

**QB50**



**FP7-284427**

**WP 250: Satellite Control  
Software**

**Deliverable D250.1:  
SCS description and  
interface control  
document**

*Update to Year-1 D250.1*

*Issue 4, Revision 1*

Prepared by:  
Stéphane Billeter



Checked by:  
Muriel Richard  
Yann Voumard

Approved by:

Swiss Space Center EPFL  
Lausanne  
Switzerland





## RECORD OF REVISIONS

ISS/REV	Date	Modifications	Created/modified by
1/1	22 January 2013	First issue. First release to QB50 project and REA	M. Richard
2/0	06 November 2013	Second issue. Version sent to CubeSat teams.	S. Billeter, M. Richard
3/0	10 May 2014	Third issue. Modified large data transfer (Service 13) description. Updated: “QB50-EPFL-SSC-ICD-TMTC_PD-3-0”, EPFL report, release 30 April 2014. New chapter/appendix about flight software. See “QB50-EPFL-SSC-SCS-ICD-FSW-1-0”. New chapter on CubeSat test with SCS. New chapter about next step for teams. Removed appendix on error correction.	S. Billeter, M. Richard, L. Masson, Y. Voumard.
3/1	17 July 2014	Added remark about service 3 for the WOD	Y. Voumard
4/0	31 July 2014	Fourth issue. First release of software for QB50 teams. Update according to service 13 and new client for uploading large data on-board. Specific tailoring of service 13. - No additional field for extended packet - No window used.	S. Billeter



---

4/1	23 March 2015	Update reference to updated documents	Y. Voumard
-----	---------------	---------------------------------------	------------



<b>RECORD OF REVISIONS</b> .....	<b>2</b>
<b>TERMS AND ABBREVIATIONS</b> .....	<b>6</b>
<b>REFERENCES</b> .....	<b>8</b>
<b>INTRODUCTION</b> .....	<b>9</b>
<b>SUMMARY</b> .....	<b>9</b>
<b>1 SCS DESCRIPTION</b> .....	<b>11</b>
1.1 SOFTWARE ELEMENTS .....	12
1.1.1 <i>Ground Station Manager [Orange]</i> .....	12
1.1.2 <i>TMTC Front End [Red]</i> .....	12
1.1.3 <i>Mission Control System (MCS) [Blue]</i> .....	12
1.1.4 <i>User Interfaces [Green]</i> .....	13
1.2 EXTERNAL INTERFACES .....	16
1.2.1 <i>Input – Science scripts</i> .....	16
1.2.2 <i>Output – Mission Server</i> .....	16
1.2.3 <i>Public Mission Data Client</i> .....	17
1.3 SOFTWARE TECHNOLOGY .....	17
1.3.1 <i>GS Manager</i> .....	17
1.3.2 <i>TMTC Front End</i> .....	17
1.3.3 <i>MCS</i> .....	17
1.3.4 <i>User Interfaces</i> .....	17
1.4 EXTENSIBILITY .....	17
1.4.1 <i>MCS</i> .....	19
1.4.2 <i>MDC</i> .....	19
1.4.3 <i>Example</i> .....	19
<b>2 TYPES OF DATA, DATA FLOW AND PROTOCOLS</b> .....	<b>20</b>
2.1 TYPES OF DATA .....	20
2.1.1 <i>Payload Data (PD)</i> .....	20
2.1.2 <i>House-Keeping Data (HSK)</i> .....	20
2.1.3 <i>Science Data (SD)</i> .....	20
2.1.4 <i>Whole Orbit Data (WOD)</i> .....	20
2.2 REQUIREMENTS RELATED TO THE USE OF RADIO AMATEUR BANDS .....	21
2.3 DATA CODING BASIC PROCESS (FOR BACKGROUND).....	21
2.4 AX.25 TRANSFER FORMAT.....	22
2.4.1 <i>Unnumbered Information Frame</i> .....	23
2.4.2 <i>Information Field specification</i> .....	23
2.4.3 <i>Overhead efficiency</i> .....	23
2.5 PACKET STRUCTURE RECOMMENDATION .....	24
2.5.1 <i>Telecommand Packet Data Field Structure</i> .....	24
2.5.2 <i>Telemetry Packet Data Field Structure</i> .....	24
2.6 ERROR MANAGEMENT.....	24
<b>3 SERVICES</b> .....	<b>25</b>
3.1 STANDARD SERVICES .....	25
3.1.1 <i>Telecommand verification - Type 1</i> .....	25
3.1.2 <i>Housekeeping data reporting - Type 3</i> .....	25
3.1.3 <i>Function management - Type 8</i> .....	26
3.1.4 <i>On-board operations scheduling - Type 11</i> .....	26
3.1.5 <i>Large data transfer - Type 13 (applicable to the uplink)</i> .....	26
3.1.6 <i>On-board storage and retrieval - Type 15</i> .....	27
3.2 QB50 MISSION-SPECIFIC SERVICES.....	27
3.2.1 <i>INMS Data management - Type 128</i> .....	27
3.2.2 <i>FIPEX Data management - Type 129</i> .....	27
3.2.3 <i>MNLP Data management - Type 130</i> .....	27



---

<b>4</b>	<b>FLIGHT SW IMPLEMENTATION EXAMPLE .....</b>	<b>28</b>
<b>5</b>	<b>USING THE SCS FOR YOUR CUBESAT TESTS .....</b>	<b>29</b>
5.1	GENERAL APPROACH.....	29
5.2	HARDWARE COMPONENTS .....	30
5.3	SOFTWARE COMPONENTS.....	31
<b>6</b>	<b>SCS AS A NETWORK.....</b>	<b>32</b>
<b>7</b>	<b>CONCLUSIONS AND PREPARATION STEPS FOR CUBESAT TEAMS .....</b>	<b>34</b>



## TERMS AND ABBREVIATIONS

Ack	Acknowledgement
ADCS	Attitude Determination and Control System
APID	Application Process ID
bpp	Bits Per Pixel
CCSDS	Consultative Committee for Space Data Systems
COM	Communication [subsystem]
CRC	Cyclic Redundancy Code
ECSS-PUS	European Cooperation on Space Standardization – Packet Utilisation Services
EPS	Electrical Power Subsystem
ESA	European Space Agency
GS Manager	Ground Station Network and Frequency Allocation Working Group
GS Manager	Ground Station Manager
HSK	House-Keeping Data
ITU	International Telecommunication Union
MCS	Mission Control System
OBT	On-Board Time
PC	Parameter Code
PFC	Parameter Format Code
PL	PayLoad [subsystem]
PSS	Procedures, Specifications, Standards
PTC	Parameter Type Code
PUS	Packet Utilization Standard
Sc-HK	Science related Housekeeping data
SCS	Satellite Control Software
SD	Science Data
SID	Structure Identification
SU	Science Unit
TMTC Front End	TeleMetry-TeleCommand Front End
UI	Unnumbered Information Frames
UTC	Universal Time Coordinated
VC	Virtual Channel
VKI	Von Karman Institute



---

WOD

Whole Orbit Data



## REFERENCES

- [R1] “QB50\_VKI\_D510.1\_Ground Segment Definition - Issue 2”, VKI report, Nov. 2013.
- [R2] “QB50-EPFL-SSC-SCS-ICD-TMTC\_PD-4-1”, EPFL report, release 23 March 2015.
- [R3] “QB50 INMS Science Unit Interface Control Document”, Issue 7, MSSSL, 4 Dec. 2013
- [R4] “Whole Orbit Data Packet Format D510.1”, Rev. 4, VKI, 23 Oct. 2014
- [R5] “QB50 Ground Station Network and Frequency Allocation Working Group – Report Number 1”, Issue 1, Revision 0, 29/07/2011.
- [R6] <https://www.qb50.eu/index.php/requirements-and-reviews>
- [R7] ECSS-E-70-41A Packet Utilization Standard.
- [R8] “QB50-EPFL-SSC-SCS-ICD-AX.25-TFF-3-1”, EPFL report, release 23 March 2015.
- [R9] “QB50-EPFL-SSC-SCS-ICD-EGSE-RI-2-0”, EPFL report, release 06 Nov. 2013.
- [R10] “QB50-EPFL-SSC-SCS-ICD-MCS-E-2-0”, EPFL report, release 06 Nov. 2013.
- [R11] “QB50-EPFL-SSC-SCS-ICD-MDC-E-2-0”, EPFL report, release 06 Nov. 2013.
- [R12] “QB50-EPFL-SSC-SCS-ICD-FSW-1-0”, EPFL report, release 30 Apr. 2014.
- [R13] QB50-EPFL-SSC-SCS-ICD-Function Definitions-1-0
- [R14] QB50-EPFL-SSC-SCS-ICD-Housekeeping Parameter Definitions-1-0
- [R15] QB50-EPFL-SSC-1-5 - Universities with SCS





## INTRODUCTION

The QB50 Satellite Control Software (SCS) developed by EPFL is proposed by the QB50 project as support to CubeSat teams. The SCS was originally developed for the SwissCube project. It has been successfully supporting operations for this mission over the last 4 years, thus validating its functionalities.

New and additional functionalities related to the QB50 science are being developed and will be tested to comply with the requirements of the QB50 project. They will be included in the SCS software package that teams will be able to download.

This document describes the SCS proposed by the EPFL as an option to the ground segment software of QB50 teams. It discusses the architecture of SCS and the flow of information that is transmitted between the interfaces, the data protocol tailored from the ECSS, the extensibility to new interfaces, and the option of creating a network using the SCS. This version also includes a brief discussion on possible implementations of the flight software to communicate with the SCS, and insertion of the SCS into functional testing of the CubeSats.

## SUMMARY

This document is also supported by 6 complementary documents that provide detailed descriptions. Table 1 lists the content of those supporting document.

### Central node

Details of the SCS network are given in the document “**QB50-EPFL-SSC-SCS-ICD-EGSE-RI-2-0**”. This describes the SCS elements and how each element of the SCS internal network is connected with each other. It also shows the detail of the communication between each node of the network. With this document, CubeSat teams are able to create specific applications in addition to the ones provided.

### Communication protocol

The protocols of communication between the ground segment and the CubeSat are detailed in two different documents:

- “**QB50-EPFL-SSC-SCS-ICD-AX.25-TFF-3-1**”
- “**QB50-EPFL-SSC-SCS-ICD-TMTC\_PD-4-1**”

The first document shows how the AX.25 frame protocol<sup>1</sup> of the amateur radio is used to transmit the data through the radio frequency link and how to build the frame to communicate with the CubeSat. The second document is the definition of packets format (lower level than the frames). It also describes all on-board services that the packets are related to.

