

# OBOSS System Integration Frame

## Contract

### OBOSS-III

### Operations Manual

**Document No.:** TERMA/SPD/OBOSS-III/012  
**Date:** 11.06.2004  
**Issue:** 1  
**Revision:** 1  
**Distribution:** ESTEC, Terma A/S  
**Author:** Keld Schultz  
**Technical Review:** Gert Caspersen  
**Authorised by:** Gert Caspersen  
**Approved by:** Carsten Jørgensen

## Document Change Record

Issue	Date	Change
1 -	05.02.2004	New document
1 1	11.06.2004	Updated after Acceptance Review. Indication of applied scheduling mechanism added as chapter 6. Installation instructions added as chapter 7.

## Document Status Sheet

Page	Issue
i - iv	1 1
1 - 19	1 1

© Terma A/S, 2004

The copyright of this document is vested in Terma A/S.

This document may only be reproduced in whole or in part, stored in a retrieval system, transmitted in any form, or by any means electronic, mechanical, photocopying, or otherwise, with the prior permission of Terma A/S.

<b>Table of Contents</b>	<b>Page</b>
<b>1 Introduction</b> .....	<b>1</b>
1.1 Purpose .....	1
1.2 Scope .....	1
<b>2 References</b> .....	<b>2</b>
2.1 Applicable Documents .....	2
2.2 Reference Documents .....	2
<b>3 Abbreviations, Terms and Definitions</b> .....	<b>3</b>
3.1 Abbreviations .....	3
3.2 Terms and Definitions .....	3
<b>4 Overview</b> .....	<b>4</b>
4.1 Packet Utilisation Standard .....	4
4.2 Data Handling System Platform .....	4
<b>5 Interpretation of PUS</b> .....	<b>6</b>
5.1 Supported Services .....	6
5.2 Telecommand Verification .....	6
5.3 Device Command Distribution .....	9
5.4 Housekeeping and Diagnostic Data Reporting.....	9
5.5 Event Reporting .....	10
5.6 Memory Management .....	11
5.7 Function Management .....	12
5.8 On-Board Operations Scheduling .....	12
5.9 On-Board Monitoring .....	13
5.10 Large Data Transfer.....	14
5.11 On-Board Storage and Retrieval.....	16
5.12 Event-Action .....	17
<b>6 Scheduling</b> .....	<b>18</b>
6.1 Scheduling Mechanism .....	18
<b>7 Installation</b> .....	<b>19</b>
7.1 Installation Instructions .....	19

## List of Figures

Figure 4.2-1: Data Handling System Platform.....	4
Figure 5.2-1: Telecommand Verification Scheme .....	7

## List of Tables

Table 5.1-1: Supported PUS Services in OBOSS-III .....	6
Table 5.2-1: Supported Telecommand Verification Service Sub-Types .....	7
Table 5.2-2: Telecommand Verification Failure Code Values .....	9
Table 5.3-1: Supported Device Command Distribution Service Sub-Types .....	9
Table 5.4-1: Supported Housekeeping and Diagnostic Data Reporting Service Sub-Types .....	10

---

Table 5.5-1: Supported Event Reporting Service Sub-Types .....	11
Table 5.5-2: Event Reports Reserved by OBOSS-III.....	11
Table 5.6-1: Supported Memory Management Service Sub-Types .....	12
Table 5.7-1: Supported Function Management Service Sub-Types.....	12
Table 5.8-1: Supported On-Board Operations Scheduling Service Sub-Types.....	13
Table 5.9-1: Supported On-Board Monitoring Service Sub-Types .....	14
Table 5.10-1: Supported Large Data Transfer Service Sub-Types .....	15
Table 5.10-2: OBOSS-III Reason Codes.....	16
Table 5.11-1: Supported On-Board Storage and Retrieval Service Sub-Types .....	16
Table 5.12-1: Supported Event-Action Service Sub-Types .....	17

---

# **1 Introduction**

## **1.1 Purpose**

The On-Board Operations Support Software (OBOSS) is a collection of reusable software components that can be used to build a data handling system. The OBOSS-III Operations Manual provides users of such a data handling system with basic information about the OBOSS functionality. A data handling system based on OBOSS-III components is referred to as an instantiation of OBOSS-III. The requirements towards OBOSS-III can be found in [AD1].

