Horus
IMSETY
User Requirements Document
Version 1.2 13th April 2007

Project team: Jeroen Keiren 0569081
Frank Koenders 0575629
Thijs Nugteren 0574426
Joeri de Ruiter 0578312
Stijn Stiefelhagen 0579816
Carst Tankink 0569954
Pim Vullers 0575766
Freek van Walderveen 0566348

Project manager: Egbert Teeselink

Senior management: L. Somers TU/e (HG 7.83)
M. v.d. Brand TU/e (HG 7.44)

Adviser: R.J. Bril TU/e (HG 5.09)

Customer: E. v. Breukelen ISIS

Computer Science, Eindhoven University of Technology, Eindhoven
Abstract

This document describes the user requirements for the IMSETY system. IMSETY shall be a system providing an easy to use interface for experiments in a microgravity environment. This document describes the environment of IMSETY and the general requirements and constraints that need to be fulfilled for IMSETY to be satisfactory for Innovative Solutions in Space BV (ISIS). These requirements were established in accordance with the requests formulated by ISIS’s representative.
Contents

1 Introduction
   1.1 Purpose .............................................. 1
   1.2 Scope .................................................. 1
   1.3 List of definitions ..................................... 1
      1.3.1 Domain definitions .............................. 1
      1.3.2 Development process definitions .......... 2
   1.4 List of references ................................... 3
   1.5 Overview ............................................. 3

2 General description
   2.1 Product perspective .................................. 4
   2.2 General capabilities .................................. 4
      2.2.1 Administration ................................ 4
      2.2.2 Experiment definition ...................... 4
      2.2.3 Experiment observation and processing .... 5
      2.2.4 Communication ............................... 5
   2.3 General constraints .................................. 5
   2.4 User characteristics ................................ 6
      2.4.1 User rights ................................... 6
   2.5 Environment description ......................... 7
   2.6 Assumptions and dependencies .................. 9

3 Specific requirements
   3.1 Capability requirements ............................. 11
      3.1.1 General requirements ......................... 11
      3.1.2 Administration ................................ 11
      3.1.3 Experiment scheduling ...................... 12
      3.1.4 Experiment observation and processing .... 13
      3.1.5 Communication ............................... 14
   3.2 Constraint requirements ............................ 14

A Use cases
   A.1 User ................................................. 15
      A.1.1 Log on to the system ......................... 15
      A.1.2 Fail to log on to the system ................ 15
      A.1.3 Log out ....................................... 15
      A.1.4 Retrieve payload logs ..................... 16
   A.2 Administrator ...................................... 16
      A.2.1 Register a satellite ......................... 16
      A.2.2 Retrieve satellite details .................. 16
      A.2.3 Update satellite details .................... 16
      A.2.4 Unregister a satellite ...................... 17
CONTENTS

A.2.5 Register a payload ........................................ 17
A.2.6 Retrieve payload registration ............................... 17
A.2.7 Update a payload registration ............................. 18
A.2.8 Unregister a payload registration ......................... 18
A.2.9 Create an account for a user ................................ 18
A.2.10 Retrieve account information of a user .................. 18
A.2.11 Modify linked payload registrations with a scientist or observer account ... 19
A.2.12 Update the account information of a user ............... 19
A.2.13 Delete the account of a user .............................. 19
A.2.14 Set up possible payload commands ....................... 20
A.2.15 Set up possible payload commands through a plain text file .............. 20
A.2.16 Set up possible payload commands through an interface .................. 20

A.3 Scientist ......................................................... 21
A.3.1 Retrieve observations of a payload ......................... 21
A.3.2 Retrieve a high-resolution picture of an observation .......... 21
A.3.3 Retrieve an observation in real time ....................... 21
A.3.4 Retrieve a video stream of an observation in real time .......... 22
A.3.5 Retrieve high-resolution pictures of a selection of a video stream .......... 22
A.3.6 Postprocess observation visualizations ................... 22
A.3.7 Compare two observations .................................. 23
A.3.8 Export observation data .................................... 23
A.3.9 Retrieve list of passes ...................................... 23
A.3.10 Program experiment ....................................... 24
A.3.11 Retrieve experiment details ............................... 24
A.3.12 Schedule experiment ..................................... 24
A.3.13 Update an experiment .................................... 24
A.3.14 Fail to update an experiment ............................. 25
A.3.15 Intervene in an executing experiment .................... 25
A.3.16 Delete an experiment ..................................... 26

A.4 Observer ....................................................... 26
A.4.1 Observe an experiment ..................................... 26

A.5 System ........................................................ 26
A.5.1 Connect with GENSO ....................................... 26
A.5.2 Fail to connect with GENSO ............................... 27
A.5.3 Connect with MCS ......................................... 27
A.5.4 Fail to connect with MCS ................................. 27
A.5.5 Retrieve observations from GENSO ....................... 27
A.5.6 Upload experiment to GENSO ............................. 28
A.5.7 Fail to communicate with a satellite ....................... 28
Document status sheet

<table>
<thead>
<tr>
<th>Document title</th>
<th>User Requirements Document</th>
</tr>
</thead>
<tbody>
<tr>
<td>Document identifier</td>
<td>IMSETY/doc/URD</td>
</tr>
<tr>
<td>Author(s)</td>
<td>Thijs Nugteren, Joeri de Ruiter, Carst Tankink, Freek van Walderveen</td>
</tr>
<tr>
<td>Version</td>
<td>1.2</td>
</tr>
<tr>
<td>Document status</td>
<td>Internally approved</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Author(s)</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>21-02-2007</td>
<td>Thijs Nugteren, Joeri de Ruiter, Carst Tankink, Freek van Walderveen</td>
<td>First version up for internal review.</td>
</tr>
<tr>
<td>0.2</td>
<td>21-02-2007</td>
<td>Thijs Nugteren, Joeri de Ruiter, Carst Tankink, Freek van Walderveen</td>
<td>Most changes from internal review included.</td>
</tr>
<tr>
<td>0.3</td>
<td>02-03-2007</td>
<td>Thijs Nugteren, Joeri de Ruiter, Carst Tankink, Freek van Walderveen</td>
<td>Changes from external reviews included. Version for second internal review.</td>
</tr>
<tr>
<td>0.4</td>
<td>02-03-2007</td>
<td>Thijs Nugteren, Joeri de Ruiter, Carst Tankink, Freek van Walderveen</td>
<td>Changes from second internal review included. Version for external review and conditional approval.</td>
</tr>
<tr>
<td>0.5</td>
<td>06-03-2007</td>
<td>Thijs Nugteren, Joeri de Ruiter, Carst Tankink, Freek van Walderveen</td>
<td>Changes incorporated from last review sessions, technical adviser input and customer input. Version for third internal review.</td>
</tr>
<tr>
<td>1.0</td>
<td>06-03-2007</td>
<td>Thijs Nugteren, Joeri de Ruiter, Carst Tankink, Freek van Walderveen</td>
<td>Changes incorporated from third review session, up for external approval.</td>
</tr>
<tr>
<td>1.1</td>
<td>06-03-2007</td>
<td>Thijs Nugteren, Joeri de Ruiter, Carst Tankink, Freek van Walderveen</td>
<td>Added failure use cases.</td>
</tr>
<tr>
<td>1.2</td>
<td>13-04-2007</td>
<td>Thijs Nugteren, Joeri de Ruiter, Carst Tankink, Pim Vullers, Freek van Walderveen</td>
<td>Added URCAR57</td>
</tr>
</tbody>
</table>
Document change report