---

<sup>1</sup> AX.25 is a requirement of the QB50 project. If you wish to use a different protocol, a waiver is required. As long as you keep CCSDS as packet format, only a small component (TM/TC Front End) of the SCS needs to be changed for adapting to your protocol. If you do not intend to use CCSDS, the provided SCS will be of very little help for your CubeSat.



## Extensibility

Two documents describe the extensibility of the SCS software, such that CubeSat teams can create their own clients on top of the ones provided by default:

- “QB50-EPFL-SSC-SCS-ICD-MCS-E-2-0”
- “QB50-EPFL-SSC-SCS-ICD-MDC-E-2-0”

## Flight software

This document presents a possible implementation of the flight software compatible with the ECSS standard and the SCS. It contains explanations on packages that are structuring the software, flowcharts of main functionalities and detailed characteristics of most important functions:

- “QB50-EPFL-SSC-SCS-ICD-FSW-1-0”

**Table 1: Supporting documents to D250.1**

QB50-EPFL-SSC-SCS-ICD-EGSE-RI-2-0	Description of the internal SCS network and communication protocol.
QB50-EPFL-SSC-SCS-ICD-TMTC_PD-4-1	Description of the packets protocol
QB50-EPFL-SSC-SCS-ICD-AX.25-TFF-3-1	Tailored description of the amateur radio protocol
QB50-EPFL-SSC-SCS-ICD-MCS-E-2-0	Extensibility of the Mission Control System
QB50-EPFL-SSC-SCS-ICD-MDC-E-2-0	Extensibility of the Mission Data Client
QB50-EPFL-SSC-SCS-ICD-FSW -1-0	Example of flight software implementation

# 1 SCS DESCRIPTION

The Satellite Control Software (SCS) is a part of the overall ground segment. The SCS manages the satellite data on the ground, uploads commands, download telemetries and science data. It interfaces with the user (CubeSat operations team), and provides decoded satellite data to the radio-amateurs. The SCS is typically located at the CubeSat university (server), although user clients allow control of the satellite from any internet-connected laptop.

The QB50 ground segment architecture is shown in its “SCS-centric way” in Figure 1. Complementary information of the overall ground segment information can be found in [R1]. The Satellite Control System is able to communicate with many different ground stations to control a CubeSat. Figure 1 shows a logical view of the software and in particular, the provided SCS is highlighted.

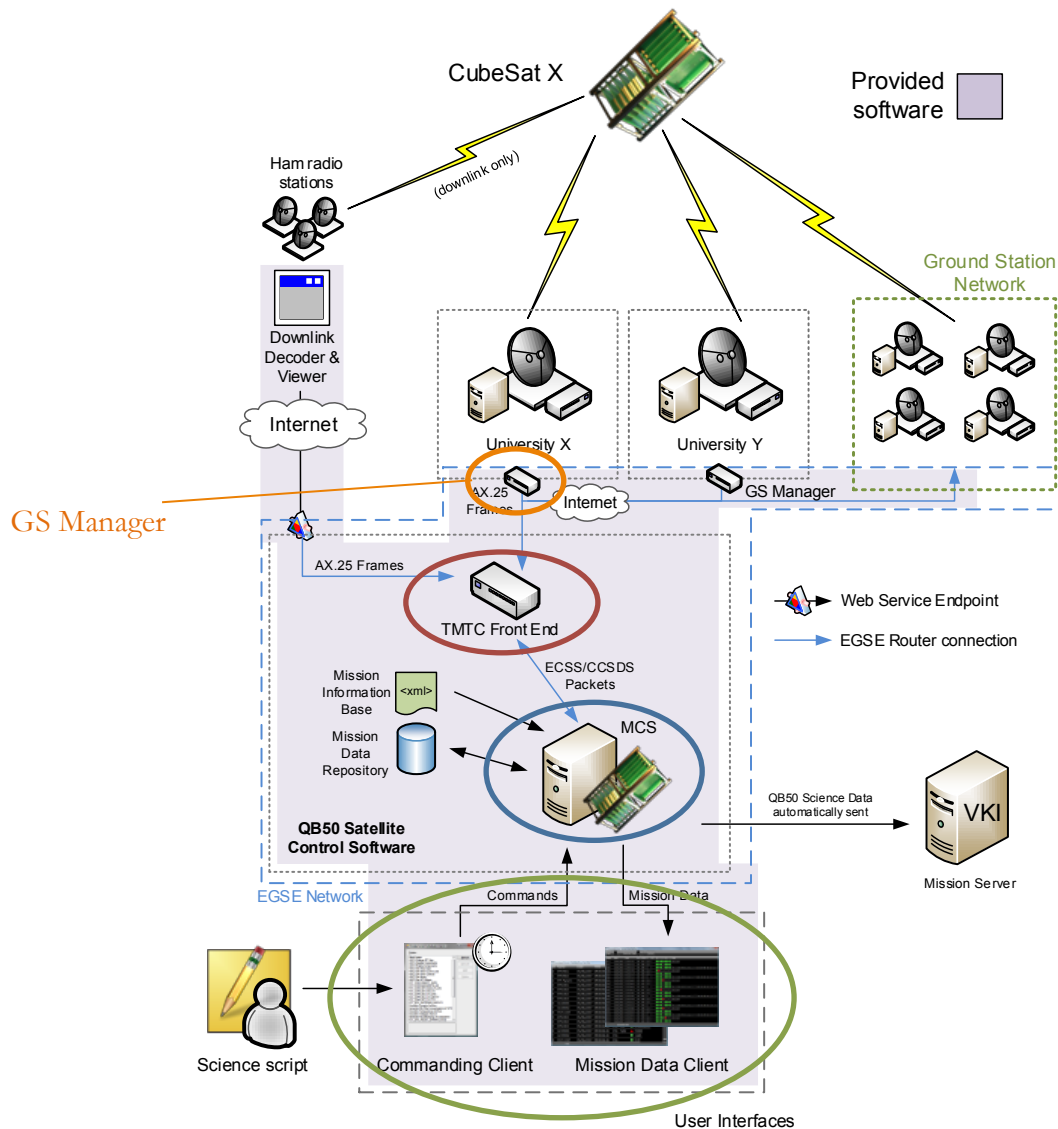


Figure 1: Overview of Satellite Control System Software.

The SCS architecture and data flow process uses a tailored version of the “ECSS-E-70-41A Packet Utilization Standard”. The proposed tailoring [R2] of this ECSS standard aimed at removing unnecessary complexity brought by the standard (designed for complex space mission), however keeping a clear structure for the management of the data. The provided supporting documents will allow the CubeSat teams to design their own tailoring of the standard if deemed necessary for them.

The next sections will describe the elements of the SCS and the external interfaces of the SCS.

## 1.1 Software elements

The provided software is composed of several elements.

### 1.1.1 Ground Station Manager [Orange]

The ground station manager is the entry gate of the satellite data into the SCS internal network. It is physically installed at the ground station. It allows the ground station to be a part of the SCS internal network. The ground station has a unique identifier on the SCS internal network to communicate with the other parts of the internal network, especially the TMTC Front End where the ground station has to send its raw frame to be processed.

There is no data processing at this level and the software only transmits the whole frame it has received to the TMTC Front End for further processing.

### 1.1.2 TMTC Front End [Red]

The TMTC Front End is the layer between ground stations and the Mission Control System. This layer contains the handling of all the features of the frame-based protocol used on the space link. It provides the following functionalities:

- Archiving of raw telemetry
- Replay of data from the raw archive (e.g. to reprocess data afterwards)
- Reassembling of telemetry from multiple AX.25 frames that may be received from multiple Ground Stations in order to maximise reconstruction success
- Encapsulation of telecommands into AX.25 frames
- Detection of on-board arrival telecommands using acknowledgements
- Configurable automatic retrial of telecommands transmission

As defined above, the TMTC Front End manage all AX.25 frames and every frame is archived to back up all raw data.

### 1.1.3 Mission Control System (MCS) [Blue]

The Mission Control System is the heart of ground segment. It performs all the packets data processing such as:

- Extracting of housekeeping, payload data, etc... out of the packets
- Storing in a database all raw and processed data for archiving and/or post-processing
- Monitoring of the health of the CubeSat and ground segment
- Generation of telecommand packets and tracking of their execution progress.

## 1.1.4 User Interfaces [Green]

The user interfaces allow operators to interact with the MCS.

### 1.1.4.1 Commanding Client

The commanding client is an application that provides a basic mean to send telecommands. Its purpose is to select which command has to be sent to the CubeSat (as shown in Figure 2).

The commanding client allows the selection of the ground station through which the uplink commands shall be sent. It allows the burst repetition of a command via a TMTC Front End feature.

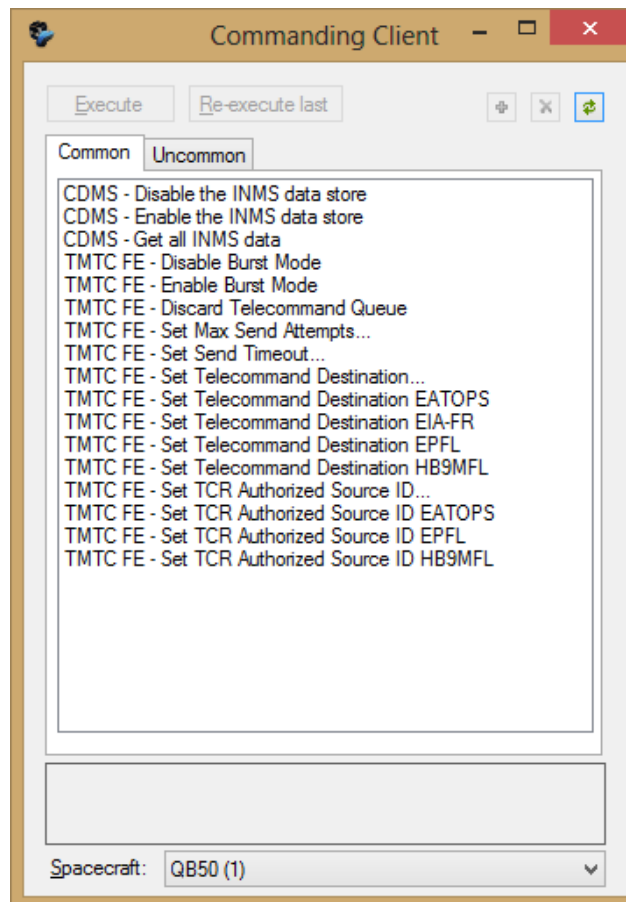


Figure 2 : Commanding Client (example).