## **1.2 Scope**

The Operations Manual is not targeted towards any specific OBOSS-III instantiation. A specific instantiation will usually only include a subset of the OBOSS-III functionality. Details about which subset of OBOSS-III that are provided must be found in mission specific documentation.

---

## **2 References**

### **2.1 Applicable Documents**

The documents, which are applicable for this document, are:

- [AD1] "OBOSS System Integration Frame Contract – Proposal for Call-Off 1", Terma A/S, doc. no. Terma/SPD/OBOSS-III/001, issue 1.1, February 2003.
- [AD2] "OBOSS System Integration Frame Contract – OBOSS-III Software Requirements", Terma A/S, doc. no. Terma/SPD/OBOSS-III/003, issue 2.1, November 2003.

### **2.2 Reference Documents**

The documents, except for the applicable documents, which are referenced in this document, are:

- [RD1] "Space Engineering: Ground Systems and Operations – Telemetry and Telecommand Packet Utilization", European Cooperation for Space Standardization, doc. no. ECSS-E-70-41A, 30 January 2003.
- [RD2] "ESA Packet Telecommand Standard", European Space Agency, doc. no. PSS-04-107, Issue 1, June 1990.
- [RD3] "ESA Packet Telemetry Standard", European Space Agency, doc. no. PSS-04-106, Issue 1, January 1988.
-

## **3 Abbreviations, Terms and Definitions**

### **3.1 Abbreviations**

Abbreviations used in this document:

APID	Application ID
CCSDS	Consultative Committee for Space Data Systems
COM	Communication System
CPDU	Command Pulse Distribution Unit
CUC	CCSDS Unsegmented Code
ESA	European Space Agency
FIFO	First In First out
ID	Identifier
I/F	Interface
OBOSS	On-Board Operations Support Software
PUS	Packet Utilisation Standard
RID	Report ID
SAU	Smallest Addressable Unit
SDU	Service Data Unit
TC	Telecommand
TM	Telemetry

### **3.2 Terms and Definitions**

Terms and definitions specific for this document:

Byte Order	The OBOSS software is developed as a set of reusable software components which in principle should be independent of the hardware platform it is running on. The current version however which is developed to run on an ERC32 processor assumes big endian format, i.e. most significant bit in most significant byte transmitted first.
------------	---

---

## 4 Overview

The goal of the OBOSS project is to develop a reusable onboard software architecture implementing a subset of the services defined in Packet Utilisation Standard [RD1] on a command and data handling platform. This architecture shall be able to support and ease (through intensive reuse) development of onboard command and data handling software for a series of future satellite missions.

Consequently, the context of the OBOSS-III is dominated by three elements: the Packet Utilisation Standard (PUS), the satellite platforms on which the OBOSS-III architecture is to build, and the reusable software components being part of the reusable architecture. Each of these is discussed in the following.

### 4.1 Packet Utilisation Standard

This standard defines several general services whose onboard implementation supports satellite operation. It defines the application level of Packet Telecommand Standard [RD2] and Packet Telemetry Standard [RD3].

It is not in the scope of the OBOSS-III project to develop a complete PUS implementation. Consequently, a subset of services and sub-services has been selected for implementation.

### 4.2 Data Handling System Platform

The OBOSS-III architecture will be affected by any assumptions made regarding the command and data handler platform on which it is to reside. This will define the environment of the command and data handler. The environment depicted in Figure 4.2-1 is considered as the baseline. Each element is briefly described below.