<table>
<thead>
<tr>
<th>Document title</th>
<th>User Requirements Document</th>
</tr>
</thead>
<tbody>
<tr>
<td>Document identifier</td>
<td>IMSETY/doc/URD</td>
</tr>
<tr>
<td>Date of changes</td>
<td>07-03-2007</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Section number</th>
<th>Reason for change</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.2.3</td>
<td>Introduction mark for deletion</td>
</tr>
<tr>
<td>3.1.3</td>
<td>URCAR36 priority changed to could have</td>
</tr>
<tr>
<td>3.1.4</td>
<td>New requirement (URCAR57)</td>
</tr>
<tr>
<td>3.1.5</td>
<td>New requirement changed IDs</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

1.1 Purpose

The user requirements document (URD) contains the requirements for the Interface Making Satellite Experiment Tracking easY (IMSETY). These requirements are a negotiated agreement between Innovative Solutions in Space BV (ISIS) and Horus. All of the listed requirements, and only these, will be implemented in IMSETY, according to their priorities. Any changes to these requirements require the full consent of both parties.

1.2 Scope

IMSETY will provide scientists and observers with an easy to use interface to conduct experiments on space based payloads and possible reference payloads on earth.

The system will provide the means to compose, schedule and observe experiments and intercede with these schedules in real time. The system will communicate to the users whether or not this scheduling and these intercessions are possible at any given moment in time.

1.3 List of definitions

1.3.1 Domain definitions

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Booking</td>
<td>A granted request to GENSO to use a pass for one-way or two-way communication with a satellite.</td>
</tr>
<tr>
<td>Experiment</td>
<td>A schedule of commands providing interaction between scientists and their samples. When this sample is part of a payload, the experiment is a microgravity experiment, whereas it is a reference experiment when the sample is based on earth. The only difference between a microgravity experiment and the corresponding reference experiment is the absence of the microgravity condition in the latter.</td>
</tr>
<tr>
<td>GENSO</td>
<td>Global Educational Network for Satellite Operations. A network of ground stations meant to increase the number of contact moments with educative satellites, by connecting individual ground stations.</td>
</tr>
<tr>
<td>Ground station</td>
<td>A ground station provides telemetry, tracking, and (not always) control functions for spacecraft and satellites in range of the ground station.</td>
</tr>
<tr>
<td>Horus</td>
<td>Software engineering team developing IMSETY.</td>
</tr>
</tbody>
</table>
CHAPTER 1. INTRODUCTION

IMSETY  Interface Making Satellite Experiment Tracking easY. The interface to satellite-based experiments, to be developed by Horus. The terms IMSETY and system will be used interchangeably.

ISIS  Innovative Solutions in Space BV, a company striving to develop miniaturized modular satellite (sub)systems that meet customer requirements, to gain a strategic position in the field of small satellite (sub)systems and to promote and stimulate the development of small satellite missions. The customer for whom IMSETY is being developed.

MCC  Mission Control Client, provides tracking through the different ground stations, used for data and control connections. Dedicated passes may be requested and will be taken care of by the GENSO booking mechanism.

MCS  Mission Control Software, provides the communication with satellites through GENSO. IMSETY will need to communicate with this software to send commands to the satellite.

Microgravity  A condition of near weightlessness. In a satellite, there is microgravity instead of true weightlessness, among others due to the fact that a satellite must apply force to stay in orbit.

Observation  Information gathered about sample. Examples are pictures, video or temperature.

Pass  The moment a satellite flies within range of communication of a ground station.

Payload  A sample and the hardware used to control and observe it, fitted by ISIS into a satellite for a scientist.

Postprocessor  A subsystem manipulating representations. This processing is for example the application of a red filter to a picture.

Preprocessor  A subsystem manipulating “raw” data. This processing is mainly the sorting of raw data into specific data types.

Real-time  Real-time communication is a form of communication in which data is observed the moment it is gathered. Due to its nature, satellite communication can never be real time. However, we use this term to refer to the form of communication which provides data as soon as possible, meaning the data is sent by a satellite and observed the moment this data arrives with a user.

Representation  A representation is data shown in a way that can be appreciated by humans.

Satellite  A nanosatellite deployed by ISIS in which they can fit one or more payloads. The nanosatellite provides communication with the payloads.

Scientist  A person conducting experiments, a client of ISIS.

Window of opportunity  A continuous series of one or more booked passes. In this window, communication with the satellite is possible.

1.3.2 Development process definitions
CHAPTER 1. INTRODUCTION

MoSCoW  The model used by Horus for the prioritization of user requirements and use cases. The letters M, S, C, and W have the following meanings:

M  Must have; these requirements are essential for the product.

S  Should have; these requirements are not critical for the product to work, but are nearly as important as the Must haves, meaning they must be implemented if at all possible.

C  Could have; requirements which are not critical to the product’s success. If they can be implemented with little development costs, they can increase customer satisfaction.

W  Won’t have; these requirements will not be implemented in this project. However, they would be nice to have in future versions of the product.

URD  The User Requirements Document for IMSETY, this document.

1.4 List of references


1.5 Overview

The remainder of this document is divided into two chapters, general description (chapter 2) and the specific requirements (chapter 3).

The general description gives a clear and concise description of IMSETY. It consists of:

• a description of the relation of IMSETY to other (sub)systems developed by third parties,

• the main capabilities of IMSETY,

• general constraints for the product. In particular, the background of constraints and the reason for their existences is given here,

• the characteristics of different user roles,

• a description of the operational environment, and

• the assumptions made during the definition of specific requirements.

The specific requirements are subdivided in capability requirements, describing what ISIS wants, and constraint requirements, describing how the requirements should be implemented.
Chapter 2

General description

2.1 Product perspective

IMSETY is a product without any predecessor currently deployed with ISIS. It shall be a crossover between a scientific experiment interface and a satellite mission interface. An example of the latter if the one developed for the AAUSAT II satellite [1]. IMSETY should use an MCS and GENSO to communicate with satellites, and abstract from its communication process. Furthermore, the system should allow the plugging in of (sub)systems for the pre-, and postprocessing and the representation of specific data types.

2.2 General capabilities

IMSETY shall have the capabilities described below in order to provide an easy interface to microgravity experiments.

2.2.1 Administration

Administration of the system is done by managing user accounts, satellites and payloads. The managing is done by registering, unregistering and updating the information of the managed entities.

The information for satellites and payloads kept by the system, is limited to the information needed by the system.

2.2.2 Experiment definition

In figure 2.1 you find a Petri net which depicts the phases an experiment goes through, as well as the order in which actions have to be taken. Note that this model is quite a simplified version of the real process, in which one would also need to take into account that communication is limited. This would highly complicate the “edit” and “intervene” parts, which is not desired in order to keep the model clear. The “edit” part of the model describes that before the execution of an experiment has been started in the satellite it can still be edited by the scientist (note that there should be opportunity to send the edited experiment to the satellite before execution of the scheduled experiment starts). “Intervene” describes that the scientist can modify the experiment while it is being executed, provided that there is a communication opportunity.