### 1.1.4.2 Upload Data Client

The upload data client is an application that provides a way to uplink the content of a file through the service 13 (as shown in Figure 3).

Its purpose is to display monitoring information regarding the progress of the upload. The client gives the user options to upload data, like the file to be uploaded, the amount of data per telecommand, the data storage as destination on board and the selected cubesat.

It also provides the ability to resend a part separately.

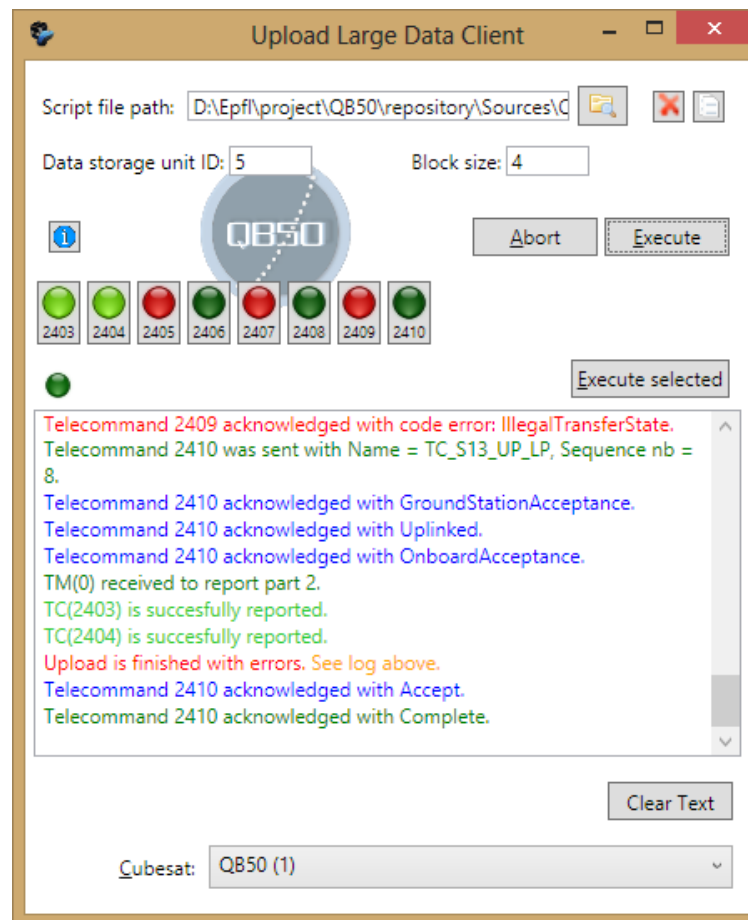


Figure 3: Upload Data Client

### 1.1.4.3 Mission Data Client

This application is the main downlink interface for the user. It provides by default visualization for all the engineering data and science data. It also allows the inclusion of mission-specific views.

Such features are available for both live monitoring and offline analysis:

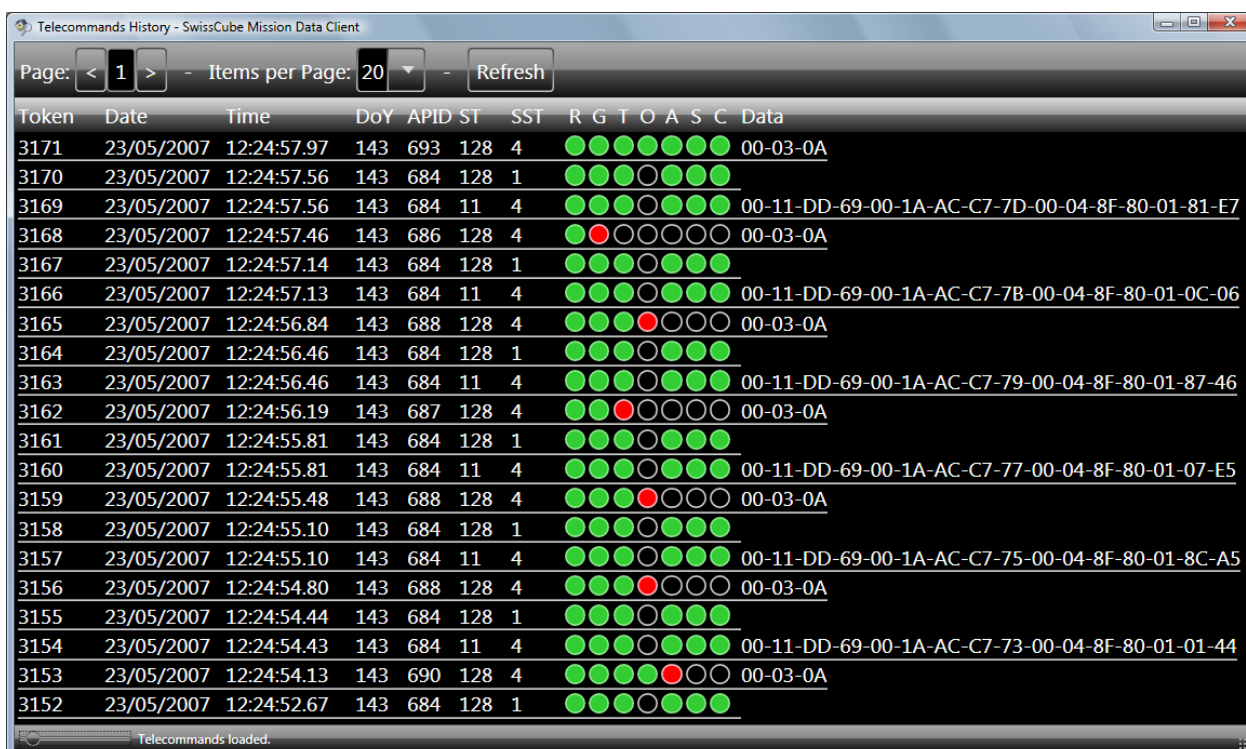
- Telecommands detail and their current acknowledgement status
- Telemetry listing (incoming decoded packets)
- Housekeeping values for each defined structures and parameters
- Payload data
- Visualisation of archived data.

An example is shown in Figure 4 and 4.



Id	Name	Date	Time	DoY	Unit	Value
0	EPSV001	05/06/2007	09:08:46.46	156	mV	19924
1	EPSV002	05/06/2007	09:08:46.46	156	mV	24046
2	EPS Panel 1	05/06/2007	09:08:46.46	156	mV	56
3	EPSC001	05/06/2007	09:08:46.46	156	mA	56187
4	EPSC002	05/06/2007	09:08:46.46	156	mA	20291
4	EPSC002	05/06/2007	09:08:46.46	156	mA	20291
5	EPSC003	05/06/2007	09:08:46.46	156	mA	1710436914
6	ACDS001	05/06/2007	09:08:46.46	156	TrM	72263
7	ACDS002	05/06/2007	09:08:46.46	156		_k&9-
8	PAYLSTAT	05/06/2007	09:08:46.46	156		!139
9	EPSC002	05/06/2007	09:08:46.46	156	mA	20291
4	EPSC002	05/06/2007	09:08:46.46	156	mA	20291
5	EPSC003	05/06/2007	09:08:46.46	156	mA	1710436914
6	ACDS001	05/06/2007	09:08:46.46	156	TrM	72263
7	ACDS002	05/06/2007	09:08:46.46	156		_k&9-
8	PAYLSTAT	05/06/2007	09:08:46.46	156		!139

Figure 4 : Example of the MDC Housekeeping display.



Token	Date	Time	DoY	APID	ST	SST	R	G	T	O	A	S	C	Data
3171	23/05/2007	12:24:57.97	143	693	128	4	●	●	●	●	●	●	●	00-03-0A
3170	23/05/2007	12:24:57.56	143	684	128	1	●	●	●	○	●	●	●	
3169	23/05/2007	12:24:57.56	143	684	11	4	●	●	●	○	●	●	●	00-11-DD-69-00-1A-AC-C7-7D-00-04-8F-80-01-81-E7
3168	23/05/2007	12:24:57.46	143	686	128	4	●	●	○	○	○	○	○	00-03-0A
3167	23/05/2007	12:24:57.14	143	684	128	1	●	●	●	○	●	●	●	
3166	23/05/2007	12:24:57.13	143	684	11	4	●	●	●	●	●	●	●	00-11-DD-69-00-1A-AC-C7-7B-00-04-8F-80-01-0C-06
3165	23/05/2007	12:24:56.84	143	688	128	4	●	●	●	●	○	○	○	00-03-0A
3164	23/05/2007	12:24:56.46	143	684	128	1	●	●	●	○	●	●	●	
3163	23/05/2007	12:24:56.46	143	684	11	4	●	●	●	○	●	●	●	00-11-DD-69-00-1A-AC-C7-79-00-04-8F-80-01-87-46
3162	23/05/2007	12:24:56.19	143	687	128	4	●	●	●	○	○	○	○	00-03-0A
3161	23/05/2007	12:24:55.81	143	684	128	1	●	●	●	○	●	●	●	
3160	23/05/2007	12:24:55.81	143	684	11	4	●	●	●	○	●	●	●	00-11-DD-69-00-1A-AC-C7-77-00-04-8F-80-01-07-E5
3159	23/05/2007	12:24:55.48	143	688	128	4	●	●	●	●	○	○	○	00-03-0A
3158	23/05/2007	12:24:55.10	143	684	128	1	●	●	●	○	●	●	●	
3157	23/05/2007	12:24:55.10	143	684	11	4	●	●	●	○	●	●	●	00-11-DD-69-00-1A-AC-C7-75-00-04-8F-80-01-8C-A5
3156	23/05/2007	12:24:54.80	143	688	128	4	●	●	●	●	○	○	○	00-03-0A
3155	23/05/2007	12:24:54.44	143	684	128	1	●	●	●	○	●	●	●	
3154	23/05/2007	12:24:54.43	143	684	11	4	●	●	●	●	●	●	●	00-11-DD-69-00-1A-AC-C7-73-00-04-8F-80-01-01-44
3153	23/05/2007	12:24:54.13	143	690	128	4	●	●	●	●	○	○	○	00-03-0A
3152	23/05/2007	12:24:52.67	143	684	128	1	●	●	●	○	●	●	●	

Figure 5 : Example of the MDC Telecommands display.





## 1.2 External interfaces

### 1.2.1 Input – Science scripts

As a specificity of the project, science commands will be sent as scripts to the science unit. This functionality is being implemented within the SCS.