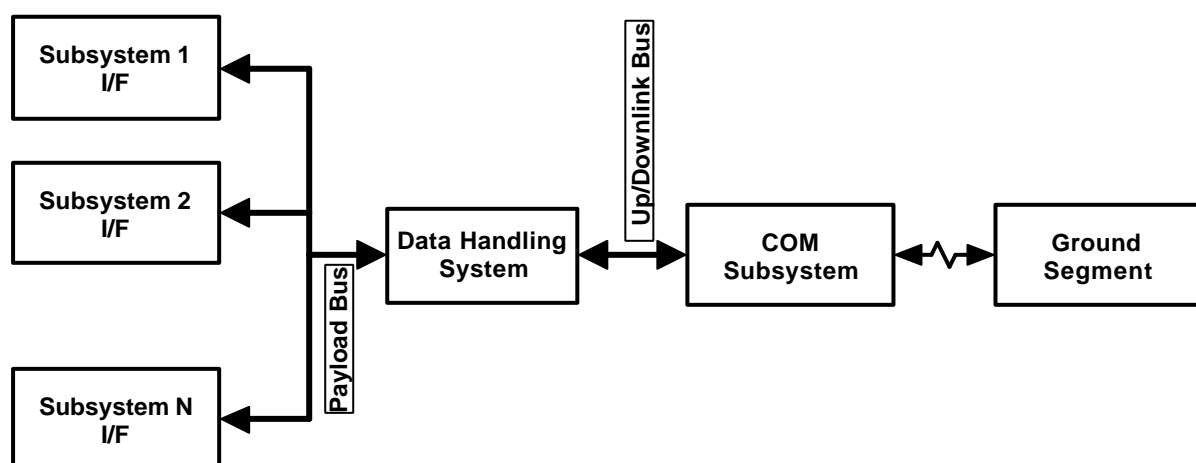


Figure 4.2-1: Data Handling System Platform

### **Ground Segment**

Control centre, ground stations etc. responsible for operation of the satellite on which command and data handler platform resides.

### **COM Subsystem**

Communication subsystem providing the onboard support for the radio link between space segment and ground segment. It implements several layers in the ESA Packet Telecommand Standard [RD2] and Packet Telemetry Standard [RD3].

### **Up/Downlink Bus**

Onboard bus dedicated to uplink and downlink data flows.

### **Data Handling System**

Onboard subsystem responsible for commanding of onboard subsystems and data collection from these.

### **Payload Bus**

Any bus chosen for interfacing to payloads. May be multidrop bus – as shown – or serial busses in star shaped configuration.

### **Subsystem 1 I/F ... Subsystem N I/F**

Collection of onboard subsystems controlled by the data handling system. Includes payloads as well as attitude control subsystem, electrical power subsystem etc.

---

## 5 Interpretation of PUS

Although the packet utilisation standard aims at being unambiguous and complete, room is still left for interpretation. Any implementation may have to deviate due to limitations imposed by the target platform. For each of the supported PUS services this chapter describes any interpretations and limitations present in the OBOSS-III implementation.

### 5.1 Supported Services

Only a subset of the services defined in the standard is supported in the OBOSS-III implementation. These are marked in Table 5.1-1.

Service Type	PUS Service	Supported
1	Telecommand Verification	✓
2	Device Command Distribution	✓
3	Housekeeping and Diagnostic Data Reporting	✓
4	Parameter Statistics Reporting	
5	Event Reporting	✓
6	Memory Management	✓
8	Function Management	✓
9	Time Management	
11	On-Board Operations Scheduling	✓
12	On-Board Monitoring	✓
13	Large Data Transfer	✓
14	Packet Forwarding Control	
15	On-Board Storage and Retrieval	✓
17	Test	
18	On-Board Operations Procedure	
19	Event-Action	✓

**Table 5.1-1: Supported PUS Services in OBOSS-III**

In the following sections, each of the supported services will be dealt with.