The administrator shall define with which payloads a scientist is allowed to work. A scientist can create an experiment on a particular payload. Experiments consist of sequences of commands executed sequentially. These experiments are then scheduled at a certain moment in absolute time and uploaded to the satellite. The scientist does not have to stay logged in for the experiment to be uploaded. IMSETY will try to upload an experiment automatically when possible if the
experiment is set for uploading by the scientist. The satellite then executes the experiment at the specified moment. If an experiment is not executing yet a new experiment can be uploaded if there is an opportunity to communicate with the satellite before the experiment is executed. During the execution of an experiment, a scientist can intervene by sending new commands to the satellite if two-way communication is possible during that period. It should also be possible to execute a reference experiment on a sample which is not located in a satellite.

### 2.2.3 Experiment observation and processing

There are two ways for experiment observation. The first is observing data in real time. The other possibility is at a later time observing data that was stored by the satellite as the experiment was executed, and that has been imported into IMSETY. The user tells IMSETY what data he would like to retrieve and IMSETY will try to download this data. The user does not have to stay logged in because IMSETY will try to download the requested data automatically when possible. When the data has been downloaded the user will be notified about this the next time he logs in.

In general, the data can be in any form, meaning the system will have to provide a plug-in system for the preprocessing, representation and postprocessing of this data. IMSETY will include a plug-in for visualizing pictures and real time 'video' (several low resolution pictures in sequence). When watching real time video, a scientist can mark the periods interesting to him, and then request the download of the pictures of these periods in higher resolution. In order to keep enough storage space available at the satellite, IMSETY will provide the possibility to mark the high resolution pictures for deletion.

### 2.2.4 Communication

The system shall communicate with a satellite using an MCS, which in turn uses GENSO to abstract several ground stations into one ground station with a large window of opportunity. This communication consists of uploading experiments and commands, booking passes and downloading or streaming requested data.

### 2.3 General constraints

There are two main reasons for constraints in the design of IMSETY: satellites and end users. Since IMSETY will communicate with satellites, several restrictions exist in the way data is transmitted. Firstly, data communication with satellites occurs at a limited bandwidth. This is slow compared to the communication speed achieved by land based communications. Secondly, since a satellite is in orbit over the Earth, there is no constant communication with the satellite, leading to a need to communicate with the satellite only on those moments when it passes a ground station. Furthermore, communication with a satellite is potentially unreliable. Therefore, the system should be able to cope with unexpected interruptions in satellite communication. This
means in case that one of the systems IMSETY uses to communicate with satellites tells it that it cannot execute the requested command, this must be communicated to the user. This also means that IMSETY does not need to check data inconsistency: this is done by the other systems.

Since IMSETY is meant as an interface to scientists, it is necessary for it to be easily understood by end users with very little knowledge of satellite and software technology.

2.4 User characteristics

There will be three groups of users using IMSETY: scientists, observers and administrators.

Scientist characteristics  A scientist is a client of ISIS executing an experiment in a micro-gravity environment provided by a satellite. The interaction with this experiment will be provided by IMSETY. To summarize, a scientist must be able to:

1. compose an experiment from a list of commands,
2. schedule experiments,
3. interact with these executing experiments in real time (when possible),
4. process the results of experiments, in any of their appearances, and
5. compare these results with the results of reference experiments on Earth.

One should bear in mind that a scientist has little or no knowledge of satellite communication, therefore the interface should provide a high level of abstraction from this communication.

Observer characteristics  An observer is invited by a scientist or by an administrator to watch the results of an experiment, either when they are conducted or afterwards. To this end, an observer must be able to process the results of experiments in any of their appearances and compare these results, but they only have read access to the complete system.

Administrator characteristics  An administrator is responsible for the state of the system. To this end, he or she must be able to register users, satellites and payloads with the system and modify, retrieve and delete the information stored about these entities.

2.4.1 User rights

In table 2.1 the rights of the different users are defined. These rights are defined only on those entities to which a specific user has access, and not all entities of a specific class in the system. For example, a scientist can retrieve payload information, but only those payloads for which an administrator has granted him rights.

In the table the following abbreviations are used:

- C  Allowed to create the specified entity.
- R  Allowed to retrieve information about an entity.
- U  Allowed to update information about an entity.
- D  Allowed to delete the specified entity.
2.5 Environment description

In this section, we shall look at the environment in which IMSETY is expected to operate. We first look at the domain of the software, in which we consider IMSETY itself to be a black box.

The domain is the area of business and technology in which a system will operate. In the case of IMSETY (hereafter, the system), the main application domain is that of satellite-based experiments.

The main users for the system will be the scientists conducting the experiments. In order to be able to conduct microgravity experiments, a sample that is to be observed and the hardware to manipulate this sample, has to be placed in a satellite. This setup of hardware and sample is called a payload. In addition to this payload in a satellite, a scientist has the opportunity to place a similar payload on Earth, as a “reference payload”.

**Experiments** After the satellite containing the payload has been launched, a scientist can define experiments on the payload which are then uploaded to the satellite. These experiments are composed by making a sequence of one or more commands, with or without pauses in between. These commands modify parameters of the payload, such as “Set temperature to 300 Kelvin” or “Start the camera”. After an experiment has been defined, it can be scheduled to run at a predefined moment. When this scheduling has been confirmed by the scientist, the system tries to upload it to the satellite at the earliest possible moment. After an experiment has been uploaded, and before it has been executed, a scientist is able to edit or remove this experiment, provided there is still opportunity to communicate with the satellite before the start of the experiment. When defining an experiment, a scientist can opt to reserve time to observe the data gathered by the payload’s sensors in real time. During this real time observation, he can intercede in the experiment by stopping the experiment and redefining it. Then, he can resume the experiment and continue observation.

An administrator determines per satellite whether one or more experiments are allowed simultaneously. This option for more experiments is currently reserved for the specific case that the satellite is the Earth.

After an experiment has been executed, a scientist can retrieve the information gathered, whether it consists of continuous sensor measurements or high volume data like pictures, at a delayed rate.

These observations are first sorted by a preprocessor, which determines what kind of data an observation is. This data is then represented by a representer, which leads to a representation which can be postprocessed. This postprocessing is for example applying an unsharpen mask or a red filter to pictures.

Next to conducting an experiment on a single payload, a scientist can also choose to apply the same experiment to a reference payload and observe the results in stereo, to observe differences between experiments in microgravity conditions and experiments in normal conditions.

A scientist could invite an observer to watch these observations with him. An observer only has access to this single experiment and the observations and measurements it produces, and has
Communication  Up until now, we have paid little notion to the entities and actions involved with the communication of IMSETY with satellites and the payloads contained therein. The diagram shown in figure 2.2 fixes this deficiency by displaying the lines of connection to IMSETY and the systems IMSETY communicates with. We will shortly describe the entities depicted in the diagram, excluding the users and the satellite.

The Mission Control Software (MCS) is a piece of software tailored to each specific satellite. It does several things which are of interest to IMSETY:

• compile commands into data which is understandable by the satellite for which it is created,
• gather data sent by the satellite, and
• check for the consistency of received data and cope with data corruption.

Thus, IMSETY sends commands to the MCS and, on request, receives satellite data from it.