The science scripts will be sent (currently via email) to each individual team during the mission operations. It is the responsibility of the team to uplink the scripts to their CubeSat and to verify correct loading and execution of the scripts. The ground part of this activity will be provided in the SCS. To do this, a standard service (Service 13) of the ECSS will be implemented. This service provides methods for uploading and downloading large unit of data such as the QB50 science scripts. The extension of the SCS will support only the uplink direction.

The science script files will be split into subcommands (see Figure 6), which are manipulated as an array of bytes. The software will not check integrity of the content of the command scripts, so the operator should control the validity of the script before sending and upon reception on-board.

@19:00:00 : OBC_ON
@19:02:00 : INMS_STIM 0xF0
@19:03:00 : INMS_HVARM
@19:04:00 : INMS_HVEN
@19:06:00 : INMS_HK 60
@NOW : INMS_CAL 5 64 28 ON ON ON
@NOW : INMS_LDP 0x09, 0x04,0x41,0x80,0x00, 0xFE,0x00,0x00,0x02,0xFF
@19:15:00 : INMS_HC 100 64 24 ON ON ON

Figure 6 : Example of script [R3]

### 1.2.2 Output – Mission Server

When a packet is defined as a QB50 related science data, it will be sent to a central mission server at VKI. Currently, a web-service is proposed to upload a file which would combine all QB50 data. The interface allows other teams not using the SCS to upload files manually as shown in Figure 6. This interface is described in [R1].

Username:	<input type="text" value="loginid (readonly)"/>
File to upload (max. filesize 10MB):	<input type="text"/> <input type="button" value="Browse..."/>
File type:	<input type="text" value="Science Data (SD)"/>
CubeSat ID:	<input type="text" value="select CubeSat"/>
Contact time UTC (YYYY-MM-DD HH:MM):	<input type="text" value="2000-01-01 00:00"/>
Checksum MD5 (optional):	<input type="text"/>
	<input type="button" value="Upload"/>

Figure 7: Web-interface for uploading science and related housekeeping to VKI.





### 1.2.3 Public Mission Data Client

Non-commercial missions such as university CubeSats usually expose parts of the mission data to the community. It can just be the health of the spacecraft or include payload data as well. Such client is typically in the form of a Web site, and can be easily integrated as a client to the SCS. The software implementation is discussed in the Extensibility chapter.

## 1.3 Software Technology

### 1.3.1 GS Manager

Application Type:                      Service Application  
Operating System:                      Microsoft Windows Server 2003 SP1, Windows XP SP2 or higher  
Development Language:                C# with .NET Framework 4.5

### 1.3.2 TMTC Front End

Application Type:                      Server Application  
Operating System:                      Microsoft Windows Server 2003 SP1, Windows XP SP2 or higher  
Development Language:                C# with .NET Framework 4.5  
Database Server:                        Microsoft SQL Server 2012

### 1.3.3 MCS

Application Type:                      Server Application  
Operating System:                      Microsoft Windows Server 2003 SP1, Windows XP SP2 or higher  
Development Language:                C# with .NET Framework 4.5  
Database Server:                        Microsoft SQL Server 2012

### 1.3.4 User Interfaces

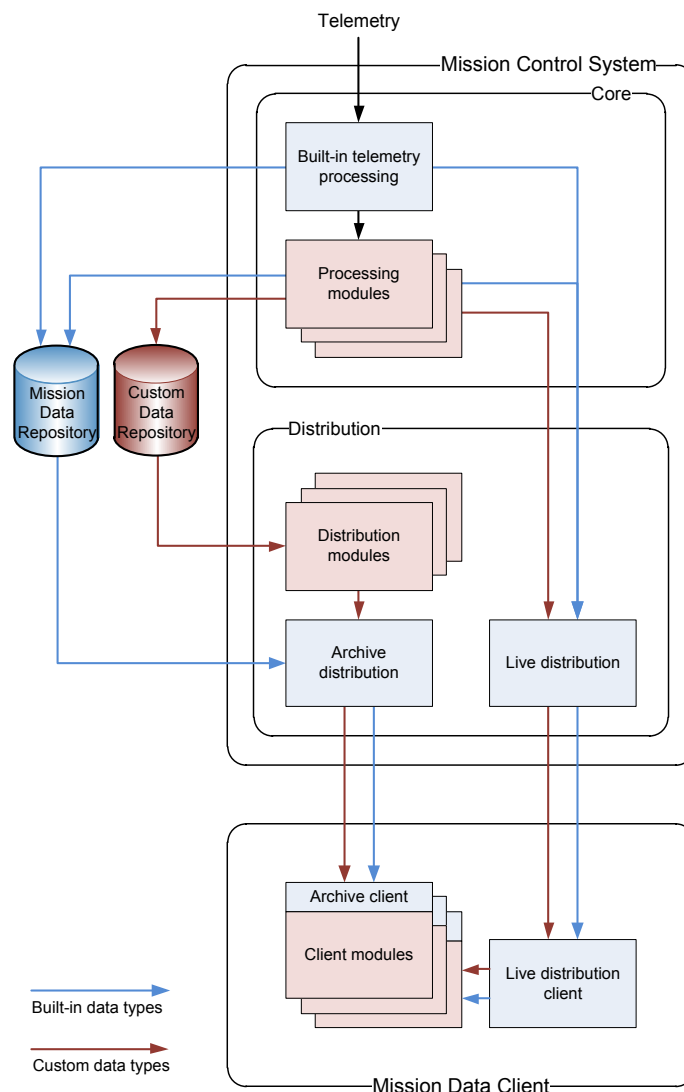
Application Type:                      Desktop Application  
Operating System:                      Microsoft Windows Server 2003 SP1, Windows XP SP2 or higher  
Development Language:                C# with .NET Framework 4.5  
Graphical Interface Library:        Windows Presentation Foundation (WPF)

## 1.4 Extensibility

The SCS offers the possibility to each QB50 team to develop their own module to monitor the CubeSat specific payload data and housekeeping. To customize the SCS the team can create new modules in the following SCS submodules (see Figure 7):

- The user interface specific for payload data: the specific UI for payload will be an additional module of the Mission Data Client (MDC)
- The type of data of the payload: the data of payload should be specified. To do so, a module of the
- Modules in the Mission Control System (MCS) have to be implemented to define, process and distribute the data;
- A module in the Mission Data Client (MDC) should be implemented to provide a user interface to visualize the

Technically, extensibility for 2 aspects above, MDC and MCS, is exposed for both through 2 interfaces. Details on the extensibility implementation capability are provided in [R9, R10, R11] as appendix documents.



**Figure 7: Data flow of the built-in and custom data.**

In Figure 7, the flow which comes in and goes out from Custom Data Repository is customized by extension in the SCS. A Processing module would be a module of the MCS and a Distribution module and a Client module would be a module of the MDC.

### 1.4.1 MCS

The extensibility of the MCS is exposed by 2 interfaces – IProcessingModule and IDistributionModule. IProcessingModule shows how the module should process the Telemetry and IDistributionModule shows how to compile the data in a specific type - CustomReportingData.

### 1.4.2 MDC

The extensibility of the MDC is exposed by 2 interfaces – IActionModule and IArchiveDistribution. IActionModule shows the entry point of the module and IArchiveDistribution is already implemented by ArchiveDistributionClient which can access all mission data.

### 1.4.3 Example

On Figure 8, each window is identified as a module of the SCS. The 3 windows on the left show the latest overall housekeeping received (left), the status of the Telecommands (middle up) and the packets received in real time (middle bottom). These are not integrated as modules and are provided by default with the SCS. The 2 right windows are CubeSat specific and are part of the extensibility modules.

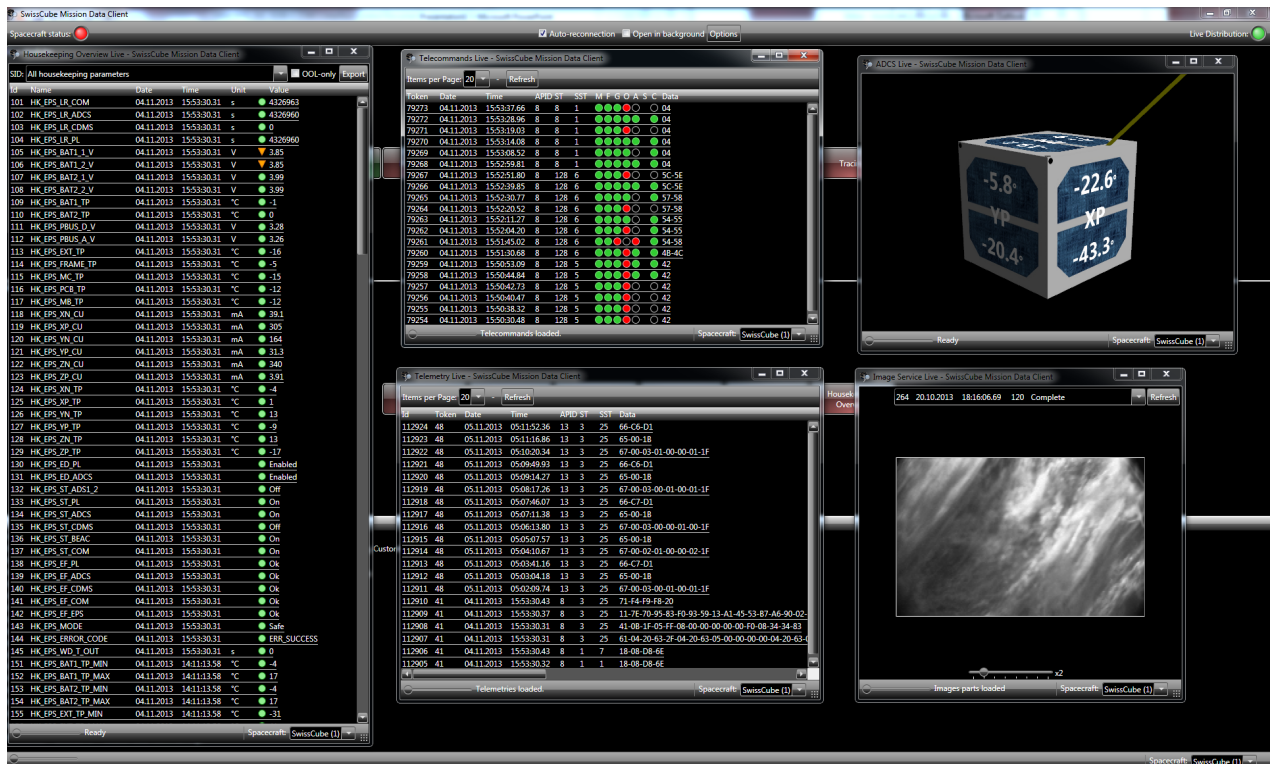


Figure 8: Example of SCS MDC modules.