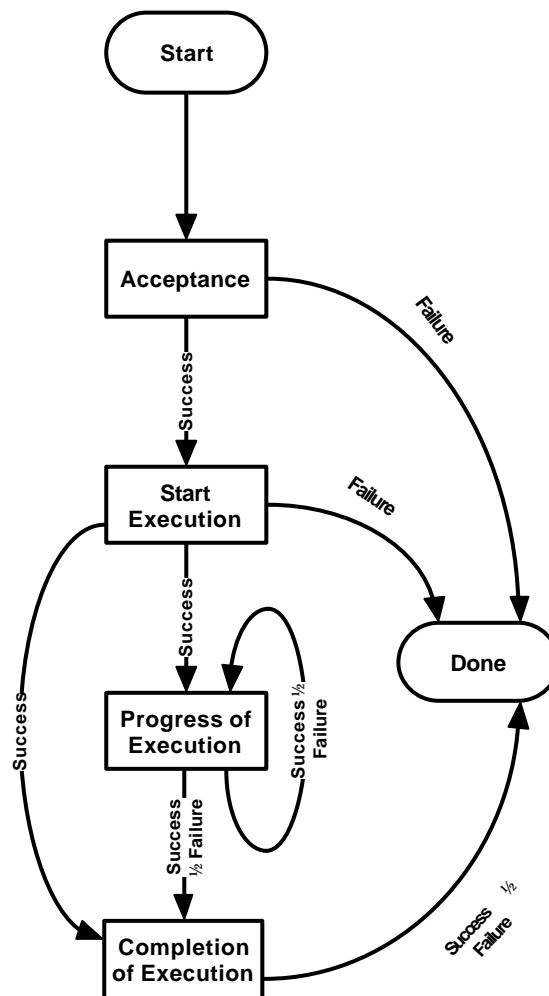
### 5.2 Telecommand Verification

For this service, all service sub-types are supported as listed in Table 5.2-1.

Service Sub-Type	Name	Supported	Comments
1	Telecommand Acceptance Report – Success	✓	
2	Telecommand Acceptance Report – Failure	✓	
3	Telecommand Execution Started Report – Success	✓	
4	Telecommand Execution Started Report – Failure	✓	
5	Telecommand Execution Progress Report – Success	✓	
6	Telecommand Execution Progress Report – Success	✓	
7	Telecommand Execution Completed Report – Success	✓	
8	Telecommand Execution Completed Report – Failure	✓	

**Table 5.2-1: Supported Telecommand Verification Service Sub-Types**

Telecommand verification packets are generated according to the scheme depicted in Figure 5.2-1.



**Figure 5.2-1: Telecommand Verification Scheme**

The general ideas are:

- If execution of a telecommand involves several steps (e.g. insertion of values into a table) then each of these will be verified by an execution progress report.
- Should execution of a step fail the following steps will still be executed.
- Completion of execution fails if any step has failed.

The generation of telecommand verification packets is controlled by the acknowledge field of the telecommand in question.

Sub-types reporting telecommand failure includes a Code field indicating the reason for telecommand failure. Table 5.2-2 lists the OBOSS specific values for the Code field. Notice that values 0 to 5 are defined by the standard.

Code	Interpretation
0	Illegal APID
1	Incomplete or invalid packet length
2	Incorrect checksum
3	Illegal packet type (Unknown command code)
4	Illegal packet sub-type
5	Illegal or inconsistent application data (Illegal parameter value)
16	Illegal command
17	Table full
18	Element already in table
19	Element not in table
20	Bus operation failed
21	Command not on schedule
22	No storage selection definition
23	No read access
24	No write access
25	Access outside memory area
26	Unknown memory access error
27	Checksums differ
28	Unsupported activity
29	Buffer full
30	Excessive Memory Data Length

Code	Interpretation
31	Unsupported Checksum Type
32	Report Generation Failure

**Table 5.2-2: Telecommand Verification Failure Code Values**

It is an OBOSS-III requirement that failure codes in the range from 0 to 15 shall be reserved for error codes defined in the standard and that error codes from 16 to 127 shall be reserved for error codes that are applicable for all application processes. Error codes from 128 and up may thus be used for error codes specific to designated application processes.

Telecommand verification packets are routed to the source of the verified telecommand.

### 5.3 Device Command Distribution

The supported service sub-types for the Device Command Distribution service are marked in Table 5.3-1.

Service Sub-Type	Name	Supported	Comments
1	Distribute On-Off Commands	✓	
2	Distribute Register Load Commands	✓	Register Data field type shall be unsigned integer.
3	Distribute CPDU Commands		

**Table 5.3-1: Supported Device Command Distribution Service Sub-Types**

### 5.4 Housekeeping and Diagnostic Data Reporting

The supported service sub-types are marked in Table 5.4-1.