To communicate with the satellite in orbit, a ground station is used. Since a privately owned ground station offers limited communication opportunities, IMSETY shall use the abstract ground station provided by GENSO to communicate with satellites. To communicate through GENSO, one must first book a pass. A pass can be either one-way, providing only downlink communication, or two-way, also providing uplink communication with the satellite. IMSETY shall negotiate the booking of passes with GENSO through a Mission Control Client (MCC) provided by GENSO. When such a booked pass is available, IMSETY shall use the MCS to send commands to or receive data from the satellite through this MCC. Since a satellite can only be communicated with during a pass, IMSETY shall have to take this fact into account when providing scientists the opportunity to schedule an experiment during which real time communication of the satellite with the system is required.

The diagram in figure 2.3 depicts the entities described in this section, along with the relations between them. This diagram is modeled following the UML standard, as explained in [2]. In addition, it also depicts an administrator entity, which is a user maintaining the system by registering payload information, defining commands, adding user accounts and registering satellites.
It is worth mentioning that, in order to cope with reference experiments on Earth, we consider it as a satellite. The association between the command and payload entities represents that for each payload a list of commands should be defined by an administrator. These commands may then be used for composing experiments. An observation can also be just a measurement, which means that no pre- or postprocessing can be applied to the observation data.

Figure 2.3: Exploratory domain model

### 2.6 Assumptions and dependencies

The following assumptions were made when drawing up the specific requirements:

- ISIS is responsible for the consistency of payload and satellite information entered in IMSETY in combination with either or both of the MCC and the MCS,
- satellites will be registered with GENSO by ISIS,
- on request, an MCC shall provide satellite and pass information,
- when requested to, an MCC shall book passes requested by IMSETY,
- an MCS shall send the commands provided by IMSETY to a satellite,
• scientists are cooperative in the scheduling of experiments and will not consciously hinder each other’s experiments, and

• data gathered by IMSETY is not inconsistent across the different systems.

IMSETY depends on other systems: the MCSes and GENSO’s MCC. Since an MCS is made to-spec for a specific satellite, the interface used to communicate with IMSETY is undefined. This means the communications interface used at IMSETY’s side of the communication cannot assume anything about protocols used.

As GENSO’s systems are more defined, IMSETY shall need to communicate with a predefined interface to the MCC. This MCC is currently under development and a specific interface is not yet available. A description of the MCC can be found in the report written by Kasper Revsbech et al. [3]. In this report, it is defined that there will be one MCC per satellite. For IMSETY, it is important that the MCC provides an interface to the GENSO booking mechanism and an abstract connection with the satellite.
Chapter 3

Specific requirements

3.1 Capability requirements

3.1.1 General requirements

URCAR1 Must have
A user shall have an account.

URCAR2 Must have
An account is either an administrator account, a scientist account or an observer account.

URCAR3 Must have
IMSETY shall be a multi-user system.

URCAR4 Must have
IMSETY shall require users to be logged in before they can use any of the system’s functionality.

URCAR5 Must have
IMSETY shall not allow administrators and scientists to be logged in more than once.

URCAR6 Should have
IMSETY shall include a demonstration experiment.

URCAR7 Must have
A user shall only be able to use those parts of the system to which he is authorized.

URCAR8 Could have
Observers shall not be authorized to manipulate experiments.

URCAR9 Must have
IMSETY shall adhere to table 2.1 for user rights.

URCAR10 Must have
IMSETY shall allow scientists and administrators to retrieve all logged communications, according to user rights.

3.1.2 Administration

Administrator

URCAR11 Must have
An administrator shall be able to register a satellite with the system.

URCAR12 Must have
An administrator shall be able to retrieve the information of a registered satellite.

URCAR13 Must have
An administrator shall be able to update the information of a registered satellite.

URCAR14 Must have
An administrator shall be able to unregister a satellite with the system.
### CHAPTER 3. SPECIFIC REQUIREMENTS

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>URCAR15</td>
<td>Must have</td>
<td>An administrator shall be able to register a payload with the system.</td>
</tr>
<tr>
<td>URCAR16</td>
<td>Must have</td>
<td>An administrator shall be able to configure whether multiple experiments may be executed simultaneously on a particular satellite.</td>
</tr>
<tr>
<td>URCAR17</td>
<td>Must have</td>
<td>An administrator shall be able to retrieve the information of a registered payload.</td>
</tr>
<tr>
<td>URCAR18</td>
<td>Must have</td>
<td>An administrator shall be able to update the information of a registered payload.</td>
</tr>
<tr>
<td>URCAR19</td>
<td>Must have</td>
<td>At any moment, only one administrator is allowed to update the information of a single payload registration.</td>
</tr>
<tr>
<td>URCAR20</td>
<td>Must have</td>
<td>An administrator shall be able to unregister a payload with the system.</td>
</tr>
<tr>
<td>URCAR21</td>
<td>Must have</td>
<td>An administrator shall be able to link a registered payload to exactly one satellite.</td>
</tr>
<tr>
<td>URCAR22</td>
<td>Must have</td>
<td>IMSETY shall allow an administrator to create an account.</td>
</tr>
<tr>
<td>URCAR23</td>
<td>Must have</td>
<td>IMSETY shall allow an administrator to retrieve account information.</td>
</tr>
<tr>
<td>URCAR24</td>
<td>Must have</td>
<td>IMSETY shall allow an administrator to update account information.</td>
</tr>
<tr>
<td>URCAR25</td>
<td>Must have</td>
<td>IMSETY shall allow an administrator to delete an account.</td>
</tr>
<tr>
<td>URCAR26</td>
<td>Must have</td>
<td>IMSETY shall allow an administrator to configure which users have particular rights to which payloads.</td>
</tr>
<tr>
<td>URCAR27</td>
<td>Must have</td>
<td>IMSETY shall allow an administrator to set up possible payload specific commands through a plain text file.</td>
</tr>
<tr>
<td>URCAR28</td>
<td>Could have</td>
<td>IMSETY shall allow an administrator to set up possible satellite commands through an interface.</td>
</tr>
<tr>
<td>URCAR29</td>
<td>Should have</td>
<td>IMSETY shall allow an administrator to configure all plug-ins for the system.</td>
</tr>
</tbody>
</table>

#### 3.1.3 Experiment scheduling

**Scientist**

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>URCAR30</td>
<td>Must have</td>
<td>IMSETY shall allow scientists to retrieve experiment details.</td>
</tr>
<tr>
<td>URCAR31</td>
<td>Must have</td>
<td>IMSETY shall communicate a list of payload specific commands to the scientist.</td>
</tr>
<tr>
<td>URCAR32</td>
<td>Must have</td>
<td>IMSETY shall let scientists select commands from this list and thereby compose an experiment.</td>
</tr>
<tr>
<td>URCAR33</td>
<td>Must have</td>
<td>IMSETY shall not allow scientists to type in commands directly.</td>
</tr>
<tr>
<td>URCAR34</td>
<td>Must have</td>
<td>A scientist shall be able to request the scheduling of the execution of experiments on a predefined moment.</td>
</tr>
</tbody>
</table>
### CHAPTER 3. SPECIFIC REQUIREMENTS