## 2 TYPES OF DATA, DATA FLOW AND PROTOCOLS

This section provides an overview of the QB50 specific definition of data and the specific implementation using the SCS.

### 2.1 Types of data

The description of the types of data provided here is consistent with [R1]. The CubeSat creates and handles 4 types of data:

#### 2.1.1 Payload Data (PD)

Data that are generated by the QB50 teams' payload. The definition and structure of the data is to be defined by the teams.

#### 2.1.2 House-Keeping Data (HSK)

Data that are generated by the CubeSat used for monitoring the status and health of the satellite. It includes parameters like bus voltages, internal temperatures, incoming solar power and more. This is as well to be defined by the teams.

#### 2.1.3 Science Data (SD)

Data that are generated by the Science Unit and send to the OBC for downlinking to the ground. Upon reception at OBC, additional parameters shall be attached to the packet. The type and format is defined by MSSL, TUD and UiO for the specific sensor units [R6]. The SD shall be send by the teams to the QB50 server.

#### 2.1.4 Whole Orbit Data (WOD)

A set of important HSK data collected over the whole orbit with a frequency of 1/60 [Hz] (once a minute) and stored during at least 96 minutes before the pass. The WOD will provide the QB50 project with a mean of monitoring the health of the overall constellation.

The parameters are defined by QB50 (see below) shall be send by the teams to the QB50 server. WOD are currently defined as: (see [R4])

	Data set 1								Data set 2 – 32
Time	Mode	Battery bus voltage	Battery bus current	Current regulated bus 3.3V	Current unregulated bus 5.0V	Temp. comms	Temp. EPS	Temp. battery	
32 bits	1 bit	8 bits	8 bits	8 bits	8 bits	8 bits	8 bits	8 bits	1767 bits

Only one time is stored for 32 data sets, the other are computed logically.

Figure 9 below shows the flow of each data type within the SCS and with external interfaces.

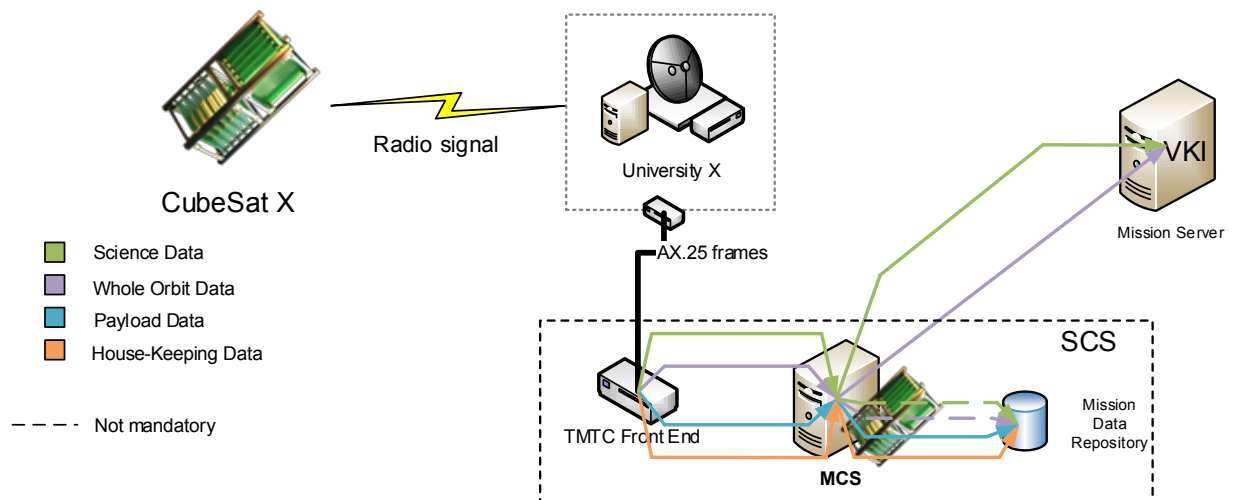


Figure 9: Overall data flow.

## 2.2 Requirements related to the use of Radio Amateur bands

The QB50 project shall be compliant with few requirements, which come out from the International Radio Regulations produced by the International Amateur Radio Union (IARU):

1. The RA service is an open system, so all amateurs should be allowed to receive and decode all transmissions. If the encoding method is not already recorded within the RA fraternity, developers have to supply for free downloading the ground segment software.
2. Command uplinks can be encrypted, but nothing else can be.
3. There can be no copyright or Intellectual Property Right (IPR) on data downlinked from a CubeSat using the amateur satellite service.

All RA must be able to decode the data and they should be able to send it at their convenience to the CubeSat's teams and the QB50 VKI servers.

## 2.3 Data coding basic process (for background)

This section has the purpose of providing background to the description of the protocols that will follow. It describes the process, also found in the GFWG report [R5], of the downlinked data. Similar processing takes place for the downlink and for the uplink. As Figure 10 shows for downlink, there are two layers of protocols that are applied on the "raw" data:

- a packet protocol (represented as wagons in Figure 10),
- and a frame protocol, which carries several packets (engine + train).

The frame protocol is assumed to be the AX.25 commonly used by the Radio-Amateur community.

As can be seen in Figure 10, all data leaves the CubeSat via the on-board telecom system in a packet format, which has been then inserted in the AX.25 frames, and then modulated in the radio-frequencies (modulation scheme baselines for QB50 are provided by the GFWG recommendations).

Once transmitted to the ground, the RF demodulation takes place at the ground station. The data is then recovered in digital form. The AX.25 decoding (“de-frame”) will most certainly be done at the ground station as well. In the SCS, this function is done, among others, at the CubeSat team server (in the TM/TC Front End). It can be decoupled from the TM/TC Front End if necessary.

The packet decoding is in principle to the discretion of the CubeSat team, as it is the responsibility of the team to choose the packet format it wishes to. We recommend a simplified version of the ECSS-PUS for use in the SCS [R7], but each team can also implement this ECSS-PUS to its liking.

The uplink process is similar with a packetization of the commands, possibly an encryption as well, followed by an encapsulation into the AX.25 protocol.

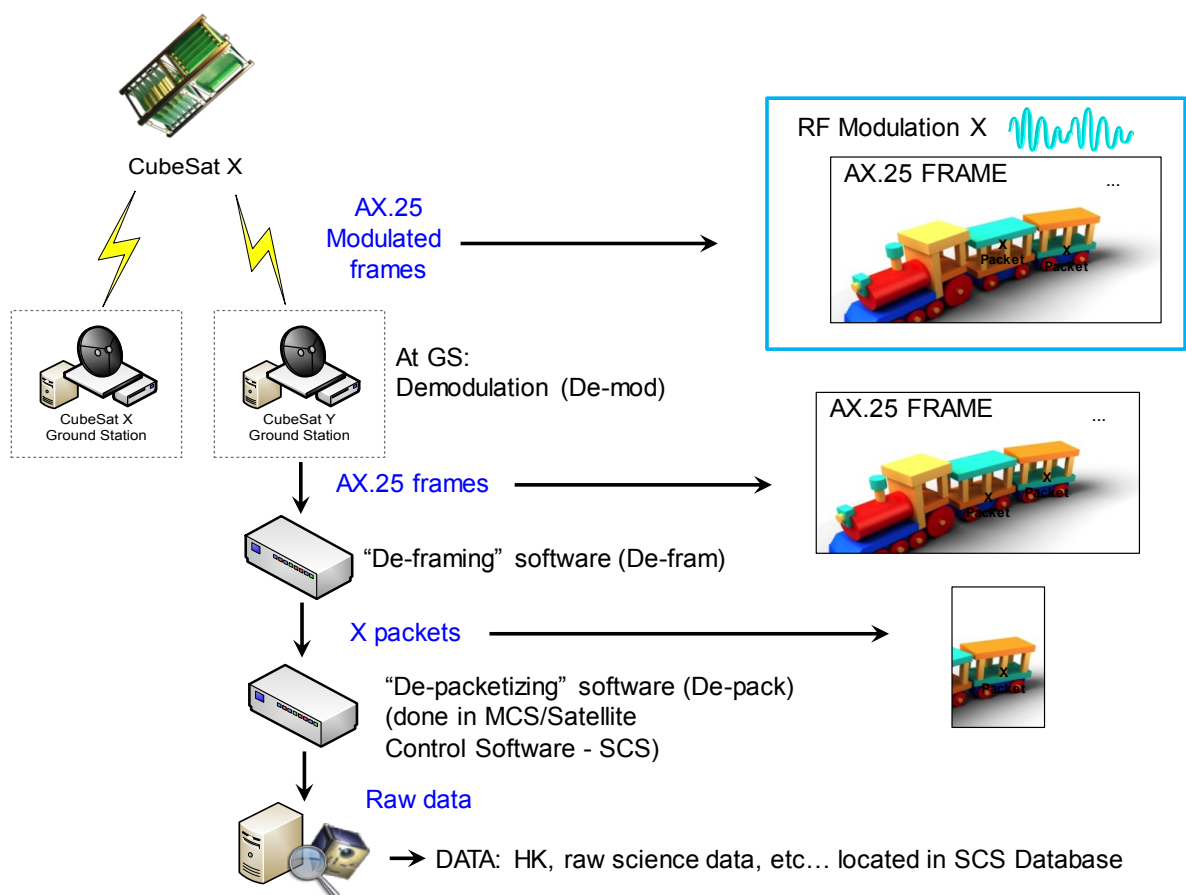


Figure 10: Simplified data downlink process and protocols.

## 2.4 AX.25 Transfer Format

The document QB50-EPFL-SSC-SCS-ICD-AX.25-TFF-3-1 [R8] describes in detail the format of the Radio-Amateur AX.25 Transfer Frames, a simple Data Layer (layer 2) protocol based on the Unnumbered Information Frames (UI-Frames) of the AX.25 protocol and adapted for space applications.

The reason it is based on the AX.25 protocol is for compatibility. As with most CubeSat projects, the QB50 project makes use of the amateur radio network, and thus of amateur radio equipment. This



equipment uses AX.25 for "high data-rate" digital transmissions. The transfer frames should therefore be compliant with the format defined in the AX.25 protocol to be able to use the existing amateur radio equipment.