Service Sub-Type	Name	Supported	Comments
1	Define New Housekeeping Parameter Report	✓	
2	Define New Diagnostic Parameter Report	✓	
3	Clear Housekeeping Parameter Report Definitions	✓	
4	Clear Diagnostic Parameter Report Definitions	✓	
5	Enable Housekeeping Parameter Report Generation	✓	
6	Disable Housekeeping Parameter Report Generation	✓	
7	Enable Diagnostic Parameter Report Generation	✓	
8	Disable Diagnostic Parameter Report Generation	✓	

Service Sub-Type	Name	Supported	Comments
9	Report Housekeeping Parameter Report Definitions	✓	
10	Housekeeping Parameter Report Definitions Report	✓	
11	Report Diagnostic Parameter Report Definitions	✓	
12	Diagnostic Parameter Report Definitions Report	✓	
13	Report Housekeeping Parameter Sampling-Time Offsets		
14	Report Diagnostic Parameter Sampling-Time Offsets		
15	Housekeeping Parameter Sampling-Time Offsets Report		
16	Diagnostic Parameter Sampling-Time Offsets Report		
17	Select Periodic Housekeeping Parameter Report Generation Mode	✓	
18	Select Periodic Diagnostic Parameter Report Generation Mode	✓	
19	Select Filtered Housekeeping Parameter Report Generation Mode	✓	
20	Select Filtered Diagnostic Parameter Report Generation Mode	✓	
21	Report Unfiltered Housekeeping Parameters	✓	
22	Report Unfiltered Diagnostic Parameters	✓	
23	Unfiltered Housekeeping Parameters Report	✓	
24	Unfiltered Diagnostic Parameters Report	✓	
25	Housekeeping Parameter Report	✓	
26	Diagnostic Parameter Report	✓	

**Table 5.4-1: Supported Housekeeping and Diagnostic Data Reporting Service Sub-Types**

There are no predefined housekeeping parameter reports or diagnostic parameter reports. When new housekeeping parameter reports or diagnostic parameter reports are defined, parameter report generation are disabled. An Enable Parameter Report Generation request is required before new parameter reports are generated.

## 5.5 Event Reporting

The supported service sub-types are marked in Table 5.5-1.

Service Sub-Type	Name	Supported	Comments
1	Normal/Progress Report	✓	
2	Error/Anomaly Report – Low Severity	✓	
3	Error/Anomaly Report – Medium Severity	✓	
4	Error/Anomaly Report – High Severity	✓	

Service Sub-Type	Name	Supported	Comments
5	Enable Event Report Generation		
6	Disable Event Report Generation		

**Table 5.5-1: Supported Event Reporting Service Sub-Types**

Report ID's (RID's) shall be globally unique. RID's in the range from 0 to 127 shall be reserved for Event Reports that are applicable for all application processes. RID's listed in Table 5.5-2 are predefined in OBOSS-III.

RID	Service Sub-type	Parameters	Interpretation
0	1	None	System start up
1	4	Task ID (8,0) (n is Unsigned Integer (3,12)) Exception Name (8,0) (n is Unsigned Integer (3,12)) Exception Message (8,0) (n is Unsigned Integer (3,12)) Exception Information (8,0) (n is Unsigned Integer (3,12))	Task exception report
2	3	None	Packet conversion failure report
4	2	TM Packet Header ID (3,12) TM Packet Sequence Control (3,12)	Unsupported telemetry packet (Event-Action service)
5	4	None	Uninstantiated sending sub-service (Large Data Transfer service)
6	2	SDU Unit Type (2,8) SDU Packet Header (3,15)	SDU conversion failure report (Large Data Transfer service)
7	2	None	Current uplink aborted (Large Data Transfer service)

**Table 5.5-2: Event Reports Reserved by OBOSS-III**

## 5.6 Memory Management

The service sub-types supported for the Memory Management are marked in Table 5.6-1.