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>URCAR35</td>
<td>Must have</td>
<td>A scientist shall be able to request an experiment, which has not yet been executed, to be unscheduled.</td>
</tr>
<tr>
<td>URCAR36</td>
<td>Could have</td>
<td>A scientist shall be able to request to change an experiment, before it is being executed.</td>
</tr>
<tr>
<td>URCAR37</td>
<td>Must have</td>
<td>A scientist shall be able to send commands to an experiment already executing.</td>
</tr>
<tr>
<td>URCAR38</td>
<td>Must have</td>
<td>At any moment, only one scientist is allowed to compose an experiment on a single payload.</td>
</tr>
</tbody>
</table>

#### System

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>URCAR39</td>
<td>Must have</td>
<td>The system shall upload experiments scheduled and queued for upload to a satellite whenever there is a window of opportunity.</td>
</tr>
<tr>
<td>URCAR40</td>
<td>Must have</td>
<td>The system shall download high volume data requested by scientists from the satellite whenever there is a window of opportunity.</td>
</tr>
</tbody>
</table>

#### 3.1.4 Experiment observation and processing

##### Scientist, observer

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>URCAR41</td>
<td>Must have</td>
<td>Experiment data can be retrieved from the system.</td>
</tr>
<tr>
<td>URCAR42</td>
<td>Must have</td>
<td>Experiment data can be observed in real time while the experiment is being executed during a window of opportunity.</td>
</tr>
<tr>
<td>URCAR43</td>
<td>Must have</td>
<td>IMSETY will support the visualization of video data.</td>
</tr>
<tr>
<td>URCAR44</td>
<td>Must have</td>
<td>IMSETY will support the visualization of picture data.</td>
</tr>
</tbody>
</table>

##### Observer

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>URCAR47</td>
<td>Could have</td>
<td>Observers are able to see representations.</td>
</tr>
</tbody>
</table>

##### Scientist

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>URCAR48</td>
<td>Must have</td>
<td>Observation data can only be gathered during an experiment.</td>
</tr>
<tr>
<td>URCAR49</td>
<td>Must have</td>
<td>Collected observation data can be exported from the system to several formats.</td>
</tr>
<tr>
<td>URCAR50</td>
<td>Should have</td>
<td>Periods can be marked during real-time observation.</td>
</tr>
<tr>
<td>URCAR51</td>
<td>Should have</td>
<td>IMSETY shall allow the requesting of downloading of high resolution images from these marked periods.</td>
</tr>
</tbody>
</table>
CHAPTER 3. SPECIFIC REQUIREMENTS

URCAR52
A plug-in system for data preprocessing shall be provided.  

URCAR53
A plug-in providing an example for data preprocessing will be provided.  

URCAR54
A plug-in providing an example of representation postprocessing will be provided.  

URCAR55
The system shall show thumbnails provided by a satellite.  

URCAR56
IMSETY shall allow a scientist to request high resolution versions of the images represented by these thumbnails to be downloaded.  

URCAR57
IMSETY shall allow the requesting of deletion of high resolution images from the satellite memory.  

3.1.5 Communication

URCAR58
IMSETY shall represent a timeline of one-way and two-way communication opportunities.  

URCAR59
IMSETY shall provide the possibility to book one-way communications.  

URCAR60
IMSETY shall provide the possibility to book two-way communications.  

URCAR61
IMSETY shall communicate failure of satellite communication to the scientist.  

URCAR62
IMSETY shall communicate failure of communication with GENSO to the scientist.  

URCAR63
IMSETY shall communicate the rejection of bookings to the scientist.  

URCAR64
IMSETY shall log all communications with satellites.  

URCAR65
IMSETY shall log all communications with the MCCs.  

URCAR66
IMSETY shall log all communications with the MCSes.  

3.2 Constraint requirements

URCOR1
The system shall interface with GENSO through an MCC for purposes of booking.  

URCOR2
The system shall interface with an MCS for communication with satellites.  

URCOR3
A protocol shall be defined for interfacing with the MCS based on a list of commands provided by the customer.  

URCOR4
The interface of the system shall be accessible on Windows 2000/XP/Vista, Mac OS X, Sun Solaris and Linux.  

URCOR5
IMSETY’s backend shall run on a Linux server.
Appendix A

Use cases

The following use cases describe sequences of actions that an actor can perform in order to complete a given task. The use cases are grouped per actor. For the layout of the use cases we used the one proposed in [2].

A.1 User

A.1.1 Log on to the system

Goals: To log on to the system.
Preconditions: One account for the user is registered with the system.
Summary: Enter credentials and log on to the system.
Related use cases: None.
Priority: Must have.
Steps:

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Choose to log on to the system</td>
<td>2. Display a view for entering credentials.</td>
</tr>
<tr>
<td>3. Enter credentials.</td>
<td>5. Log on to the system with the given credentials.</td>
</tr>
<tr>
<td>4. Choose to log on.</td>
<td></td>
</tr>
</tbody>
</table>

A.1.2 Fail to log on to the system

Goals: To log on to the system.
Preconditions: None
Summary: Enter credentials and try to log on to the system, which fails.
Related use cases: Extension of: A.1.1 (extension point: step 4: Choose to log on)
Priority: Must have.
Steps:

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>4. Choose to log on.</td>
<td>5. Credentials are incorrect.</td>
</tr>
<tr>
<td>Return to step 2.</td>
<td></td>
</tr>
</tbody>
</table>

A.1.3 Log out

Goals: To log out.
Preconditions: The user is logged on to the system.
Summary: Choose to log out.
Related use cases: None.
APPENDIX A. USE CASES

Priority: Must have.

Steps:

- **Actor actions**
- **System responses**
  - 1. Choose to log out
  - 2. Log out

A.1.4 Retrieve payload logs

Goals: To retrieve the logs of a payload.

Preconditions: None

Summary: Select a registered payload and retrieve its logs.

Related use cases:

- Includes: [4.2.6]

Priority: Must have.

Steps:

- **Actor actions**
- **System responses**
  - 1. Retrieve payload details (included use case).
  - 2. Choose to retrieve its logs.
  - 3. Display payload logs.

A.2 Administrator

A.2.1 Register a satellite

Goals: To register a satellite with the system.

Preconditions: None.

Summary: Enter the details for the satellite and register it with the system.

Related use cases: None.

Priority: Must have.

Steps:

- **Actor actions**
- **System responses**
  - 1. Open ‘Satellites’ view.
  - 2. Display a list of all registered satellites.
  - 3. Choose to register a satellite.
  - 4. Display a view for entering satellite details.
  - 5. Enter details for the satellite.
  - 6. Choose to register the satellite.
  - 7. Register the satellite.

A.2.2 Retrieve satellite details

Goals: To retrieve a list of satellite details of a registered satellite.

Preconditions: There exists at least one registered satellite in the system.

Summary: Select and retrieve the details of a registered satellite.

Related use cases: None.

Priority: Must have.

Steps:

- **Actor actions**
- **System responses**
  - 1. Open ‘Satellites’ view.
  - 2. Display a list of all registered satellites.
  - 3. Select a satellite.
  - 4. Choose to retrieve its details.
  - 5. Display satellite details.

A.2.3 Update satellite details

Goals: To update the details of a registered satellite.

Preconditions: There exists at least one registered satellite in the system.

Summary: Select a registered satellite and update its details.

Related use cases:
APPENDIX A. USE CASES

Includes: A.2.2
Priority: Must have.