Only the Unnumbered Information Frames (UI-Frames) of the protocol are used to allow for easy implementation on-board satellites which have limited resources. But to provide the necessary functionalities of space-to-ground transportation in the Transfer Frames (e.g. frame loss detection, on-board arrival detection, time correlation, etc.), a Secondary Header has been added to the Telemetry Transfer Frame. The fields of this Secondary Header are adapted from the Packet Telemetry Standard [R7].

In result, the hereby defined protocol is asymmetric, which is a property common to space-to-ground protocols. This is due to the fact that resources are very unbalanced between the spacecraft hardware and ground hardware.

We refer the reader to [R8] for details.

### 2.4.1 Unnumbered Information Frame

The frame in compliance with AX.25 protocol is as followed:

Flag	AX.25 Transfer Frame Header (128 bits)				Information Field	Frame-Check Sequence	Flag
	Destination Address	Source Address	Control Bits	Protocol Identifier			
8	56	56	8	8	0-2048	16	8

### 2.4.2 Information Field specification

This field contains the data to transmit. In case of telemetry, additional information is necessary in a secondary header. This secondary header is as such:

AX.25 Transfer Frame Information Field (0-2048)										
Telemetry Transfer Frame Secondary Header (32 bits)						Data	Telemetry Transfer Frame Trailer (8-72 bits)			
Frame Identification (8 bits)			Master Frame Count	Virtual Channel Frame Count	First Header Pointer		Frame Status (8 bits)			Time
Version Number	Virtual Channel ID	Spare					Time Flag	Spare	TC Count	
2	3	3	8	8	8	0-2008	4	2	2	0-64

### 2.4.3 Overhead efficiency

According to this format, in case of downlinked data, the size of the frame is 2208 bits out of which 264 bits are overhead (12%). 104 of them to account for packet sequencing, virtual channels management and time correlation.





## 2.5 Packet Structure Recommendation

As per the ECSS, the management of the data is done via standard structure (on-board applications and services). That structure is reflected in the organization of the packet protocol. Each field of the packet protocol is described in detail in QB50-EPFL-SSC-SCS-ICD-TMTC\_PD-4-1 [R2].

The highest packet structure are provided here for information for the telecommands and telemetries. We recommend to the reader to refer to [R2] for further understanding.

### 2.5.1 Telecommand Packet Data Field Structure

The uplink telecommands shall comply with the following format:

Packet Header (48 Bits)						Packet Data Field (Variable)			
Packet ID				Packet Sequence Control		Packet Length	Telecommand Data Field Header	Application Data	Packet Error Control
Version Number	Type	Data Field Header Flag	APID	Sequence Flags	Sequence Count				
3	1	1	11	2	14				
16				16		16	24	Variable	16

### 2.5.2 Telemetry Packet Data Field Structure

The downlink telemetries shall comply with the following format:

Packet Header (48 Bits)						Packet Data Field (Variable)			
Packet ID				Packet Sequence Control		Packet Length	Telemetry Data Field Header	Source Data	Packet Error Control
Version Number	Type	Data Field Header Flag	APID	Grouping Flags	Source Sequence Count				
3	1	1	11	2	14				
16				16		16	64	Variable	16

## 2.6 Error management

If a team wants to avoid using the protocol AX.25 and wish to implement a bit correction code, the SCS is developed in module and it would be possible to develop a replacement of the TMTC Front End to manage another protocol as input of this module and provide the ECSS packet as output. This development would have to be done by the team.





## 3 SERVICES

A service is a set of capabilities in a well-defined scope of functionalities that drive the relations between the flight and ground segment. The use of those services is a recommendation by ECSS based on experiences with past and present space missions. In the flight-proven version of the SCS, some of those services are already implemented. PUS are used as a menu from which the applicable services and service-levels are selected for the mission.

Note that the SCS has tailored the ECSS PUS to utilize services useful for CubeSat missions. It thus provides a light version of the ECSS PUS.

This chapter will present all services which are available. The detail is described in “QB50-EPFL-SSC-SCS-ICD-TMTC\_PD-4-1” [R2].

### 3.1 Standard services

#### 3.1.1 Telecommand verification - Type 1

This service provides the capability for explicit verification of each distinct stage of execution of a telecommand packet, from on-board acceptance through to completion of execution. It allows knowing at which stage of the communication an error has occurred or at which level the execution of a telecommand is running.

Although this service provides the possibility for explicit telecommand verification, there is no implication that all telecommands shall necessarily be verifiable at each distinct stage. For some of them, the application process can have little or no knowledge of the command execution.

The SCS has multiple level of acknowledgment to help users see at which stage telecommands are. Some of them are coming from the Cubesat's OBC:

- A: Telecommand acceptance
- S: Telecommand execution started
- C: Telecommand execution completed

All those acknowledgements are described in the Packet Utilization Standard (PUS) and their use is recommended.

#### 3.1.2 Housekeeping data reporting - Type 3

This service provides the capability of reporting to the ground of all information of operational significance that is not explicitly provided within the reports of another service.

It provides housekeeping data reporting, periodically and filtered. This reports sample sets of housekeeping parameters stored on-board. The list of parameter should monitor health and status of the CubeSat.



### **3.1.2.1 Whole Orbit Data (WOD)**

The Housekeeping Parameter Report (3, 25) can be used to downlink the Whole Orbit Data without repetition of the time as defined in the WOD Packet Format, see 2.1.4. For this, a housekeeping structure containing the required measures (voltages, currents, etc.) as parameters can be defined as repeating every minutes (1/60Hz). The time used to compute the actual measurement time is the time of the telemetry packet, which thus needs to be equal to the measurement time of the first data set.

The provided sample MIB already contains this definition and the automatic upload of the WOD to the VKI server will require the presence of this housekeeping structure.

### **3.1.3 Function management - Type 8**

An application process may be designed to support software functions that are not implemented as standard or mission-specific services, but whose execution may nevertheless be controlled from the ground segment. Such functions may control operations of a subsystem.

Each function shall be uniquely identified by a "Function ID". The Function ID define a specific execution of a subsystem, e.g. start manipulation, stop manipulation, configure, etc. A set of parameters may be necessary for a given function.

### **3.1.4 On-board operations scheduling - Type 11**

This service provides the capability to command and release pre-loaded telecommands at their due time. The on-board operations scheduling service shall maintain a command schedule which contains telecommand packets and their associated scheduling information. It should also have the capability to enable the scheduling, or disable it. The manipulation of telecommands as adding, delete and shifting in time should be possible.

### **3.1.5 Large data transfer - Type 13 (applicable to the uplink)**

This service is used by the ground system software to transfer large service data units in a controlled manner. The choice to use the large data transfer service is made by the initiator of a given service data unit.

The QB50 project requires being able to uplink script for science unit. As the size of those script could exceed the maximal size of a packet, this service authorizes and describes the way to uplink and transfer the science script from the ground to the CubeSat.

The process is as follow:

1. The script is split into parts. The maximum size of a single part is 65528 bytes.
2. Each part is sent as an individual telecommand packet.
3. All parts are sent in a row.
4. The client waits for acknowledgements (service 1).
5. When the last part is sent, the client waits for a telemetry report confirming the reception (service type 13, subtype 14). This report indicates the last consecutive part correctly received.
6. The user can resend missing parts until completion of the transfer.
7. The CubeSat OBC reconstructs the original data and is able to extract the script out of the application data.



The packets uplinked have a CRC16 and corrupted packets are detected. Nevertheless, once all packets are up on the satellite, the OBC shall check the overall script again as stated in the QB50 CubeSat requirements and SU ICDs.

### 3.1.6 On-board storage and retrieval - Type 15

The on-board storage and retrieval service is used by other services to store data and retrieve it to be downlinked to the ground.

In the scope of the mission, this service is used to store the science data that should be downlinked when there is a contact with the ground station. Especially mission-specific services are using this service.

## 3.2 QB50 Mission-Specific Services

QB50 Mission-Specific services are developed to handle the data that are produced by the science unit in each CubeSat.

This service however requires a complex handling of the communication state between both parties. This is easily done on-ground, but not necessarily on-board. For simplification the management of this data, three mission-specific services containing a single data report telemetry packet are developed.

Each of those services defines a packet for downloading science data that is stored in a specific memory of the CubeSat.

Triggering the actual download should be performed via the service 15 (on-board storage and retrieval), which manages the internal stores. Alternatively, but not recommended, it could also be triggered directly by a function (service 8).

The SU data reports are different for each service and are defined in the ICD of the specific science unit.

### 3.2.1 INMS Data management - Type 128

Type	Subtype	Signification
128	1	INMS Data Report

### 3.2.2 FIPEX Data management - Type 129

Type	Subtype	Signification
129	1	FIPEX Data Report

### 3.2.3 MNLP Data management - Type 130

Type	Subtype	Signification
130	1	MNLP Data Report

## 4 FLIGHT SW IMPLEMENTATION EXAMPLE

This section aims at helping CubeSat teams in their flight software implementation in a compatible way with the SCS. A separate appendix of this document has been created, QB50-EPFL-SSC-SCS-ICD-FSW-1-0 [R12], where an example of SwissCube flight software is presented (the reader is encouraged to read that appendix).

The flight software of SwissCube was initially intended to support a wide variety of services; more specifically, service types 1, 3, 4, 8 and 15 on top of the custom payload service type 128. Service types 1, 3, and 8 have been implemented as per the standard's definition. Service type 4 (Statistics reporting) has been merged with the service type 3 (Housekeeping & diagnostic data reporting), and downlinks of telemetry packets from these services are obtained through a service type 8 request. Furthermore, the functionality of service type 15 has been simplified by creating housekeeping parameter archives that are also downlinked by service type 3 through a service type 8 request. These modifications stay compliant with the CCSDS PUS and are explained in detail in document [R12]. Despite these slight alterations with the ECSS standard, the flight software of SwissCube remains a good starting example for a developer looking to implement the CCSDS PUS on a new CubeSat.

As an example of the specifics to be implemented for QB50, we discuss here a possible implementation of the Science Unit (SU) data flow, as illustrated on Figure 11. The Science Unit (SU) board takes measures depending on the running script. When the OBC receives SU data, it computes a header containing time, attitude and position and adds it to the data in order to get a SU packet. The SU packet is encapsulated into a CCSDS packet and stored in the memory or a specific storage unit. When the CubeSat operators want to get the SU packets to the ground, they issue a service 15 telecommand with the OBC as destination. This telecommand requests the downlink of all or part of a specific on-board store, i.e. the SU packet store. The COM board is responsible for deframing the AX.25 protocol (or the one, which is used) and check the integrity of the data in the frame. Telecommands inside the frame are then transmitted to their destination, i.e. the OBC. When the service 15 telecommand is delivered to the OBC, stored CCSDS packets are extracted from the memory and sent to the COM board as they are for encoding and transmission.