Service Sub-Type	Name	Supported	Comments
1	Load Memory using Base plus Offsets		
2	Load Memory using Absolute Addresses	✓	Checksum at data level is not supported. SAU is always 1 octet.
3	Dump Memory using Base plus Offset		
4	Memory Dump using Base plus Offset Report		
5	Dump Memory using Absolute Addresses	✓	SAU is always 1 octet.
6	Memory Dump using Absolute Addresses Report	✓	Checksum at data level is not supported. SAU is always 1 octet.
7	Check Memory using Base plus Offset		
8	Memory Check using Base plus Offset Report		
9	Check Memory using Absolute Addresses	✓	SAU is always 1 octet.
10	Memory Check using Absolute Addresses Report	✓	SAU is always 1 octet.

**Table 5.6-1: Supported Memory Management Service Sub-Types**

## 5.7 Function Management

OBOSS-III supports the service sub-type provided by the Function Management service as it can be seen in Table 5.7-1.

Service Sub-Type	Name	Supported	Comments
1	Perform Function	✓	N field shall always be present.

**Table 5.7-1: Supported Function Management Service Sub-Types**

## 5.8 On-Board Operations Scheduling

The service sub-types supported by OBOSS-III are marked in Table 5.8-1.

Service Sub-Type	Name	Supported	Comments
1	Enable Release of Telecommands	✓	
2	Disable Release of Telecommands	✓	
3	Reset Command Schedule	✓	
4	Insert Telecommands in Command Schedule	✓	Interlocking functionality is not supported <sup>1)</sup>

Service Sub-Type	Name	Supported	Comments
5	Delete Telecommands	✓	
6	Delete Telecommands over Time Period		
7	Time-Shift Selected Telecommands		
8	Time-Shift Telecommands over Time Period		
9	Report Subset of Command Schedule in Detailed Form	✓	
10	Detailed Schedule Report	✓	Interlocking functionality is not supported <sup>1)</sup>
11	Report Subset of Command Schedule in Detailed Form over Time Period	✓	
12	Report Subset of Command Schedule in Summary Form	✓	
13	Summary Schedule Report	✓	
14	Report Subset of Command Schedule in Summary Form over Time Period	✓	
15	Time-Shift All Telecommands		
16	Report Command Schedule in Detailed Form	✓	
17	Report Command Schedule in Summary Form	✓	
18	Report Status of Command Schedule		
19	Command Schedule Status report		

- 1) The following fields are not present: Interlock Set ID, Interlock Assessed ID, Assessment Type, and Execution Timeout.

**Table 5.8-1: Supported On-Board Operations Scheduling Service Sub-Types**

Note that jumps in CUC representation of onboard time are not detected.

The on-board schedule is empty at system start-up. Release of telecommands is disabled at schedule and sub-schedule level, but enabled at application process level.

## 5.9 On-Board Monitoring

The On-Board Monitoring service sub-types supported by OBOSS-III are marked in Table 5.9-1.

Service Sub-Type	Name	Supported	Comments
1	Enable Monitoring of Parameters	✓	
2	Disable Monitoring of Parameters	✓	
3	Change Maximum Reporting Delay		

Service Sub-Type	Name	Supported	Comments
4	Clear Monitoring List	✓	
5	Add Parameters to Monitoring List	✓	Delta check functionality is not supported (Delta #REP and NOD field are not present).
6	Delete Parameters from Monitoring List	✓	
7	Modify Parameter Checking Information	✓	Delta check functionality is not supported (NOD field is not present).
8	Report Current Monitoring List	✓	
9	Current Monitoring List Report	✓	Delta check functionality is not supported (Delta #REP and NOD field are not present).
10	Report Current Parameters Out-of-Limit List	✓	
11	Current Parameters Out-of-Limit List Report	✓	Delta check functionality is not supported.
12	Check Transition Report	✓	Delta check functionality is not supported.

**Table 5.9-1: Supported On-Board Monitoring Service Sub-Types**

Maximum Reporting Delay is a system parameter common to all application processes.

The monitoring list is per default empty and monitoring of parameters is disabled at service and parameter level.

## 5.10 Large Data Transfer

OBOSS-III supports all Large Data Transfer service sub-types as it can be seen in Table 5.10-1.

