Herschel - SPIRE

On Board Software

Software Specification Document

Document No: SPIRE-IFS-DOC-PRJ-001036

Issue: Draft 0.1

Prepared by:	Anna Maria Di Giorgio Claudio Codella Stefano Pezzuto	Date:	18/05/2001
Checked by:	Riccardo Cerulli-Irelli	Date:	

Distribution list:

1 Introduction

1.1 Purpose of the document

TBW

1.2 Definitions, Acronyms and Abbreviations

TBW

1.3 References

- Tom DeMarco, *Structured Analysis and Systems Specification*. Englewood Cliffs, N.J.: Prentice-Hall, 1979.
- Derek J. Hatley and Imtiaz A. Pirbhai, *Strategies for Real-Time System Specification*. New York, N.Y.: Dorset House Publishing, 1987.

1.4 Overview of the document

2 Model Description

The logical model decomposition shall be inserted here. To be copied from the IFSI technical note.

3 Specific Requirements

3.1 Functional requirements

Req. ID	Description	Reference
	SWITCH-ON REQUIREMENTS	
	See document AD? for a detailed list of the software requirements for the DPU PROM switch on procedure.	

	TELECOMMAND acceptance.	
OBS-CSR-FU01	The OBS shall be able to receive TC source packets and to issue the TC acceptance report (acknowledge acceptance, see req. OBS-CSR-FU0…) within 15msec from the reception (~1/64 sec).	OBS-CUR-TC06
OBS-CSR-FU02	The OBS shall implement the Telecommand Verification Service. The TC Verification Reports shall have packet types and subtypes in accordance with AD2, section 5.1.2. (Acceptance success: type 1,1; Acceptance failure, type 1,2; Execution Progress, type 1,5; Execution success, type 1,7; Execution failure, type 1,8)	AD2 Sect. 5.1 OBS-CUR-TC03 OBS-CUR-AF10
OBS-CSR-FU02.1	The OBS shall be able to extract the Packet Sequence Control Field from the received TC packet. This is structured into 2 subfields: - seq_flags (2 bits): it shall be set to 11. The OBS shall check this field (TBC) and discard the packet in case of error (? TBC). - Seq_count (14 bits): the OBS shall not check this field. The OBS shall accept the TC regardless the seq_count contents, and shall copy the whole Packet Sequence Control Field into the corresponding part of the TC verification report.	AD2 Sect. 3.1.1.2 and 5.1.2.1 OBS-CUR-TC03
OBS-CSR-FU02.2	The OBS shall be able to interpret the APID field in the TC Packet Header according to the APID values contained in AD2 – Appendix 3. In case of wrong APID, the TC packet shall be rejected and TC acceptance Report/Failure shall be issued, structured according to AD2 sect. 5.1.2.1.	AD2 Sect. 3.1.1.1 OBS-CUR-TC09
OBS-CSR-FU02.3	The OBS shall be able to check the conformity of the Packet Length in the TC Packet header to the actual length (in bytes) of the received TC Packet data field. In case of wrong Length, the TC packet shall be rejected and TC acceptance Report/Failure shall be issued, structured according to AD2 sect. 5.1.2.1.	
OBS-CSR-FU02.4	The OBS shall be able to decode the Ack field in the Data Field Header of the TC Packet Data Field and to generate a TC Acceptance Report indicating acknowledgement of successful acceptance and/or execution of the TC (depending on the bit settings in the Ack field, see AD2, section 3.1.2.1). The acknowledge acceptance of the packet is mandatory.	AD2 Sect. 3.1.2.1 OBS-CUR-TC05
OBS-CSR-FU02.5	The OBS shall be able to verify the integrity of the received TC by checking the Packet Error Control field in the Data Field Header. In case of incorrect Checksum, the TC packet shall be rejected and TC acceptance Report/Failure shall be issued, structured according to AD2 sect. 5.1.2.1. The checksum algorithm is fixed for the complete mission and is defined in AD2, Appendix 4.	

OBS-CSR-FU02.6	The OBS shall be able to check the conformity of the Packet Type/subtype fields in the TC Data Field Header to the allowed values. In case of wrong type/subtype, the TC packet shall be rejected and TC acceptance Report/Failure shall be issued, structured according to AD2 sect. 5.1.2.1.	AD2 Sect. 3.1.2.1 OBS-CUR-TC05
OBS-CSR-FU02.7	The OBS shall be able to check the conformity of the Application Data contained in the TC Data Field Header to the values allowed for that Packet Type/Subtype. In case of wrong Application data, the TC packet shall be rejected and TC acceptance Report/Failure shall be issued, structured according to AD2 sect. 5.1.2.1.	
OBS-CSR-FU02.8	The OBS shall reject non valid packets as the first action after reading the TCs from the input buffer. The command sequence contained in the rejected TCs shall not be executed at all.	AD2 Sect. 5.1.2.1 OBS-CUR-TC10
OBS-CSR-FU02.9	The reasons for success/failure of acceptance of a telecommand shall be reported in the TC Acceptance Report.	OBS-CUR-TC11 OBS-CUR-TC12
	HOUSEKEEPING AND DIAGNOSTIC DATA REPORTING	
OBS-CSR-FU03	The OBS shall acquire the instrument HK data from all instrument subsystems (DPU included) during all nominal modes of the instrument, including any instrument SAFE mode. A nominal HK report will be defined, containing all (TBC) instrument parameters. The structure of the packet is specified in the HK ICD document. *(SPIRE, PACS???)* In addition to the HK paramaters, the OBS shall be able to put in the report the Instrument operating mode (instrument status) at the epoch of the HK acquisition.	OBS-CUR-TM08
OBS-CSR-FU04	The OBS shall packetise the collected HK parameters into dedicated output packets by implementing the HK and Diagnostic data reporting service described in section 5.3 of AD2. (packet type,subtype: 3,25; packet APID in accordance to Appendix 3).	OBS-CUR-TM03 OBS-CUR-TM04
OBS-CSR-FU05	The OBS shall be able to provide the nominal HK report at a normal reporting rate of 1 Hz. For SPIRE and HIFI, the OBS shall send dedicated commands to the subsystems for HK request. These commands will have a lower priority with respect to the measurement commanding. This choice will imply a jitter in the HK acquisition rate of the order of 1-5msec (TBC).	OBS-CUR-TM06
OBS-CSR-FU06	The OBS shall be able to modify, accordingly to dedicated TCs, the pre-defined sets of HK parameters to be included into the HK TM. *The service to be used for this purpose is still TBD.*	OBS-CUR-TM07
OBS-CSR-FU07	The OBS shall provide only actual values of the HK parameters and not changes (or delta values) since the last readout: the *filtered* reporting mode is not allowed. *IS THIS*	OBS-CUR-TM09