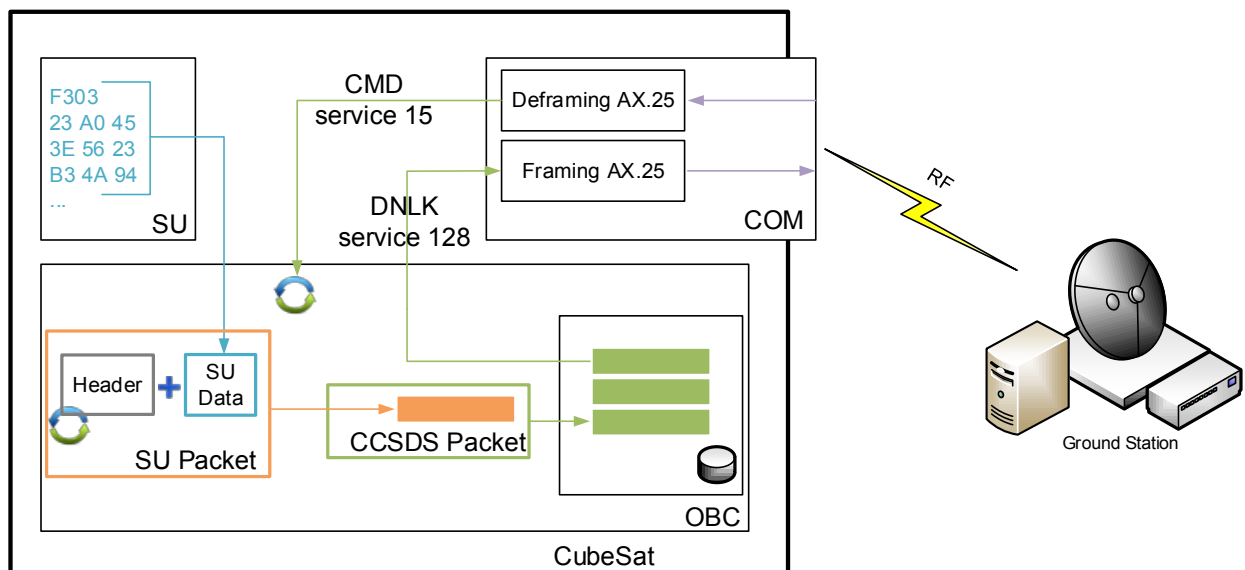


Figure 11 : On-board software save and downlink process

## 5 USING THE SCS FOR YOUR CUBESAT TESTS

The SCS can be used to test your CubeSat, and it is actually highly recommended to do so as it validates their compatibility. Figure 12 shows the test equipment needed for the SwissCube satellite, as an example. It is meant to have the CubeSat teams understand what additional software and hardware interfaces might be needed.

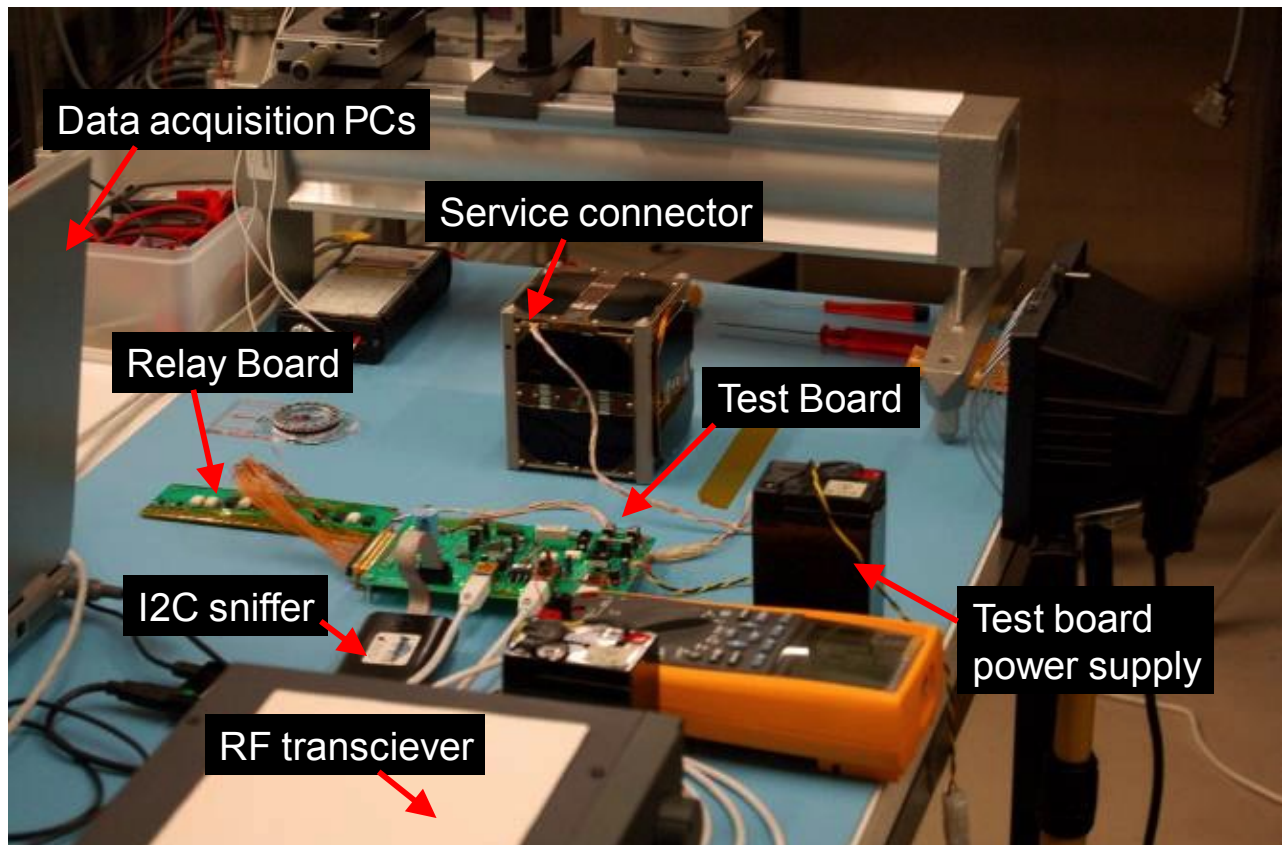


Figure 12 : Functional Test Equipment for the SwissCube CubeSat.

### 5.1 General Approach

SwissCube had a service connector, which allowed getting additional information during the development and testing phases than usually available via the regular communication channels. This connector allowed direct wired communication between the test equipment and the satellite. The service connector pins provide information about the batteries voltage, bus voltage and beacon signal as well as a sniffer on the I2C data bus and a digital link connected respectively at the input/output of the COM microcontroller (uplink/downlink).

In addition to the service connector the CubeSat was also tested via the telecom link (RF).

For this, two PCs served as interface with the test operator to send telecommands and acquire data from the satellite. One PC (PC1) was dedicated to the RF link while the other PC (PC2) was used for the wired link, i.e. the data from the service connector.



Each PC had an installation of the mission control software (Mission Data Client Monitoring, MCS, TM/TC Front End), which is used to transmit TC and receive TM during flight.

With this configuration, communication with the CubeSat could be done either via the service connector digital link or via the RF link or both. The digital link (uplink/downlink) was directly connected to the input (for uplink) and output (for downlink) pins of the COM microcontroller, the same pins were used for the RF. Thus all downlink information received by the service connector was also transmitted to the RF transmitter of the communication board. For the uplink, a switch on the test board allowed commands to be sent either via RF or digital link.

This setup enabled efficient testing of the communication channels and the system during integration while providing addition information critical to the investigation of encountered issues. Later, it gave realistic (as during mission operations) information on the system during the qualification and acceptance campaigns. Moreover, it adapted easily to the variety of development and test environments required for the elaboration of a CubeSat.

Figure 13 below shows the overall setup. It contains other software and hardware elements that are described in the next sections.

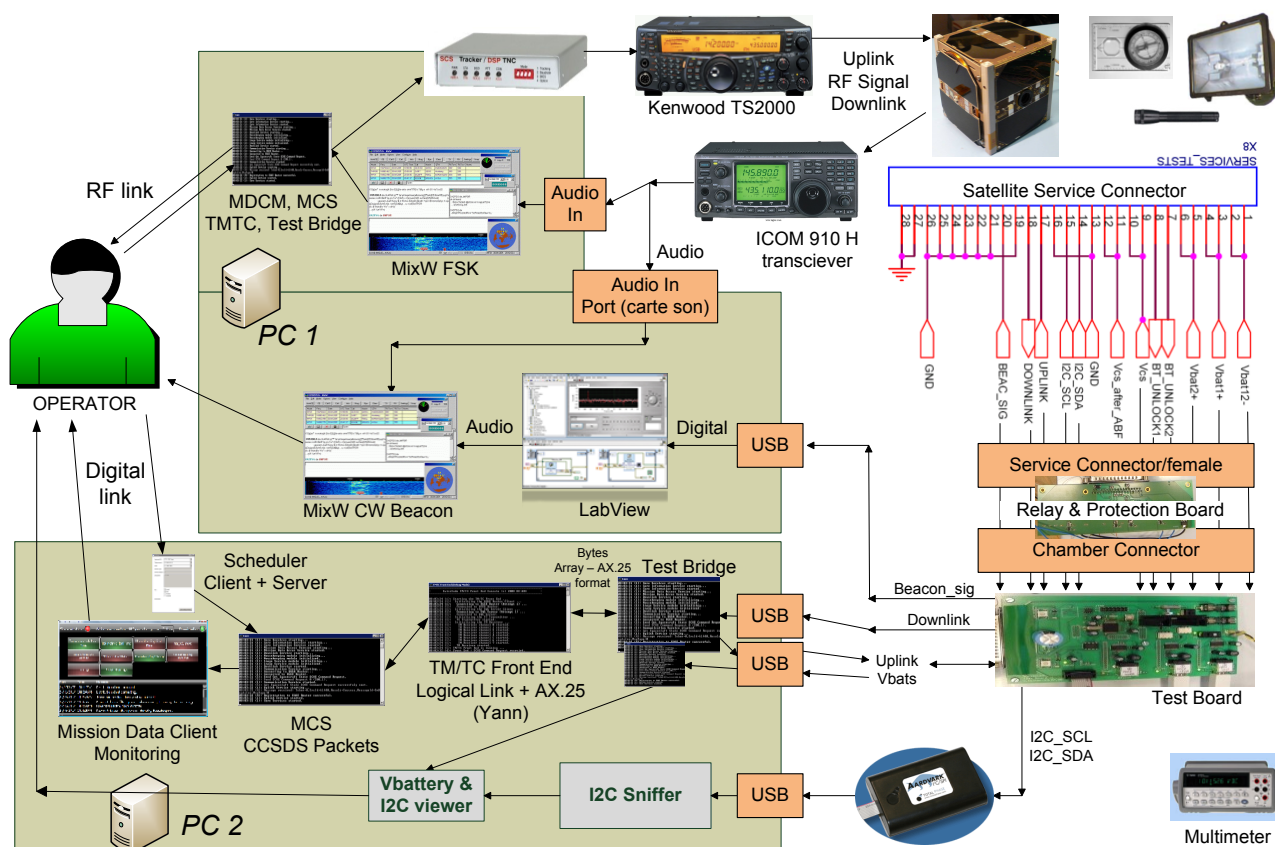


Figure 13: Hardware and Software Test Setup for the SwissCube Mission

## 5.2 Hardware Components

In addition to the two PCs, the following pieces of hardware equipment were developed or bought.