Steps:

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Retrieve satellite details (included use case).</td>
<td></td>
</tr>
<tr>
<td>2. Choose to update its details.</td>
<td></td>
</tr>
<tr>
<td>3. Display a view for editing satellite details.</td>
<td></td>
</tr>
<tr>
<td>4. Update its details.</td>
<td></td>
</tr>
<tr>
<td>5. Choose to update the details.</td>
<td></td>
</tr>
<tr>
<td>6. Update the satellite details.</td>
<td></td>
</tr>
</tbody>
</table>

A.2.4 Unregister a satellite

Goals: To unregister a registered satellite.
Preconditions: There exists at least one registered satellite with no registered payloads in the system.
Summary: Select a registered satellite and unregister it.
Related use cases: None.
Priority: Must have.
Steps:

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Open ‘Satellites’ view.</td>
<td></td>
</tr>
<tr>
<td>2. Display a list of all registered satellites.</td>
<td></td>
</tr>
<tr>
<td>3. Select a satellite.</td>
<td></td>
</tr>
<tr>
<td>4. Choose to unregister it.</td>
<td></td>
</tr>
<tr>
<td>5. Unregister the satellite.</td>
<td></td>
</tr>
</tbody>
</table>

A.2.5 Register a payload

Goals: To register a payload.
Preconditions: There exists at least one registered satellite in the system.
Summary: Select a registered satellite and register a new payload by filling out its details.
Related use cases:
Includes: A.2.2
Priority: Must have.
Steps:

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Retrieve satellite details (included use case).</td>
<td></td>
</tr>
<tr>
<td>2. Choose to register a payload.</td>
<td></td>
</tr>
<tr>
<td>3. Display a view for entering payload details.</td>
<td></td>
</tr>
<tr>
<td>4. Enter details for the payload.</td>
<td></td>
</tr>
<tr>
<td>5. Choose to register the payload.</td>
<td></td>
</tr>
<tr>
<td>6. Register the payload.</td>
<td></td>
</tr>
</tbody>
</table>

A.2.6 Retrieve payload registration

Goals: To retrieve payload details.
Preconditions: There exists at least one payload and related satellite registration in the system.
Summary: Select payload registration and choose to retrieve the details of a certain payload.
Related use cases:
Includes: A.2.2
Priority: Must have.
Steps:

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Retrieve satellite details (included use case).</td>
<td></td>
</tr>
<tr>
<td>2. Select a payload.</td>
<td></td>
</tr>
<tr>
<td>3. Choose to retrieve its details</td>
<td></td>
</tr>
<tr>
<td>4. Display details of the payload.</td>
<td></td>
</tr>
</tbody>
</table>
A.2.7 Update a payload registration

**Goals:** To update the details of a payload registration.

**Preconditions:** There exists at least one payload and related satellite registration in the system.

**Summary:** Select payload registration and update its details.

**Related use cases:**

- Includes: A.2.6

**Priority:** Must have.

**Steps:**

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Retrieve payload details (included use case).</td>
<td>3. Display a view for editing payload details.</td>
</tr>
<tr>
<td>2. Choose to update its details.</td>
<td>4. Update details for the payload registration.</td>
</tr>
<tr>
<td>5. Choose to update the details.</td>
<td>6. Update the payload registration.</td>
</tr>
</tbody>
</table>

A.2.8 Unregister a payload registration

**Goals:** To unregister a payload registration.

**Preconditions:** There exists at least one payload and related satellite registration in the system.

**Summary:** Select and unregister a payload registration.

**Related use cases:**

- Includes: A.2.2

**Priority:** Must have.

**Steps:**

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Retrieve satellite details (included use case).</td>
<td>3. Choose to unregister it.</td>
</tr>
<tr>
<td>2. Select a payload registration.</td>
<td>4. Unregister the payload registration.</td>
</tr>
</tbody>
</table>

A.2.9 Create an account for a user

**Goals:** To create an account for a user.

**Preconditions:** None.

**Summary:** Enter rights for the user and create an account.

**Related use cases:** None.

**Priority:** Must have.

**Steps:**

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Open ‘Users’ view.</td>
<td>2. Display a list of all registered users.</td>
</tr>
<tr>
<td>3. Choose to create an account for a user.</td>
<td>4. Display a view for entering access rights.</td>
</tr>
<tr>
<td>5. Enter rights for the user.</td>
<td>7. Create the account.</td>
</tr>
</tbody>
</table>

A.2.10 Retrieve account information of a user

**Goals:** To retrieve account information of a user.

**Preconditions:** There exists at least one account of a user in the system.

**Summary:** Select a user and retrieve his information.

**Related use cases:** None.

**Priority:** Must have.

**Steps:**
### A.2.11 Modify linked payload registrations with a scientist or observer account

**Goals:** To modify the linked payload registrations with a scientist account.

**Preconditions:** There exists at least one account of a scientist or observer and one payload registration in the system.

**Summary:** Select a scientist or observer account and modify the linked payload registrations with it.

**Related use cases:** A.2.10.

**Priority:** Must have.

**Steps:**

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Retrieve account information of a user (included use case).</td>
<td>2. Display a list of all linked payload registrations with the account.</td>
</tr>
<tr>
<td>3. Choose to link payload registrations with the account.</td>
<td>4. Choose to modify the links.</td>
</tr>
<tr>
<td>5. Display the links.</td>
<td>6. Choose to save the link modifications.</td>
</tr>
<tr>
<td>7. Choose to save the link modifications.</td>
<td>8. Save the link modifications.</td>
</tr>
</tbody>
</table>

### A.2.12 Update the account information of a user

**Goals:** To update the account information of a user.

**Preconditions:** There exists at least one account of a user in the system.

**Summary:** Select a user and update his information.

**Related use cases:** Includes: A.2.10

**Priority:** Must have.

**Steps:**

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Retrieve user details (included use case).</td>
<td>2. Display a view for editing account information.</td>
</tr>
<tr>
<td>3. Choose to update his account information.</td>
<td>4. Update account information.</td>
</tr>
<tr>
<td>5. Choose to update the account information.</td>
<td>6. Update the account of the user.</td>
</tr>
</tbody>
</table>

### A.2.13 Delete the account of a user

**Goals:** To delete the account of a user.

**Preconditions:** There exists at least one account of a user in the system.

**Summary:** Select a user and delete his account.

**Related use cases:** None.

**Priority:** Must have.

**Steps:**

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Open ‘Users’ view.</td>
<td>2. Display a list of all registered users.</td>
</tr>
<tr>
<td>3. Select a user.</td>
<td>4. Choose to delete his account.</td>
</tr>
<tr>
<td>5. Delete the account of the user.</td>
<td></td>
</tr>
</tbody>
</table>
A.2.14 Set up possible payload commands

**Goals:** To set up a set of possible commands for a registered payload.
**Preconditions:** There is at least one registered payload.
**Summary:** Modify the payload commands.

**Related use cases:**
- Includes: A.2.6
- Generalization of: A.2.15, A.2.16
**Priority:** Must have.

**Steps:**

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Retrieve payload details (included use case).</td>
<td>2. Open ‘Command view’.</td>
</tr>
<tr>
<td>3. Show commands for selected payload.</td>
<td></td>
</tr>
<tr>
<td>4. Modify the payload commands.</td>
<td>5. Choose to save the list of payload commands.</td>
</tr>
<tr>
<td>6. Save the list of payload commands.</td>
<td></td>
</tr>
</tbody>
</table>

A.2.15 Set up possible payload commands through a plain text file

**Goals:** To set up a set of possible commands for a registered payload.
**Preconditions:** There is at least one registered payload.
**Summary:** Import a configuration file with possible commands.