	STILL APPLICABLE???	
OBS-CSR-FU08	For diagnostic purposes, it will be possible to generate additional HK packets containing oversampled data. The maximum sampling rate will be 100 samples per second (TBC). *THIS REQUIREMENT IS STILL TBC FOR ALL INSTRUMENTS.*	OBS-CUR-TM11
OBS-CSR-FU09	The OBS shall be able to modify the HK acquisition frequency. *The TC, or sequence of TCs, to be used for this purpose is TBD, as well as the service to implement in board.*	
	EVENT REPORTING	
OBS-CSR-FU10	The OBS shall implement service type 5 (Event Reporting service, AD2, section 5.5) to transmit reports of asynchronous events detected on board. The reports structure/content is defined in TBD applicable document and will be stored in a table on board. The reports type,subtypes will be related to the criticality of the event (nominal event: type,subtype=5,1; Exception event: 5,2; Failure event: 5,4)	OBS-CUR-AF02 OBS-CUR-AF03 OBS-CUR-AF05
OBS-CSR-FU11	There shall be a minimum period (TBD) before the next event packet reporting the same event (e.g. a transition to out-of-limits) can be issued.	OBS-CUR-AF06
	MEMORY MANAGEMENT	
OBS-CSR-FU12	The OBS shall support service type 6 (Memory management service, AD2, section 5.6) to command the memory handling (both DPU and other subsystem memories) procedures on board. The following TC will be supported:	
OBS-CSR-FU12.1	Memory load using absolute addresses: type /subtype 6,2. The OBS shall be able to copy the memory segment contained in the Application data of the TC, in the part of memory indicated in the memory address field. This field, in the case of PACS, will be used also to address memories in other subsystems.	
OBS-CSR-FU12.2	Memory dump using absolute addresses: type /subtype 6,5. The corresponding dump TM report, will have type/subtype 6,6.	
OBS-CSR-FU12.3	Memory check using absolute addresses: type /subtype 6,9. The corresponding dump TM report, will have type/subtype 6,10.	
OBS-CSR-FU12.4	All the Memory reports will have an APID TBD.	
	FUNCTION MANAGEMENT	
	All requirements TBW. THEY ARE INSTRUMENT DEPENDENT.	

3.2 Performance requirements

Req. ID	Description	Reference
	TELECOMMANDS REQUIREMENTS	
OBS-CSR-PE01	The OBS shall be able to receive TC source packets and to issue the TC acceptance report (acknowledge acceptance, see req. OBS-CSR-FU0…) within 15msec from the reception (~1/64 sec).	OBS-CUR-TC06

3.3 Interface requirements

Req. ID	Description	Reference
	MIL-STD-1553B PROTOCOL IMPLEMENTATION REQUIREMENTS:	
OBS-CSR-IF01	The OBS shall act as a Remote Terminal (RT) implementing all the functionalities allowed by the MIL1553B STD. The RT shall be compliant with the Transfer Layer protocol described in AD2 Appendix 9. In particular:	
OBS-CSR-IF02	RT shall use the interface subaddresses allocated according to table 3.2.3-1 in AD2, Appendix 9.	AD2 3135, 3140-TFL-T
OBS-CSR-IF03	The satellite Data Bus Protocol has a periodic structure, based on a 1 sec period called Frame, divided into 64 subframes. Each subframe has a timing structure of 24 message slots. The slots have different time durations, allocated according to AD2, Appendix 9, table 4.1.3.1-1. The first message slot of each subframe (except the first subframe of the frame) is dedicated to the transmission of the mode command “Sync with data word”: RT shall be able to decode the Subframe User field of this command (see AD2, Appendix 9, table 4.2-1), in order to see if it is allowed to send its TM data within the current subframe.	AD2 4275-TFL-T AD2 4280-TFL-T
OBS-CSR-IF04	The first message slot of the first subframe every second is dedicated to the transmission of the “Sync without data word” mode command. RT shall not use this subframe for TM packets transfer.	AD2 4125-TFL-T AD2 4130-TFL-T AD2 4135-TFL-T
OBS-CSR-IF05	RT shall have an internal subframe counter, which shall be reset to 0 at the reception of the first subframe every	AD2 4300-TFL-T

	second, and shall be incremented by 1 at the reception of each one of the other subframes.	
OBS-CSR-IF06	RT shall be able to provide the BC with its health status data and additional information (according to the layout in AD2, Appendix 9, figure 4.4-1) directly into SA1T, at least once per second. (low-level requirement?)	AD2 4350-TFL-T AD2 4355-TFL-T AD2 4360-TFL-T
OBS-CSR-IF07	RT shall be able to acquire all the 1553 messages belonging to the same TC packet in the same subframe.	AD2 4400-TFL-T
OBS-CSR-IF08	RT shall be able to execute command identification through the interpretation of a TC counter set by the BC.	AD2 4410-TFL-T
OBS-CSR-IF09	RT shall read the TC packet into the TC Data Receive SAs, starting from SA 11R.	AD2 4415-TFL-T
OBS-CSR-IF10	RT shall read the Packet Transfer Descriptor Message in SA 27R and interpret it according to AD2, Appendix 9, table 4.5.1-1. The Descriptor is referred to the TC packet to be transmitted within the next subframe (after the next subframe synch).	AD2 4420-TFL-T AD2 4425-TFL-T
OBS-CSR-IF11	RT shall provide a TC packet Confirmation, by putting the info listed in AD2, Appendix 9, table 4.5.1-2 into SA 27T.	AD2 4430-TFL-T
OBS-CSR-IF12	For a correct TC low level acceptance, RT shall implement the procedure suggested in AD2, Appendix 9, fig. 4.5.1-1	AD2 4435-TFL-N
OBS-CSR-IF13	RT shall support the receiving of asynchronous short event driven TC packets, issued by BC for special instrument control functions (TBC). Sub-addresses SA3R and SA 4R will be used for this. (*Shall the procedure in this case be interrupt driven? See note at the end of section 4.5.2)*	AD2 4450-TFL-T
OBS-CSR-IF14	No on board time synchronisation through the time information provided by the BC will be performed by the OBS. The beginning of the mode command "Synch without data word" in the first message slot of subframe 1 shall be used for implementing the time sinchronisation high level procedure described in AD2 sect. ????. The corresponding time accuracy will be of 150 microsec (TBC).	AD2 4330-TFL-T
OBS-CSR-IF15	RT low level commanding requirements: TBW. *Are they necessary?*	
OBS-CSR-IF16	In order to transfer TM packets, RT shall implement the procedure whose logical flow is reported in figure 4.6.1.3-1 of AD2, Appendix 9.	AD2 4690-TFL-T
OBS-CSR-IF17	RT shall request a TM packet transfer by setting a Transfer Request into SA 10T, with a layout in accordance to the description in table 4.6.1.1-1 (AD2, Appendix 9) and to the detailed requirements of section 4.6.1.1.	AD2 4500 ÷4570-TFL-T
OBS-CSR-IF18	RT shall support an internal circular packet counter, with an allowed range from 0 to 255, to be used in order to detect the completion of an TM packet transfer transmission. The value of the counter shall be reported in the Transfer Request and shall be checked in the Packet	AD2 4575 ÷4600-TFL-T

