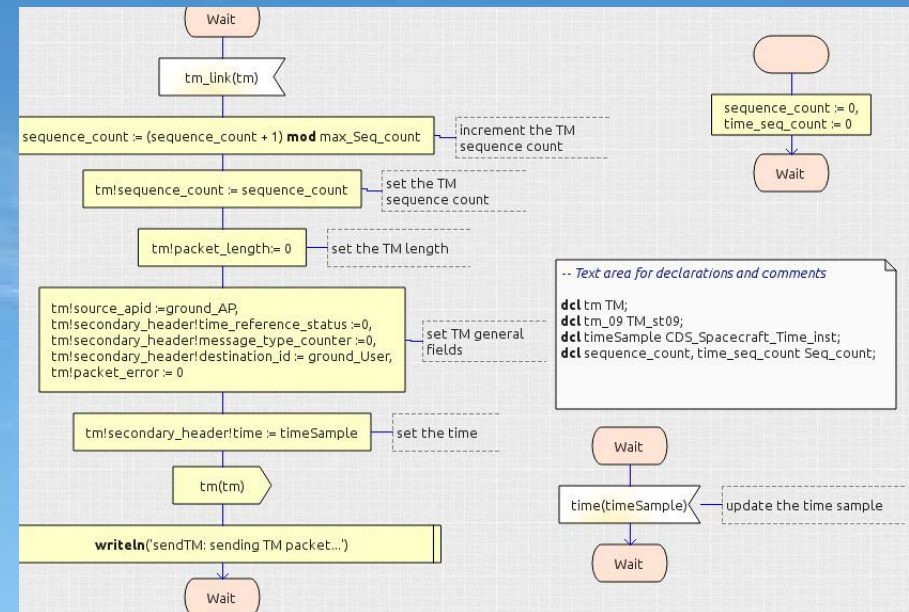
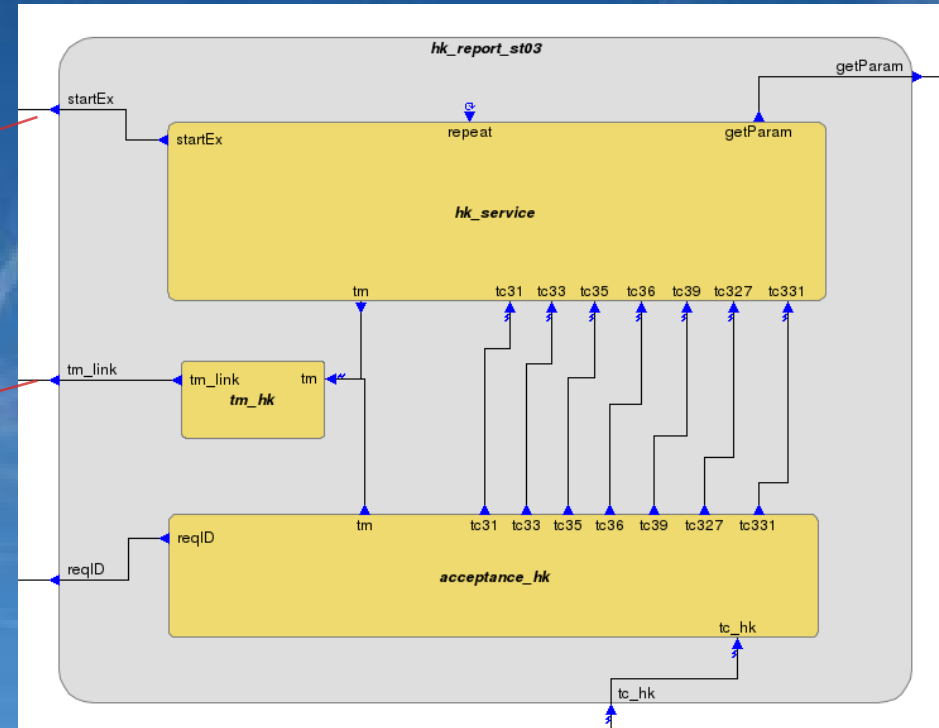
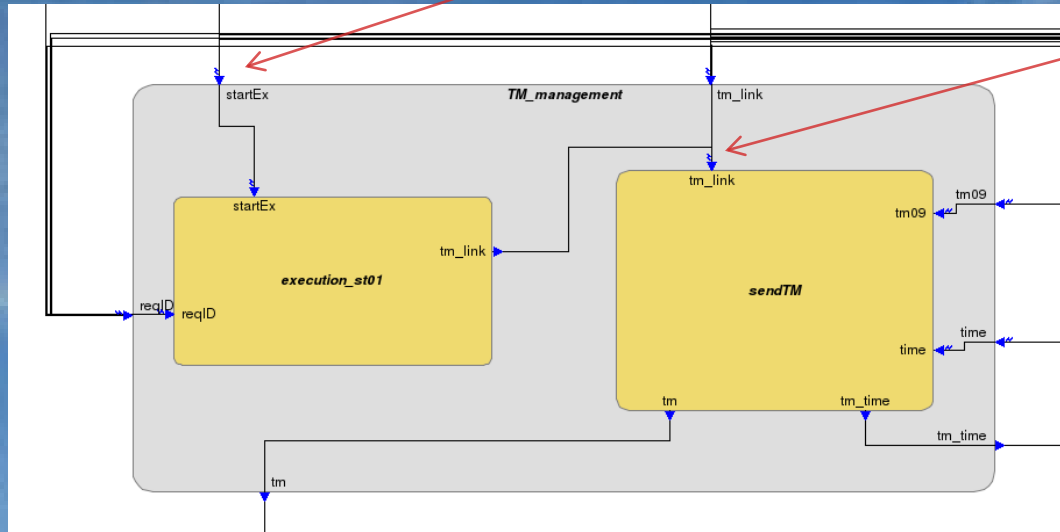
TAPS components and data flow