**Related use cases:**
- Includes: A.2.6
- Generalization: A.2.14
**Priority:** Must have.

**Steps:**

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Retrieve payload details (included use case).</td>
<td>2. Open ‘Command view’.</td>
</tr>
<tr>
<td>3. Show commands for selected payload.</td>
<td></td>
</tr>
<tr>
<td>4. Choose to import command list.</td>
<td>5. Show file selector.</td>
</tr>
<tr>
<td>6. Select file to be imported.</td>
<td>7. Import commands.</td>
</tr>
<tr>
<td>8. Choose to save the list of payload commands.</td>
<td>9. Save the list of payload commands.</td>
</tr>
</tbody>
</table>

A.2.16 Set up possible payload commands through an interface

**Goals:** To set up a set of possible commands for a registered payload.
**Preconditions:** There is at least one registered payload.
**Summary:** Edit a set of possible commands for a registered payload.

**Related use cases:**
- Includes: A.2.6
- Generalization: A.2.14
**Priority:** Could have.

**Steps:**
APPENDIX A. USE CASES

Actor actions                     System responses
1. Retrieve payload details (included use case).
2. Open ‘Command view’.
3. Show commands for selected payload.
4. Choose to edit command list.
5. Display command editor.
6. Edit commands.
7. Choose to save the list of payload commands.
8. Save the list of payload commands.

A.3 Scientist

A.3.1 Retrieve observations of a payload

Goals: To retrieve observations of a payload.
Preconditions: There exists at least one payload and related satellite registration in the system.
Summary: Select a payload and retrieve its observations.
Related use cases:
   Includes: A.2.2
   Generalization of: A.3.5
Priority: Must have.
Steps:
   Actor actions                     System responses
   1. Retrieve satellite details (included use case).
   2. Select a payload.
   3. Choose to retrieve its observations.
   4. Display the list of observations of the payload.

A.3.2 Retrieve a high-resolution picture of an observation

Goals: To retrieve a high-resolution picture of an observation.
Preconditions: There exists at least one payload and satellite registration in the system.
Summary: Select a payload and retrieve a high-resolution picture of an observation.
Related use cases:
   Includes: A.3.1
Priority: Must have.
Steps:
   Actor actions                     System responses
   1. Retrieve observations of a payload (included use case).
   2. Select a thumbnail.
   3. Choose to retrieve the high-resolution picture.
   4. Schedule to retrieve and store the high-resolution picture.

A.3.3 Retrieve an observation in real time

Goals: To retrieve a data stream of an observation.
Preconditions: There exists at least one payload and satellite registration in the system and we are in a window of opportunity.
Summary: Select a payload and retrieve a data stream of an observation.
Related use cases:
   Includes: A.2.6
Priority: Must have.
A.3.4 Retrieve a video stream of an observation in real time

Goals: To retrieve a video stream of an observation.
Preconditions: There exists at least one payload and satellite registration in the system and we are in a window of opportunity.
Summary: Select a payload and retrieve a video stream of an observation.

Related use cases:
- Generalization: A.3.3
- Includes: A.2.6

Priority: Must have.

Steps:

1. Retrieve a payload (included use case).
2. Select to retrieve data stream.

A.3.5 Retrieve high-resolution pictures of a selection of a video stream

Goals: To retrieve high-resolution pictures of a selection of a video stream.
Preconditions: None.
Summary: Select a period of a video stream and download one or more high-resolution pictures of this period.

Related use cases:
- Includes: A.3.1

Priority: Should have.

Steps:

1. Retrieve observations of a payload (included use case).
2. Select a video observation.
3. Choose to retrieve the video observation details.
4. Display video observation details.
5. Choose to retrieve high-resolution pictures.
6. Display a view for selecting a period of the video.
7. Select a period of the video.
8. Display thumbnails of pictures made in this period.
9. Select one or more thumbnails.
10. Choose to retrieve the high-resolution picture(s).
11. Schedule to retrieve and save the high-resolution picture(s).

A.3.6 Postprocess observation visualizations

Goals: To postprocess observation visualizations.
Preconditions: At least one observation is visualized.
Summary: A visualization is selected and processed.

Related use cases:
- Includes: A.3.2, A.3.5 and A.3.3

Priority: Should have.

Steps:
APPENDIX A. USE CASES

Actor actions | System responses
---|---
1. Retrieve high resolution picture of an observation (included use case [A.3.2]) or retrieve high resolution pictures of a selection of a video (included use case [A.3.5]) or retrieve observation stream in real time (included use case [A.3.3]).
2. Select ‘Postprocessing’ view.
3. Show possible postprocessors.
4. Choose postprocessor.
5. Apply processor to observation.
6. Repeat steps 4-5, as needed.
7. Choose to close view.

A.3.7 Compare two observations

Goals: To compare two payload observations.
Preconditions: There exists at least one payload and satellite registration in the system.
Summary: Select observations of two payloads and compare them.

Related use cases:
- Extends: [A.3.1]
- Priority: Should have.

Steps:

Actor actions | System responses
---|---
1. Retrieve observations of a payload (included use case).
2. Choose to compare observations.
3. Display a list of authorized payload registrations.
4. Select a payload.
5. Choose to compare with this payload.
6. Display the list of observations of the payload.
7. Select an observation.
8. Display the two observations simultaneously.

A.3.8 Export observation data

Goals: To export the observation data to an other file format.
Preconditions: The selected observation data is exportable and the experiment is finished.
Summary: Select observation data and export it.

Related use cases:
- Includes: [A.3.1]
- Priority: Should have.

Steps:

Actor actions | System responses
---|---
1. Retrieve observations of a payload (included use case).
2. Choose to export observation data.
3. Display export options.
4. Select options.
5. Choose to export the observation data.
6. Export the data.

A.3.9 Retrieve list of passes

Goals: To get a list of passes.
Preconditions: There exists at least one registered satellite in the system.
Summary: Select a satellite and an interval to retrieve a list of all passes in that interval.

Related use cases:
- Includes: [A.2.6]
APPENDIX A. USE CASES

Priority: Must have.

Steps:

Actor actions System responses
1. Retrieve payload registration (included use case).
2. Choose to retrieve a list of passes. 3. Display a view for entering an interval.
4. Enter an interval.
5. Choose to list all passes of the satellite in the given interval.
6. List all passes of the selected satellite in the given interval.

A.3.10 Program experiment

Goals: To program an experiment.

Preconditions:
Summary: Add one or more commands to an experiment.
Related use cases: Includes: A.2.6
Priority: Must have.

Steps:

Actor actions System responses
1. Retrieve payload registration (included use case).
2. Choose to add an experiment. 3. Display possible commands.
4. Select a command. 5. Display a view for the details of the commands.
6. Enter command details. 7. Add the command.
8. Repeat steps 4-7, as needed. 9. Create the experiment.

A.3.11 Retrieve experiment details

Goals: To retrieve the details of an already programmed experiment.