	Transfer Confirmation provided by BC.	
OBS-CSR-IF19	RT shall read the TM Packet transfer confirmation in SA10R and decode it according to the layout in table 4.6.1.2-1 (AD2, Appendix 9). This shall be done at the latest after the next Synch mode command, in order to prepare the next packet data transfer: TM packet data and the new (if any) transfer request shall be ready within 2 msec after the Synch Mode command reception.	4635 4690
OBS-CSR-IF20	The TM packets transmission ends up with the issuing of a particular TM Packet Transfer request, with the first word set to 0 and without any increment in the packet count field.	4695
OBS-CSR-IF21	Burst mode detailed requirements TBW.	
OBS-CSR-IF22	In order to transfer Event TM Packets , RT shall implement the procedure whose logical flow is reported in figure 4.6.2-1 of AD2, Appendix 9. (Event packets are the small data packets provided as output of services 1, TC verification service, and 5 , event reporting)	4865
OBS-CSR-IF23	RT shall put the event messages in the dedicated sub-addresses SA5T and SA6T. (maximum length of messages = 64 bytes)	4800, 4805 3165-DLL-T
OBS-CSR-IF24	RT shall be able to set the event flag in the Packet Transfer Request. The flag shall be reset only after a successful transfer.	4805, 4840
OBS-CSR-IF25	RT shall check if the event has been correctly transferred by controlling the contents of SA 5R and SA6R. This operation shall be done every subframe Synch until the event has been transferred.	4810, 4835
OBS-CSR-IF26	RT can rise a new event flag of the same type (either event in SA5T or event in SA6T) only after two subframes after the flag reset.	4845
	SUBSYSTEM INTERFACES	
OBS-CSR-IF27	There are two types of interfaces with the subsystems *(VALID FOR SPIRE AND HIFI ONLY…NEED INPUT FROM STEFANO; FOR THE 1355 I/F USED IN PACS)* The OBS shall support the communication protocols described in the ICD documents.	
OBS-CSR-IF28	There shall be different dedicated tasks for handling the two interfaces: Low speed interface nominal commanding task; HK acquisition handling task; High speed interface task: Interrupt driven, being the interrupt linked to the half FIFO full HW signal. (the minimum number of these HS I/F tasks is TBD) Another high priority interrupt driven procedure shall be foreseen to guarantee the most stringent timing requirements on some commands (HIFI and SPIRE)…. *PROBABLY THIS RFEQ SHALL BE MOVED TO THE*	

	OPERATIONAL.	
OBS-CSR-IF29		
OBS-CSR-IF30		
OBS-CSR-IF15		
OBS-CSR-IF16		

3.4 Operational requirements

Req. ID	Description	Reference
OBS-CSR-OP01	All parameters used to assess the conformity of the packet shall be stored in updateable on board tables. The detailed format of the tables is described in TBD. *The minimum list of tables needed for each instrument is :* – *APID,* – *Allowed Packet type/subtype,* – *Application data tables - subsystem parameters tables, possibly structured in order to contain in each row the function ID, which will be the key to access the table, followed by the maximum and minimum allowed values of all the parameters uploadable from ground.*	OBS-CUR-TC08
OBS-CSR-OP02	There shall be an input buffer for the received TCs dimensioned considering a maximum command length of 248 bytes and able to couple the maximum command transmission rate with the time needed on average to unpack and execute the commands.	OBS-CUR-TC04
OBS-CSR-OP03	There shall be in the OBS a task dedicated only to the CDMS interface handling, communicating with other OBS processes via the IPC protocols supported by the adopted operating system.	OBS-CUR-TC07
OBS-CSR-OP04	The Tranfer Layer Protocol *mode command* decoding procedure shall not last more than 100 microsec. The CDMS interface task shall be interrupt driven, being the interrupt linked to the presence of the new Mode command word in SA 00 of the RT. This high priority interrupt will have a periodicity of 1/64 sec.	4070-TFL-T 4330-TFL-T IFSI
OBS-CSR-OP05	*The Burst Mode is foreseen for PACS only. Its presence, coupled with the limited 1355 I/F internal buffering capability, has important consequences on the dimensioning of the science data internal buffers. (STEFANO, puoi inserire qui i conti che avevamo fatto??)*	IFSI TBW
OBS-CSR-OP06	The OBS shall execute the TCs in the order they are received and stored in the input buffer. There shall be the possibility of executing immediate TCs (e.g. ABORT) when they are received (within TBD μs), out of the TC sequence.	OBS-CUR-TC06
OBS-CSR-OP07	In different modes of the instruments, some of the HK parameters to be reported into the nominal HK packet will not be valid. This info will be stored onboard in the HK	OBS-CUR-TM05

	packet definition table. The OBS shall flag the invalid parameters in the output report, according to the indication in the table.	
OBS-CSR-OP08	There shall be a dedicated buffer for storing the asynchronous events generated on board. This buffer will be polled by the CDMS I/F task as a first priority at its activation, and the presence of an event will be signalled to the CDMS immediately. The maximum latency of an event on board shall be equal to two subframes.(TBC)	
OBS-CSR-OP09	All the buffers on board will be circular buffers, accessible by more than one process, via the utilisation of separated input and output pointers. When a process will access a buffer (either for reading it or for writing on it) it will lock it with a dedicated semaphore *(mutex?, TBC).*	
OBS-CSR-OP10	The OBS shall support the program memory (PM) patching. This will imply the modification of the executable and this cannot be done while the code is running. Therefore the development of a dedicated procedure shall be foreseen, to be run separated from the OBS application. The overall patching procedure will be performed by shutting down the OBS application, by running the patching function and by restarting the OBS based on the new executable. TBC. This procedure will guarantee a RAM patching. If an EEPROM patching is needed, before restarting the OBS application, the new executable shall be copied from RAM to EEPROM and a complete system reboot shall be performed. *THE OVERALL PROCEDURE SHALL BE DISCUSSED. PROBABLY THE REQUIREMENT SHALL BE SPLITTED IN MANY SUBREQUIREMENTS.* *THE PATCHING PROCEDURE WILL BE DESCRIBED IN THE DDD.*	
OBS-CSR-OP11		
OBS-CSR-OP12		
OBS-CSR-OP13		
OBS-CSR-OP14		
OBS-CSR-OP15		
OBS-CSR-OP16		

3.5 Resource requirements

3.6 Verification requirements

3.7 Acceptance testing requirements

3.8 Documentation requirements

3.9 Security requirements

3.10 Portability requirements

3.11 Quality requirements

3.12 Reliability requirements

3.13 Maintainability requirements

3.14 Safety requirements

4 System Design

4.1 Design Method

We have adopted the methodology of Real-Time Structured Analysis and Architecture Modeling. by T. de Marco, known as Yourdon - De Marco Structured Analysis (RD1), and D. J. Hatley & I.A. Pirbhai (RD2).