Process the HK request

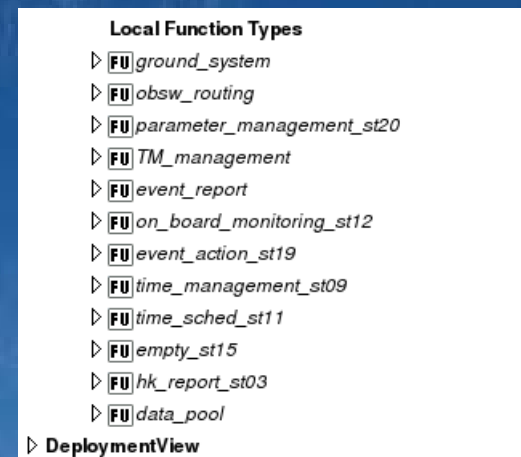


TAPS components and data flow

> sending TM reports



Building a PUS library



> Refactor and document the PUS functions

3.3 How to instantiate the HK service parameters:

- Set the values for the following HK service parameters according to your mission requirements:

max-CollectionInterval:	the maximum value for the collection interval parameter, units of MSI - minimum sampling interval
n1-simply-HK:	the maximum number of simply commutated HK parameters
n2-super-HK:	the maximum number of super commutated HK parameters
maxNo-HK-structures:	the maximum no. of HK parameter report structures
HK-structure-ID-inst:	the HK parameter report structure IDs
Simply-HK-Parameters-IDs-inst:	the HK simply commutated HK parameters IDs enumeration
Super-HK-Parameters-IDs-inst:	the HK super commutated HK parameters IDs enumeration
no-Of-ParamValues	$n1\text{-simply-HK} + (\text{max-CollectionInterval} * n2\text{-super-HK})$
Parameter-Value-inst	the HK parameter value CHOICE structure

3.5 How to use the Housekeeping service:

- select the 'housekeeping' application data for the TC
- create a HK parameter report structure: pus-3-1-new-HK
 - select the HK structure identifier
 - select the collection interval: collection_int
 - select a list of simply commutated HK parameters IDs
 - select a list of sets of super commutated HK parameters Ids
- delete a HK parameter report structure: pus-3-3-del-HK
 - select the HK structure identifier
- report HK parameter report structures: pus-3-9-report-HK
- generate a one shot report for HK parameter report structures: pus-3-27-one-shot-HK
- modify the collection interval: pus-3-31-new-col-int
- enable the periodic generation of HK parameter reports: pus-3-5-periodic-HK-on
- disable the periodic generation of HK parameter reports: pus-3-6-periodic-HK-off



TASTE case studies for PUS Services (TAPS)



Conclusions:

➤ PUS case studies implemented in TAPS:

Request verification ST[01]

Device access ST[02]

Housekeeping ST[03]

Event Reporting ST[05]

Memory management ST[06]

Time management ST[09]

Time-based scheduling ST[11]

On-board monitoring ST[12]

On-board storage and retrieval ST[15]

Event-action ST[19]

Parameter management ST[20]





TASTE case studies for PUS Services (TAPS)



Conclusions:

- › Validate the TASTE code generation & the SDL editor
- › Make a library of reusable PUS components:

Housekeeping ST[03]
Event Reporting ST[05]
Time management ST[09]
Time-based scheduling ST[11]
On-board monitoring ST[12]
Event-action ST[19]
Parameter management ST[20]