Service Sub-Type	Name	Supported	Comments
Data Downlink Operation (Sending)			
1	First Downlink Part Report	✓	Large Data Unit ID is not supported.
2	Intermediate Downlink Part Report	✓	Large Data Unit ID is not supported.
3	Last Downlink Part Report	✓	Large Data Unit ID is not supported.
4	Downlink Abort Report	✓	Large Data Unit ID is not supported.
5	Downlink Reception Acknowledgement	✓	Large Data Unit ID is not supported.
6	Repeat Parts	✓	Large Data Unit ID is not supported.

Service Sub-Type	Name	Supported	Comments
7	Repeated Part Report	✓	Large Data Unit ID is not supported.
8	Abort Downlink	✓	Large Data Unit ID is not supported. Reason codes are not included in abort requests.
Data Uplink Operation (Receiving)			
9	Accept First Uplink Part	✓	Large Data Unit ID is not supported.
10	Accept Intermediate Uplink Part	✓	Large Data Unit ID is not supported.
11	Accept Last Uplink Part	✓	Large Data Unit ID is not supported.
12	Accept Repeated Part	✓	Large Data Unit ID is not supported.
13	Abort Reception of Uplinked Data	✓	Large Data Unit ID is not supported. Reason codes are not included in abort requests.
14	Uplink Reception Acknowledgement Report	✓	Large Data Unit ID is not supported.
15	Unsuccessfully Received Parts Report	✓	Large Data Unit ID is not supported.
16	Reception Abort Report	✓	Large Data Unit ID is not supported.

**Table 5.10-1: Supported Large Data Transfer Service Sub-Types**

Large Data Transfer is limited to one Service Data Unit (SDU) at a time. Sliding-window functionality is not supported.

The reason codes listed in Table 5.10-2 are predefined for OBOSS-III implementations of the Large Data Transfer service sub-types 4 and 16.

Reason Code	Sending/Receiving	Interpretation
1	Sending	Wrong sequence number in acknowledge
2	Sending	Reception acknowledge timeout
3	Sending	Part stream allocation failed
4	Sending	Part packet allocation failed
5	Sending	Part packet deposit failed
6	Sending	Illegal part from sender state
7	Sending	Sender logic error
8	Receiving	Timeout waiting for part
9	Receiving	Illegal command waiting for part
10	Receiving	Illegal part sequence number
11	Receiving	Repeated part erroneous

Reason Code	Sending/Receiving	Interpretation
12	Receiving	Receiver logic error

**Table 5.10-2: OBOSS-III Reason Codes**

## 5.11 On-Board Storage and Retrieval

The On-Board Storage and Retrieval service sub-types supported by OBOSS-III are marked in Table 5.11-1.

Service Sub-Type	Name	Supported	Comments
Packet Selection Sub-Service			
1	Enable Storage in Packet Stores	✓	
2	Disable Storage in Packet Stores	✓	
3	Add packets to Storage Selection Definition	✓	Packet Selection Sub-Service shall be provided by the originating application process <sup>1)</sup>
4	Remove Packets from Storage Selection Definition	✓	Packet Selection Sub-Service shall be provided by the originating application process <sup>1)</sup>
5	Report Storage Selection Definition	✓	
6	Storage Selection Definition Report	✓	Packet Selection Sub-Service shall be provided by the originating application process <sup>1)</sup>
Storage and Retrieval Sub-Service			
7	Downlink Packet Store Contents for Packet Range	✓	
8	Packet Store Contents Report	✓	
9	Downlink Packet Store Contents for Time Period	✓	
10	Delete Packet Store Contents up to Specified Packets	✓	
11	Delete Packet Store Contents up to Specified Storage Time	✓	
12	Report Catalogues for Selected Packet Stores		
13	Packet Store Catalogue Report		

1) N1 and Application ID field are not present.

**Table 5.11-1: Supported On-Board Storage and Retrieval Service Sub-Types**

Storage in packet stores is per default disabled. After a storage selection definition has been defined an Enable Storage in Packet Stores request must be issued.

A Delete Packet Store Contents up to Specified Packets with the Deletion Set field set to zero will delete the entire packet store regardless of whether it has been downlinked or not.