One of the most important concepts of the method is that the Structured Analysis model should be independent of the technology that will be used to implement the system. The technology decisions will be added in the Architecture model.

In Structured Analysis, the system is intended as a *process* that transforms the inputs of the system into its outputs. This process is then decomposed or refined into a set of simpler *child processes*. The child processes are then further refined as necessary until processes that require no further refinement are reached. The processes that require no further refinement are called *primitive processes*. A primitive process is one that completely describes it transformation of inputs to outputs in an unambiguous and testable way.

Process input and output data are defined as data dictionary items, or, *data items* for short. Each data item is defined and decomposed to its primitive elements. The data item definition is composed of a textual description and a structural description . Primitive data items only have a textual description. A simplified BNF (Backus-Naur) format is used to describe the data item structure. Data items are stored in the Data Dictionary.

Architectural Modeling is a method of defining the components that will be used to implement the essential requirements defined in the Structured Analysis model.
In Architecture Modeling, a *module* is a component of the system that implements a set of Structured Analysis processes and a *channel* serves as a link carrying Structured Analysis data items between modules.
Architecture flow diagrams, *architecture interconnect diagrams* and *module specifications* are used to describe a module.
An architectural flow diagram is a graphical means to describe a module. It shows the flow of information (data items) between modules. The top level architecture flow diagram is the *architecture context diagram*.
An architecture interconnect diagram is a graphical means to describe a module. It shows the channels which connect the modules and via which the data flows between the modules.
The integrity of the system is validated by checking the consistency between the Structured Analysis and Architecture Models.

In the Logical model description, the following tools will be used:

- Data Flow Diagrams
- Control Flow Diagrams

- Data Dictionary
- Control and Flow Process Specifications
- Decision Tables, Process Activation Tables and State/Event Matrices

Architecture Diagram tools TBW.