For providing the digital/analog signal interfaces between the service connector and the USB ports on the PCs, a test board was developed. This board also allowed charging the satellite's batteries between the test sessions.

As the CubeSat can be remotely placed (a few meters) of the test board, e.g. when in a chamber for thermal or vacuum tests, a protection board with optocouplers was inserted between the satellite's service connector and the test board.

The RF communication link was ensured by two transceivers, a TNC, attenuators and, when required, an antenna inside the test chamber.

Finally, an I2C bus data sniffer that connects directly to the test equipment was used to provide the bus signal via USB to PC2.

### **5.3 Software Components**

In addition to the software systems used during the flight (MCS, TM/TC Front End, etc.), the following pieces of software were used during testing.

A software counter-part for the test board was developed to send to and receive data from it via USB. A simple data exchange protocol is used between this "test bridge" and the test board. This allows sending commands and receiving telemetry directly via the service connector. It also reads all additional information provided by the test board about the CubeSat's health (battery voltage, bus voltage, etc.).

PC2 was connected to the I2C bus data sniffer. An "I2C Sniffer" application was used to read data from the hardware sniffer and send them to a GUI. This user interface called "Vbat & I2C Viewer" also displayed the current state of the batteries.

## 6 SCS AS A NETWORK

The SCS can be used to create a network of ground stations (GS), as shown in Figure 1 (see also Figure 14). Of course, the ground stations must be compatible with each other such that the CubeSat's frequencies, modulation and frame protocols can be retrieved any ground stations. The link is created with the "Ground Station Manager" software shown in Figure 1.

Each team can have its own network to make the separation of data easier. If two teams use the same ground stations, then two "Ground Station Managers" will be installed.

Once the network is established, the following operation scenario can occur:

- 1) Check the availability of the partner ground station for the day or pass that you need it;
- 2) Before the pass, the CubeSat operator will choose from its mission control center which GS to uplink the commands (in the Commanding Client); he/she can switch GS as he/she wishes during the pass (for instance if one ground station is resetting its position and is unavailable, or if it fails);
- 3) As the communication is established with the CubeSat, the SCS will handle receiving the data from various listening GS;
- 4) In the case 2 or 3 consecutive GS (located along the ground track of the CubeSat) are used, the operator will have to select the uplink GS as the CubeSat moves along (in the Commanding Client). The downlink will be picked by all GS, which are connected, at any moment and automatically sent to the SCS; the explanation of installation of the Ground Station Manager will be found in D250.3, SCS User Manual.

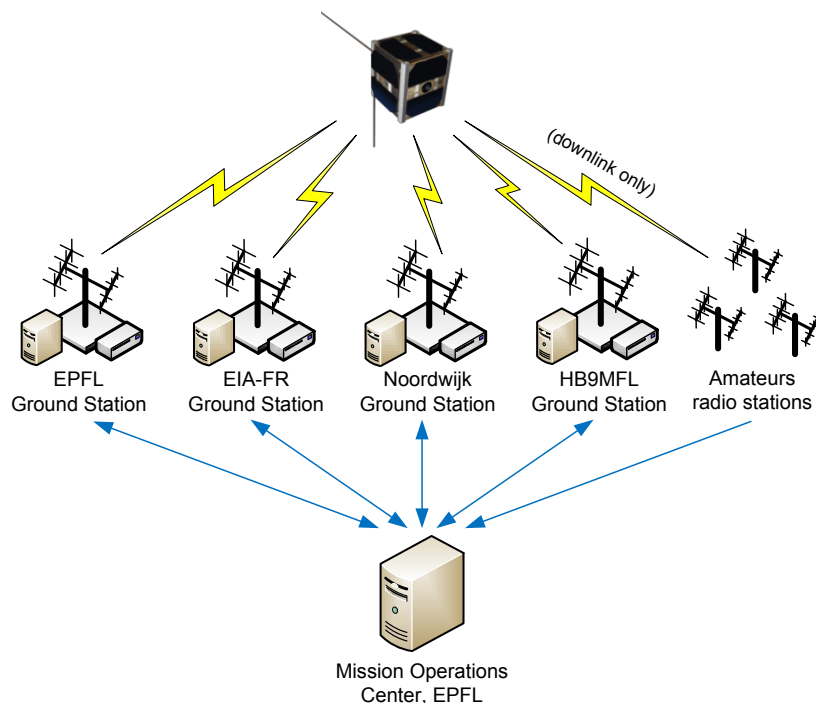


Figure 14: Example ground station network for SwissCube

Figure 15 shows the current list of QB50 teams that are planning to use the SCS [R15]. Your team should then follow the steps:



- 1) Get in touch with 2-3 teams (or more!) that are nicely located with your own ground station (optimization to provide the required science data return needs to be performed for you);
- 2) Come up with a formal (oral or written) agreement of support; work out what time of the teams' ground station availability you can use;
- 3) Once your network is identified, each of the partner teams need to install Ground Station Manager of the other CubeSat teams to receive data from each satellite.

	interest	confirmed	CDR	CubeSat name
<b>Europe</b>				
Aalto University	👍	✅	Ok	Aalto-2 (FI01)
INSSET	👍	✅	Ok	IP2 SAT (FR03)
ISAE	👍	✅	Ok	EntrySat (FR02)
UPEC	👍	✅	Ok	OGMS-SA (FR04)
FH Aachen University of Applied Sciences	👍	✅	Ok	DragSail (DE04)
TU-Dresden	👍	✅		Somp2 (DE02)
Universidad Politecnica de Madrid	👍	✅	Ok	QBITO (ES01)
Istanbul technical University (SSDTL)	👍	✅		BEEAGLESAT (TR01)
Cranfield University	👍	✅	Ok	DeltaDsat (GB01)
VZLU	👍	✅	Ok	VZLUsat-1 (CZ02)
University of Patras	👍	✅	Ok	UPSat (GR02)
VKI	👍	✅		Qarman (BE05)
UCL	👍	✅		UCLSat (GB03)
<b>Asia</b>				
Herzliya Science Center	👍	✅	Ok	Hoopoe (IL01)
Istanbul Technical University / Air Force Academy			Ok	BeEagleSat (TR01)
Beihang University (BUAA)	👍	✅	No	BUSat-1 (CN01)
National University of Defense Technology	👍	✅	?	NUDTSat (CN06)
Nanjing University of Science and Technology	👍	✅	?	NJUST-1 (CN03)
Northwestern Polytechnic University	👍	✅	?	Aoxiang-1 (CN04)
ShanghaiTech University	👍		?	STU-1 (CN07)
National University of Singapore	👍		Ok	NUSQB50 (SG01)
Anna University	👍	✅	Ok	Anusat-2 (IN01)
South Korea Advanced Institute of Science and Technology	👍	✅	Ok	KAIST/LINK (KR01)
<b>Oceania</b>				
University of Adelaide	👍		No	SUSat (AU01)
University of Sydney	👍	✅	Ok	i-INSPIRE II
<b>North America</b>				
University of Alberta	👍	✅	Ok	AlbertaSat (CA03)
Stanford University	👍	✅	No	Stanford (US03)
<b>South America</b>				
Universidad Distrital Francisco José de Caldas	👍	✅	Ok	QB-Colombia (CO01)
<b>Africa</b>				
University of Stellenbosch	👍	✅	?	ZA-Aerosat (AZ01)
	29	24	17	

Figure 15 : Teams that have an interest or confirmed the use of SCS (as of 30-04-2014)



## 7 CONCLUSIONS AND PREPARATION STEPS FOR CUBESAT TEAMS

This document provides an overview of the SCS description, interfaces, tailoring and extensibility capability. It also provides sections meant to help CubeSat teams with their flight software implementation of the communication with the SCS, and insertion of their ground segment into a network. The details of each of these topics can be found in accompanying appendix documents.

### Next steps for the QB50 CubeSat teams

Before the SCS is delivered to you, you can start preparing for it. The software will come along with an installation manual that will guide through its deployment. The requirements for this installation are already provided in **Error! Reference source not found.** section.

The SCS is a versatile and generic mission control system and, once deployed, it requires configuration and maybe customization for your cubesat. Most of the services and functionality you will need are already natively supported by the SCS and requires only configuration.

The mission configuration is contained in a database called Mission Information Base (MIB). This database is an XML file in the QB50 SCS. This file defines:

- the mission constants and general properties;
- the applications and subsystems;
- the housekeeping structures and their parameters;
- the parameter calibrations;
- the on-board functions;
- the telecommands and their parameters;
- the SCOE commands and their parameters.

An example is provided to show the structure of the file and its content. The modification of the MIB and therefore the configuration of the SCS require basic knowledge in CCSDS/PUS [R7].

When the SCS is delivered to CubeSat teams, the MIB will be empty, but its structure will be defined and teams will use parameters they have completed in xls sheet [R13/R14] to fill in the specific MIB.

As a preparatory activity, you can list the functions and the housekeeping parameters for your cubesat. Two Excel documents [R13, R14] are available in the annex of this document to support you in this task. The first worksheet contains instructions. The second contains tables with all relevant information that will be needed for the actual configuration. Finally, the last worksheet contains examples.

If you need or want to tailor the SCS more, the Mission Data Client Extensibility User Guide [R11] and the Mission Control System Extensibility User Guide [R10] are the starting point to design such changes.