If current storage selection definition is empty, and a Storage Selection Definition Report is requested then a verification failure is returned in response to the telecommand. This allows ground to distinguish among all packet types being selected and no packet types being selected.

## 5.12 Event-Action

OBOSS-II supports all Event-Action service sub-types as it can be seen in Table 5.12-1.

Service Sub-Type	Name	Supported	Comments
1	Add Events to the Detection List	✓	
2	Delete Events from the Detection List	✓	
3	Clear the Event Detection List	✓	
4	Enable Actions	✓	
5	Disable Actions	✓	
6	Report the Event Detection List	✓	
7	Event Detection List Report	✓	

**Table 5.12-1: Supported Event-Action Service Sub-Types**

There is no default event-action relations defined. The detection list is empty at system start-up. When new events are added to the event detection list, actions are initially disabled.

## **6 Scheduling**

### **6.1 Scheduling Mechanism**

The OBOSS-III baseline is based on the standard Ada 95 scheduling mechanism, being *priority-based, preemptive scheduling*. The OBOSS-III implementation adheres to the Ada-95 subset referred to as the 'Ravenscar Profile'. This implies that priority assignments shall adhere to the *Priority ceiling protocol* and that the scheduling policy shall be *first-in, first-out (FIFO) within priorities*.

---

## 7 Installation

### 7.1 Installation Instructions

The current section will outline the required steps to be performed when building an OBOSS demonstrator being the sample prototype distributed along with the OBOSS-III baseline.

- Extract the source files into a given directory. In the remaining instructions, this directory will be referred to as “\$sources”.
- Enter the subdirectory \$sources/Demonstrator.
- Set up the GNAT search path to cover all the subdirectories residing under \$sources. This search path is a colon-separated list held in the environment variable ‘ADA\_INCLUDE\_PATH’:

```
setenv ADA_INCLUDE_PATH
```

```
“$sources/Demonstrator:$sources/Demonstrator/Data_Handling_System:$sources/Demonstrator/Low_Level_Drivers:$sources/Demonstrator/Onboard_Storage:$sources/Demonstrator/Payload:$sources/Demonstrator/Power_Conditioning_System:$sources/Stubs:$sources/PUS_Services:$sources/PUS_Services/Device_Command_Distribution:$sources/PUS_Services/Event_Action:$sources/PUS_Services/Event_Reporting:$sources/PUS_Services/Event_Scheduler:$sources/PUS_Services/Function_Management:$sources/PUS_Services/HK_Collector:$sources/PUS_Services/Large_Data_Transfer:$sources/PUS_Services/Memory_Management:$sources/PUS_Services/Monitor:$sources/PUS_Services/On_Board_Scheduler:$sources/PUS_Services/Storage_And_Retrieval:$sources/PUS_Services/TC_Verification:$sources/PUS_Services/Task_Management:$sources/CDH_Structure:$sources/CDH_Structure/CDH_Structure_Initialiser:$sources/CDH_Structure/Common_IF:$sources/CDH_Structure/Dynamic_Application_Process_Descrs:$sources/CDH_Structure/Packet_Router:$sources/CDH_Structure/Up_Down_Link_Bus:$sources/Basic_Services:$sources/Basic_Services/Basic_Services_Initialiser:$sources/Basic_Services/Containers:$sources/Basic_Services/Control_Structures:$sources/Basic_Services/External_PUS:$sources/Basic_Services/Internal_PUS:$sources/Basic_Services/Low_Level_Stuff:$sources/Basic_Services/Mission_Parameters:$sources/Basic_Services/Parameter_Structure_Descriptions:$sources/Basic_Services/Platform_Parameters:$sources/Basic_Services/Resource_Manager:$sources/Basic_Services/Source_Data”
```

- Build the demonstrator main program ‘demonstrator.adb’ using the command ‘sparc-ork-gnatmake’:

```
sparc-ork-gnatmake -g -j6 demonstrator -cargs -c -g -gnato -gnata -fstack-check -bargs -larg -k -mcpu=cypress -specs ork_specs
```