4.2 Decomposition description

4.2.1 OBS Architecture: Architecture context diagram

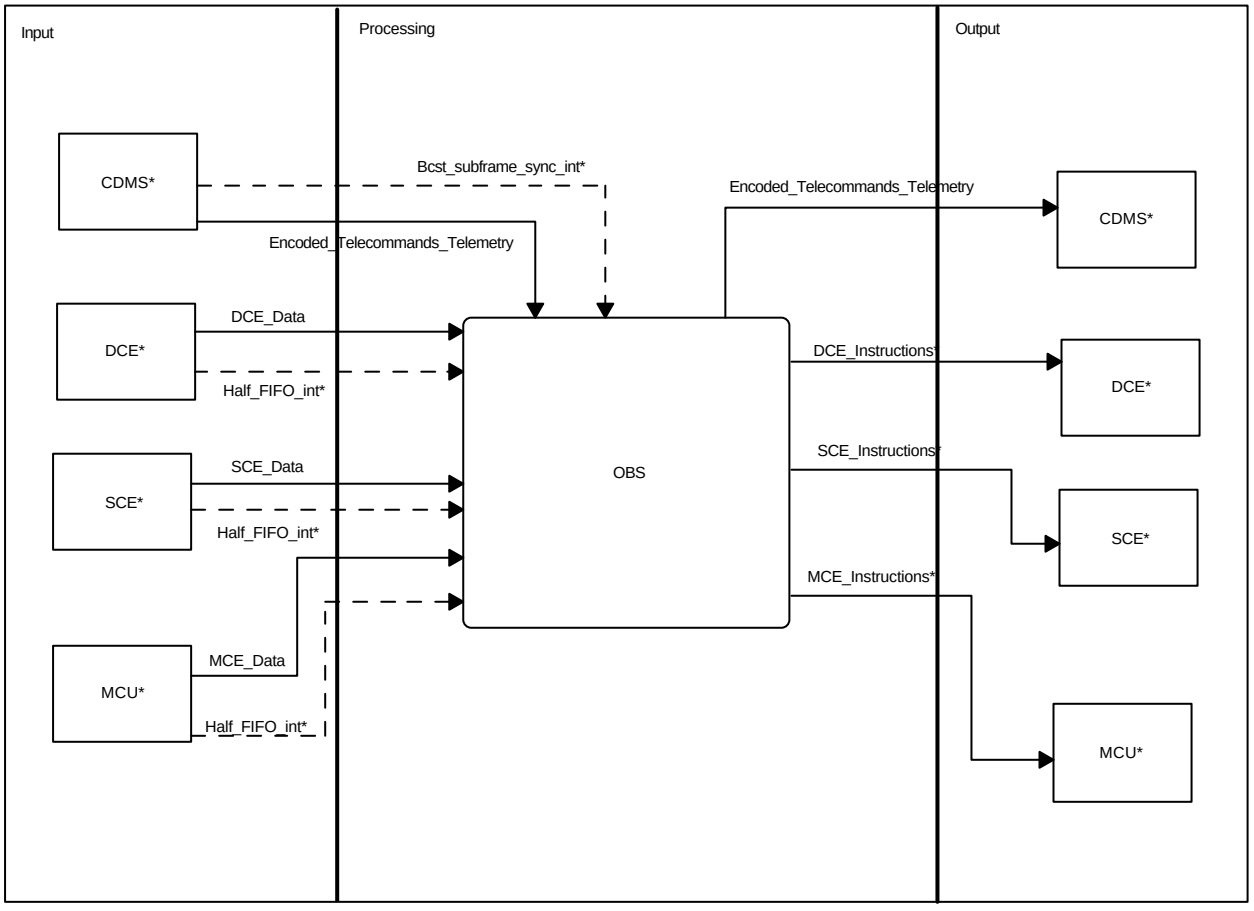

Figure 1: Architecture Diagram OBS_arch0

Short description TBW

4.2.2 OBS Architecture: level 0 decomposition

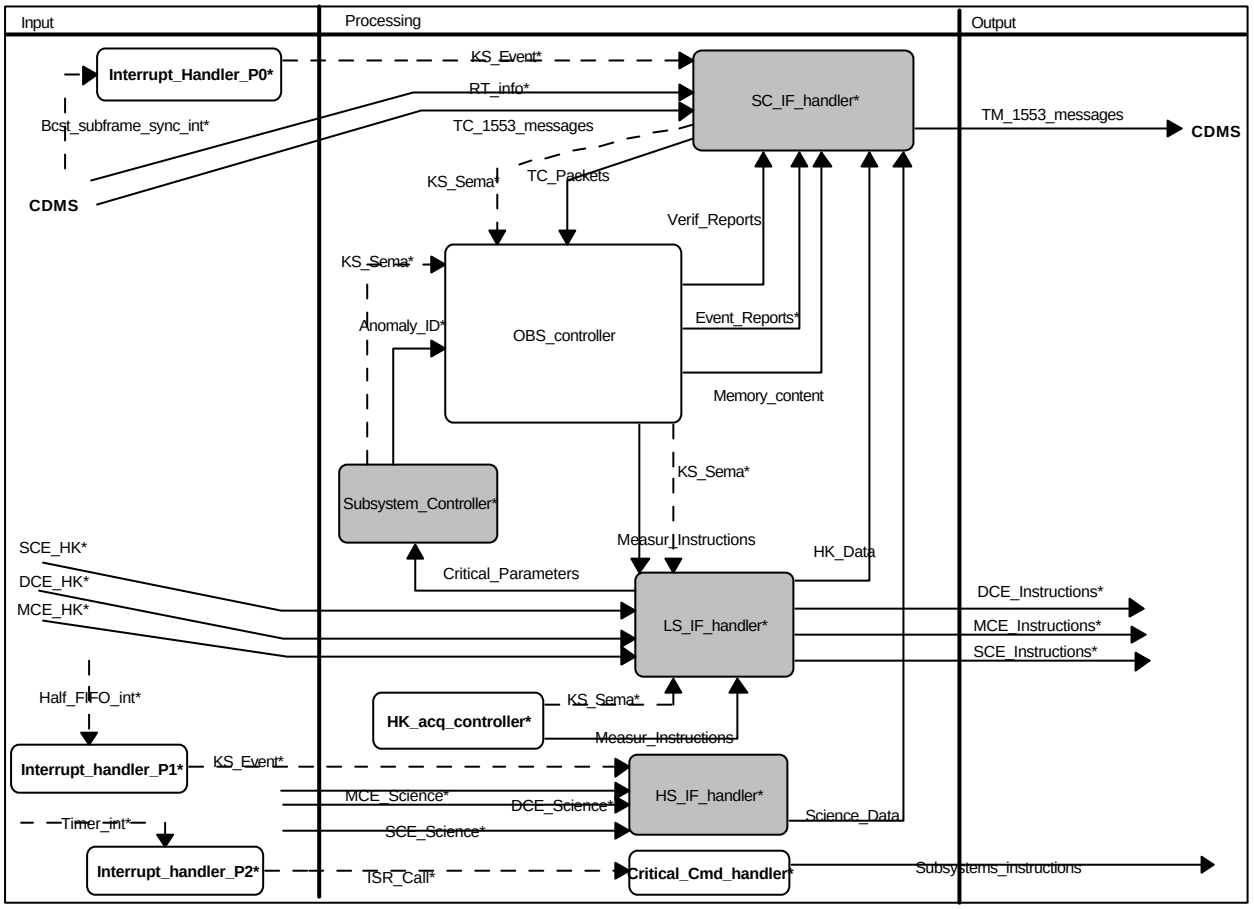

Figure 2: Architecture Diagram OBS

Description TBW

5 Component description

5.1 OBS Architecture: OBS_controller task module decomposition

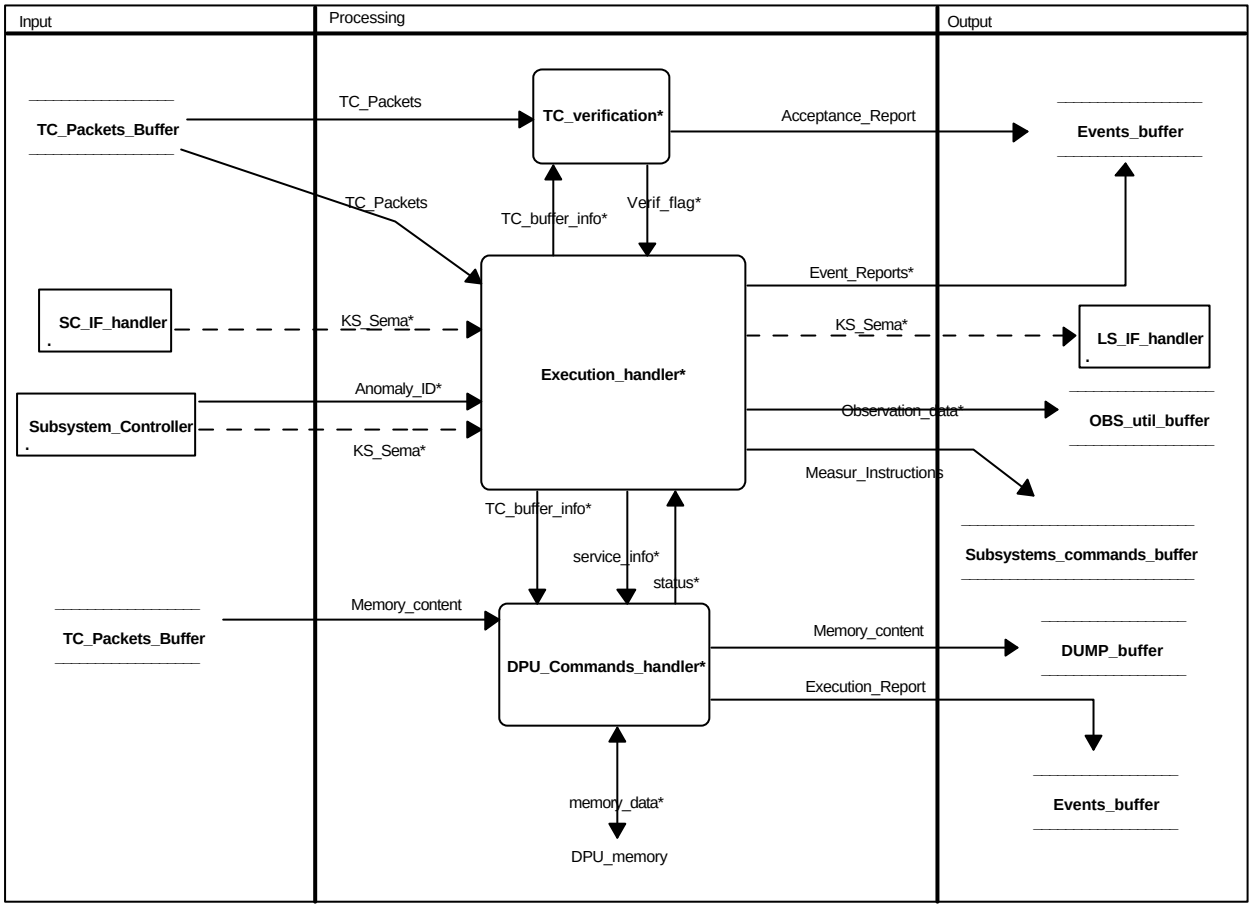