Preconditions: There has already been at least one experiment programmed.
Summary: Select an experiment and retrieve its details.
Related use cases: Includes: A.2.6
Priority: Must have.

Steps:

Actor actions System responses
1. Retrieve payload registration (included use case).
2. Select an experiment. 3. Display experiment details.
3. Choose to retrieve its details. 4. Display experiment details.

A.3.12 Schedule experiment

Goals: To schedule an already programmed experiment.

Preconditions: There has already been at least one experiment programmed.
Summary: Select an experiment and schedule it.
Related use cases: Includes: A.3.11
Priority: Must have.

Steps:
### APPENDIX A. USE CASES

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Retrieve experiment details (included use case).</td>
<td>3. Display a view for the details of the schedule.</td>
</tr>
<tr>
<td>2. Choose to schedule the experiment.</td>
<td>6. Schedule the experiment.</td>
</tr>
<tr>
<td>4. Enter schedule details.</td>
<td>5. Choose to schedule the experiment.</td>
</tr>
</tbody>
</table>

#### A.3.13 Update an experiment

**Goals:** To update an experiment.

**Preconditions:** There is at least one experiment programmed.

**Summary:** Modify the course of an experiment by updating its commands.

**Related use cases:**
- **Includes:** [A.3.11]

**Priority:** Must have.

**Steps:**

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Retrieve experiment details (included use case).</td>
<td>3. Display a view for the details of the experiment.</td>
</tr>
<tr>
<td>2. Choose to update the experiment.</td>
<td>6. Update the experiment.</td>
</tr>
<tr>
<td>4. Update experiment details.</td>
<td>5. Choose to update this experiment.</td>
</tr>
</tbody>
</table>

#### A.3.14 Fail to update an experiment

**Goals:** To update an experiment.

**Preconditions:** There is at least one not yet executing experiment.

**Summary:** Modify the course of the experiment by updating its commands, which fails.

**Related use cases:**
- **Extension of:** [A.3.13] (extension point: step 5: Choose to update this experiment)

**Priority:** Must have.

**Steps:**

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>5. Choose to update this experiment.</td>
<td>6. No window of opportunity before scheduled start to upload experiment.</td>
</tr>
</tbody>
</table>

#### A.3.15 Intervene in an executing experiment

**Goals:** To intervene in an executing experiment.

**Preconditions:** There is at least one uploaded and executing experiment and there is a window of opportunity to communicate with the satellite on which this experiment is executing.

**Summary:** Send commands to an already executing experiment.

**Related use cases:**
- **Includes:** [A.3.11]

**Priority:** Must have.

**Steps:**

---

*URD 1.2* 25
APPENDIX A. USE CASES

Actor actions                                System responses
1. Retrieve experiment details (included use case).
2. Choose to intervene in the experiment. 3. Display a view for entering satellite commands.
4. Enter satellite commands.
5. Choose to send commands to the satellite. 6. Send commands to the satellite.

A.3.16 Delete an experiment

Goals: To delete an experiment.
Preconditions: There is at least one experiment programmed.
Summary: Select and delete an experiment.
Related use cases:
  Includes: A.3.11
Priority: Must have.
Steps:
  Actor actions                                System responses
  1. Retrieve experiment details (included use case).
  2. Choose to delete the experiment. 3. Delete the experiment.

A.4 Observer

A.4.1 Observe an experiment

Goals: To observe an experiment.
Preconditions: There is at least one experiment programmed which the observer may observe.
Summary: Select and observe an experiment.
Related use cases:
  Includes: A.3.11
Priority: Could have.
Steps:
  Actor actions                                System responses
  1. Retrieve experiment details (included use case).
  2. Choose to observe the experiment. 3. Display the observations of the payload on which the experiment is executed.

A.5 System

A.5.1 Connect with GENSO

Goals: To connect with GENSO.
Preconditions: None.
Summary: The system makes a connection with GENSO.
Related use cases: None.
Priority: Must have.
Steps:
  Actor actions                                GENSO responses
  1. Try to connect with GENSO. 2. Establish connection.
A.5.2 Fail to connect with GENSO

Goals: To connect with GENSO.
Preconditions: The system may retry \( i \) times to connect with GENSO.
Summary: The system tries to connect with GENSO, which fails.
Related use cases:
- Extension of: [A.5.1] (extension point: step 1: Try to connect with GENSO)
Priority: Must have.
Steps:

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>GENSO responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Try to connect with GENSO.</td>
<td>2. None.</td>
</tr>
<tr>
<td>3. Repeat ( i ) times step 1-2.</td>
<td></td>
</tr>
<tr>
<td>4. Log and inform user of connection failure.</td>
<td></td>
</tr>
</tbody>
</table>

A.5.3 Connect with MCS

Goals: To connect with MCS.
Preconditions: None.
Summary: The system makes a connection with MCS.
Related use cases: None.
Priority: Must have.
Steps:

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>MCS responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Try to connect with MCS.</td>
<td>2. Establish connection.</td>
</tr>
</tbody>
</table>

A.5.4 Fail to connect with MCS

Goals: To connect with MCS.
Preconditions: The system may retry \( i \) times to connect with MCS.
Summary: The system tries to connect with MCS, which fails.
Related use cases:
- Extension of: [A.5.3] (extension point: step 1: Try to connect with MCS)
Priority: Must have.
Steps:

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>MCS responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Try to connect with MCS.</td>
<td>2. None.</td>
</tr>
<tr>
<td>3. Repeat ( i ) times step 1-2.</td>
<td></td>
</tr>
<tr>
<td>4. Log and inform user of connection failure.</td>
<td></td>
</tr>
</tbody>
</table>

A.5.5 Retrieve observations from GENSO

Goals: To retrieve observations from GENSO.
Preconditions: There has already been at least one experiment programmed.
Summary: The system connects with GENSO and queries the observations of a certain experiment.
Related use cases:
- Includes: [A.5.1]
Priority: Must have.
Steps:

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>GENSO responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Connect with GENSO (included use case).</td>
<td></td>
</tr>
<tr>
<td>2. Ask for observations of a certain experiment.</td>
<td>3. Send observations.</td>
</tr>
<tr>
<td>4. Receive and store observations.</td>
<td></td>
</tr>
</tbody>
</table>
A.5.6 Upload experiment to GENSO

Goals: To upload an experiment to GENSO.
Preconditions: There has already been at least one experiment programmed.
Summary: The system connects with GENSO and uploads the experiment.

Related use cases: A.5.1

Priority: Must have.

Steps:

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>GENSO responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Connect with GENSO (included use case).</td>
<td></td>
</tr>
<tr>
<td>2. Upload the experiment.</td>
<td></td>
</tr>
<tr>
<td>3. Send acknowledgement of a successful upload.</td>
<td></td>
</tr>
</tbody>
</table>

A.5.7 Fail to communicate with a satellite

Goals: To communicate with a satellite.
Preconditions: A connection to the corresponding MCS has been established.
Summary: The system tries to communicate with the satellite, which fails.

Priority: Must have.

Steps:

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>MCS responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Upload data to or download data from a satellite.</td>
<td></td>
</tr>
<tr>
<td>2. Send failure message.</td>
<td></td>
</tr>
<tr>
<td>3. Notify user of failure.</td>
<td></td>
</tr>
</tbody>
</table>