Figure 3: Architecture Diagram OBS_controller

Description TBW

5.1.1 OBS Architecture: Execution_handler - function decomposition

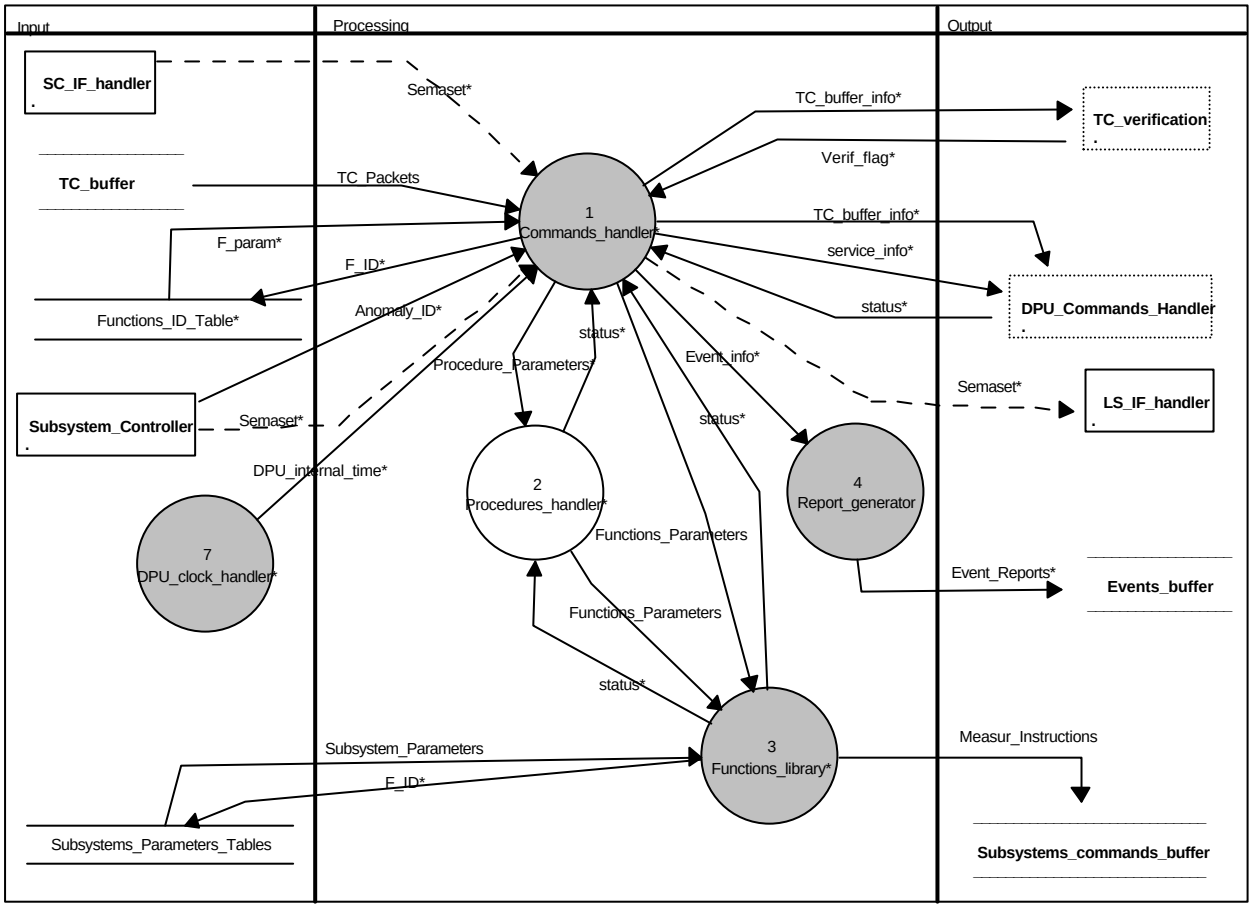

Figure 4: Enhanced Flow Diagram Execution_handler

Description TBW

5.1.2 OBS Architecture: DPU_Commands_handler – functions decomposition

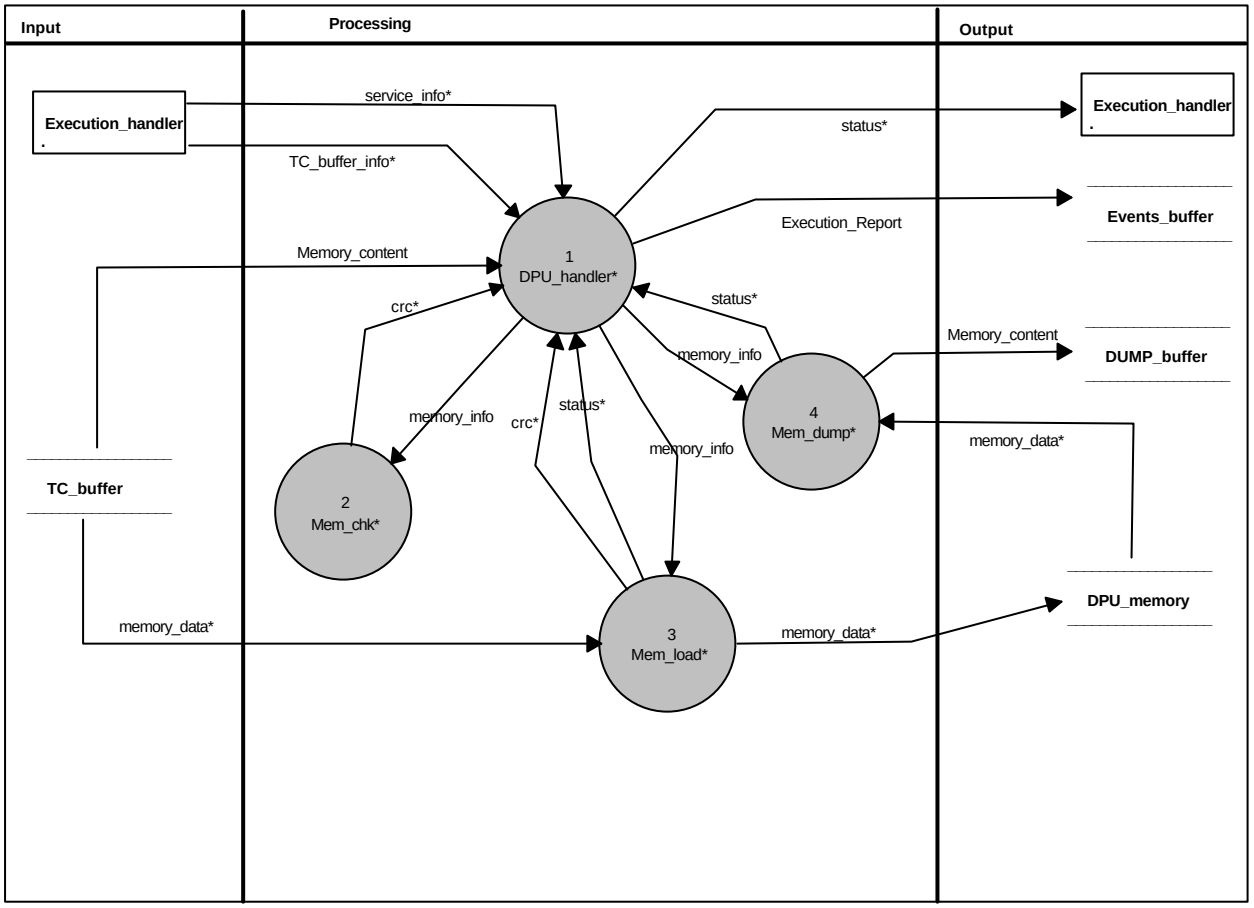

Figure 5: Enhanced Flow Diagram DPU_Commands_handler

Description TBW

5.2 OBS Architecture: SC_IF_handler task/module – function decomposition

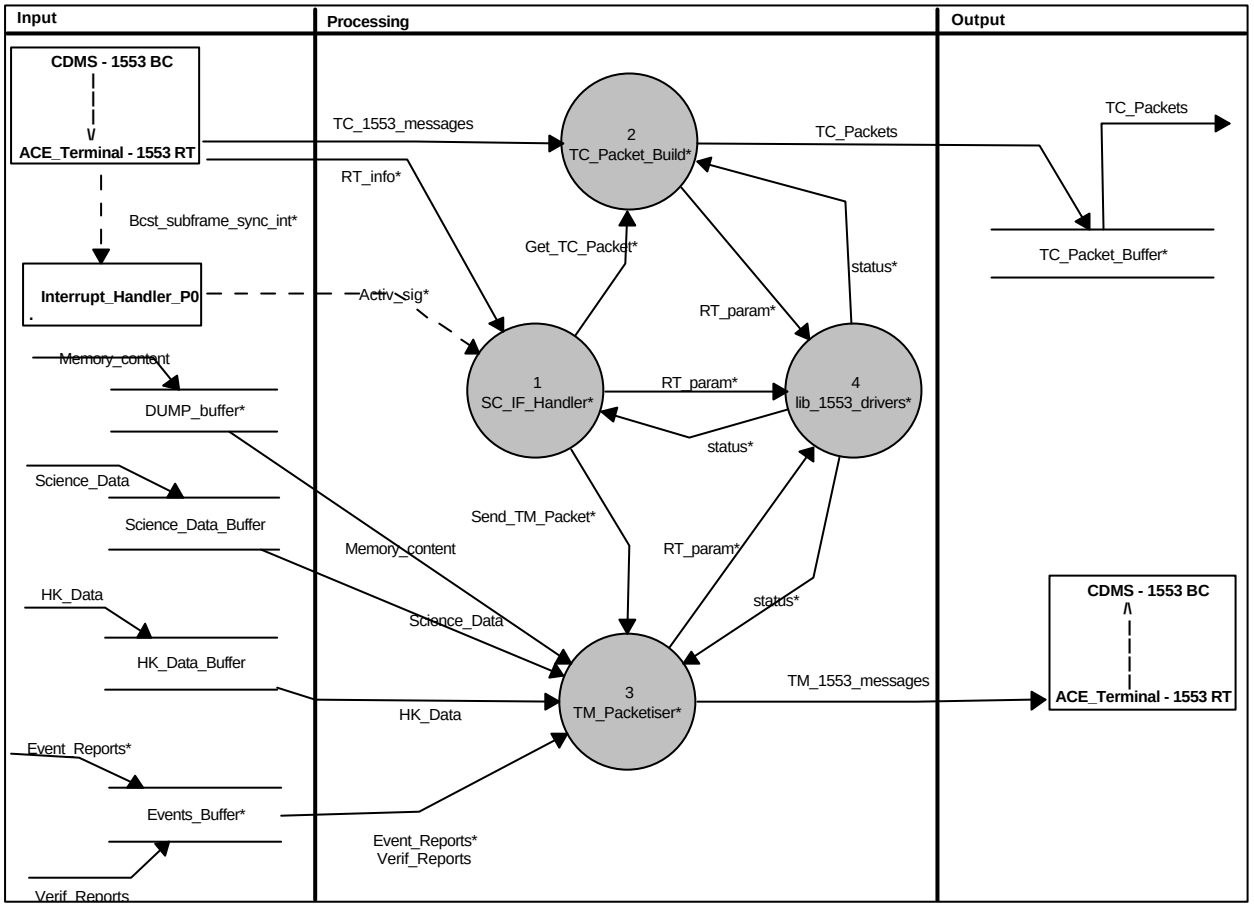

Figure 6: Enhanced Flow Diagram SC_IF_handler

Description TBW

5.3 OBS Architecture: LS_IF_handler task/module – function decomposition

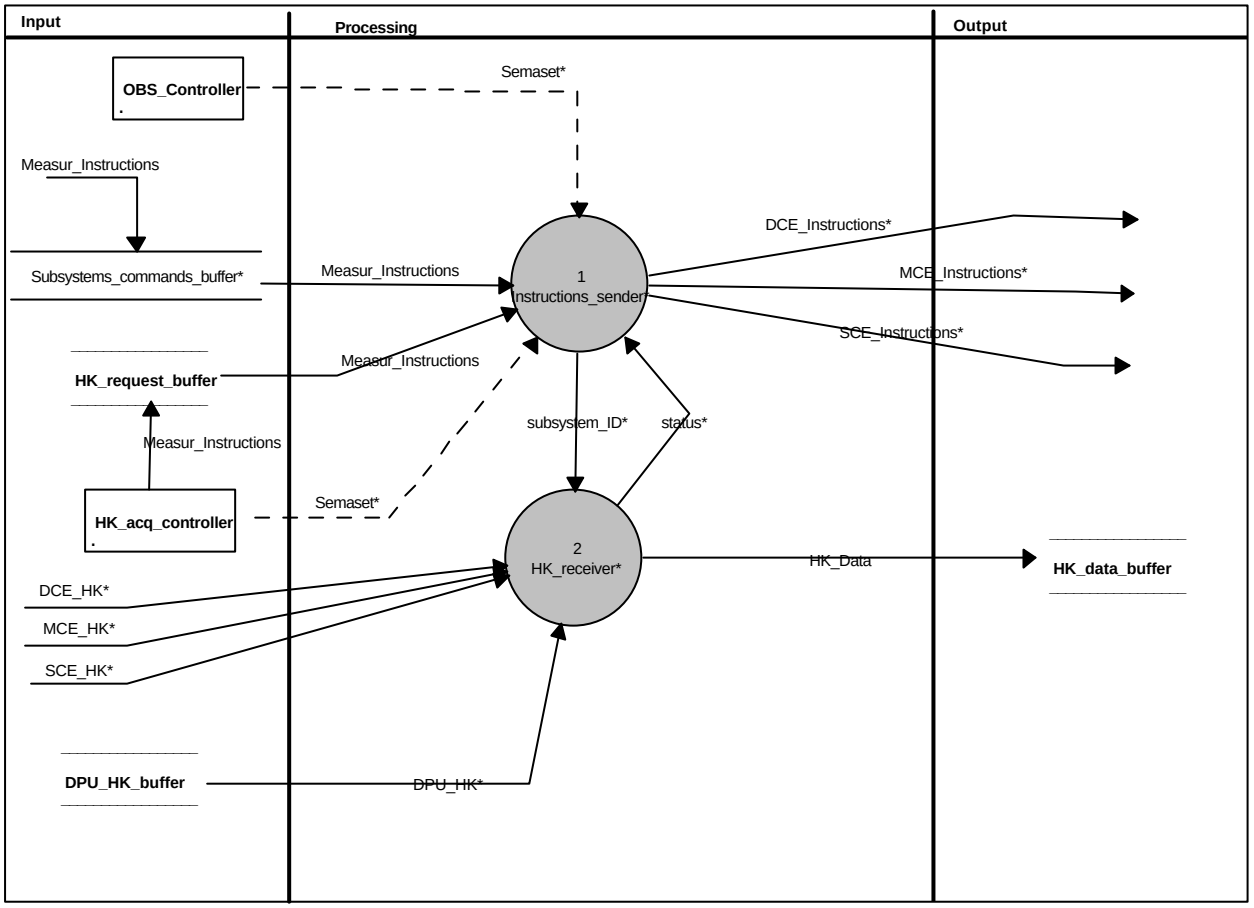

Figure 7: Enhanced Flow Diagram LS_IF_handler

Description TBW

5.4 OBS Architecture: HS_IF_handler task/module – function decomposition

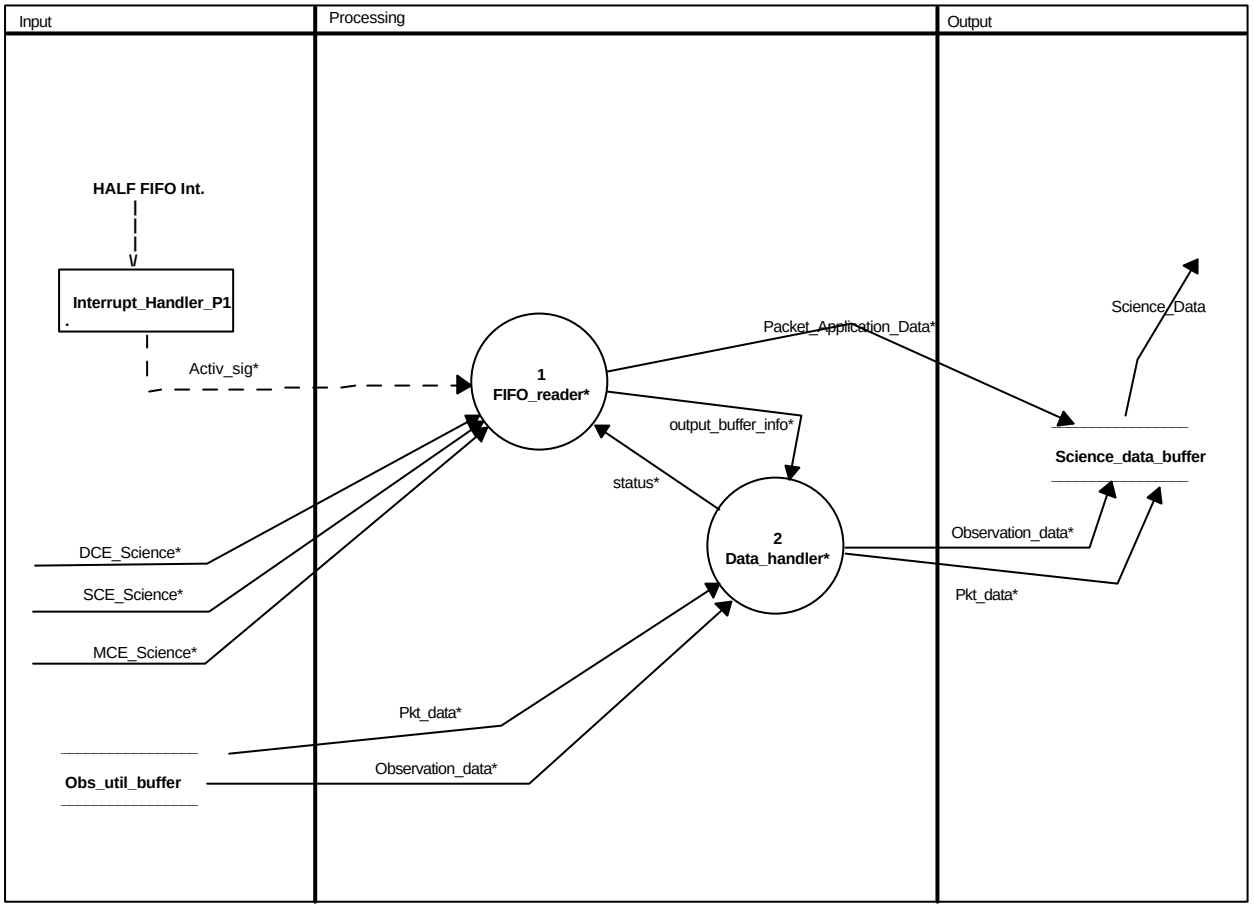

Figure 8: Enhanced Flow Diagram HS_IF_handler

Description TBW

5.5 OBS Architecture: Subsystem_Controller

TBW

5.6 OBS Architecture: Interrupt_Handler_P0

TBW

5.7 OBS Architecture: Interrupt_handler_P1

TBW

5.8 OBS Architecture: Interrupt_handler_P2

TBW

6 Feasibility and Resources Estimates

TBW

7 User Requirements vs Software Requirements Traceability matrix

TBW

8 Software Requirements vs Components Traceability matrix

TBW