Development of a CubeSat communications system based on CCSDS and ECSS standards

Open Source CubeSat Workshop 2018
Madrid 25.09.2018

Milenko Starcik
TU Darmstadt Space Technology e. V.
Our Team
Our Mission
Our CubeSat

1U - CubeSat

Open Source

ECSS & CCSDS Standards

Technology Demonstrator

Reflectarray Antenna
CCSDS and ECSS standards

Global

Blue: Recommended Standards
Red: Draft Recommended Standard
Magenta: Recommended Practices
Green: Informational Reports
## CCSDS and ECSS standards

### OSI Layers

<table>
<thead>
<tr>
<th>OSI Layers</th>
<th>CCSDS Standards</th>
</tr>
</thead>
<tbody>
<tr>
<td>7 Application</td>
<td>Mission Operation (MO) Services</td>
</tr>
<tr>
<td>6 Presentation</td>
<td>Message Abstraction Layer (MAL)</td>
</tr>
<tr>
<td>5 Session</td>
<td>Message Abstraction Layer Space Packets Binding</td>
</tr>
<tr>
<td>4 Transport</td>
<td>Space Packets</td>
</tr>
<tr>
<td>3 Network</td>
<td>Unified Space Data Link</td>
</tr>
<tr>
<td>2 Data Link</td>
<td>Telemetry Synchronisation and Channel Coding</td>
</tr>
<tr>
<td></td>
<td>Telecommand Synchronisation and Channel Coding</td>
</tr>
<tr>
<td>1 Physical</td>
<td>Radio Frequency and Modulation System</td>
</tr>
</tbody>
</table>
User friendly?
## CCSDS and ECSS standards

<table>
<thead>
<tr>
<th>OSI Layers</th>
<th>CCSDS Standards</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>7 Application</td>
<td>Mission Operation (MO) Services</td>
<td>65+62+71+290</td>
</tr>
<tr>
<td>6 Presentation</td>
<td>Message Abstraction Layer (MAL)</td>
<td>178</td>
</tr>
<tr>
<td>5 Session</td>
<td>Message Abstraction Layer Space Packets Binding</td>
<td>72</td>
</tr>
<tr>
<td>4 Transport</td>
<td>Space Packets</td>
<td>49</td>
</tr>
<tr>
<td>3 Network</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 Data Link</td>
<td>Unified Space Data Link</td>
<td>168</td>
</tr>
<tr>
<td></td>
<td>Telemetry Synchronisation and Channel Coding</td>
<td>92</td>
</tr>
<tr>
<td></td>
<td>Telecommand Synchronisation and Channel Coding</td>
<td>43</td>
</tr>
<tr>
<td>1 Physical</td>
<td>Radio Frequency and Modulation System</td>
<td>280</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 1370</td>
</tr>
</tbody>
</table>
Pros and Cons

Pro:
- Everything for free
- Robust
- Cooperation
- Reusability

Contra:
- Difficult to understand
- Overhead
- Unused functionality
Summer of Code in Space 2017

“CCSDS frame exchange over Radio Link in GNURadio”
Problems

Documentation
Version Control
Installation
Bugs
hackRF-one
Keep it simple

Solutions
Telemetry
- Error-control coding
  - Reed-Solomon coding
- Frame validation
- Synchronization
- Pseudo-randomizing
Unified Space Data Link

Physical Channel

Virtual Channels
Unified Space Data Link

```python
class TransferFrame:
    '''
    A Transfer Frame is used to carry the data in the Unified Space Data Link Protocol.
    '''

    def __init__(self,
                 transfer_frame_primary_header,
                 transfer_frame_insert_zone,
                 transfer_frame_data_field,
                 operational_control_field,
                 frame_error_control_field):
        ...

        :param transfer_frame_primary_header: Contains information about the Frame
        :type transfer_frame_primary_header: TransferFramePrimaryHeader
        :param transfer_frame_insert_zone: Is used by the Insert Service
        :type transfer_frame_insert_zone:
        :param transfer_frame_data_field: Contains the data of the Frame
        :type transfer_frame_data_field: TransferFrameDataField
        :param operational_control_field: Provides a mechanism to report some real-time functions
        :type operational_control_field:
        :param frame_error_control_field: Provides the capability to detect errors
        :type frame_error_control_field:
        ...

        self.transfer_frame_primary_header = transfer_frame_primary_header
        self.transfer_frame_insert_zone = transfer_frame_insert_zone
        self.transfer_frame_data_field = transfer_frame_data_field
        self.operational_control_field = operational_control_field
        self.frame_error_control_field = frame_error_control_field
```
Welcome to Unified Space Data Link Protocol’s documentation!

Transfer Frame

class TransferFrame(transfer_frame_primary_header, transfer_frame_insert_zone, transfer_frame_data_field, operational_control_field, frame_error_control_field)

A Transfer Frame is used to carry the data in the Unified Space Data Link Protocol.

| __init__ (transfer_frame_primary_header, transfer_frame_insert_zone, transfer_frame_data_field, operational_control_field, frame_error_control_field) |

Parameters:
- transfer_frame_primary_header (TransferFramePrimaryHeader) – Contains information about the Frame
- transfer_frame_insert_zone – Is used by the Insert Service
- transfer_frame_data_field (TransferFrameDataField) – Contains the data of the Frame
- operational_control_field – Provides a mechanism to report some real-time functions
- frame_error_control_field – Provides the capability to detect errors

Transfer Frame Data Field

class TransferFrameDataField

The Transfer Frame Data Field contains the data that should be delivered by the frame.

| __init__ (transfer_frame_data_field_header, transfer_frame_data_zone) |

Parameters:
- transfer_frame_data_field_header (TransferFrameDataFieldHeader) – Contains information about the Transfer Frame Data Field
- transfer_frame_data_zone (std) – Contains the real data

Transfer Frame Data Field Header

class TransferFrameDataFieldHeader


## OSI Layers

<table>
<thead>
<tr>
<th>OSI Layer</th>
<th>CCSDS Standards</th>
</tr>
</thead>
<tbody>
<tr>
<td>7 Application</td>
<td>Mission Operation (MO) Services</td>
</tr>
<tr>
<td>6 Presentation</td>
<td>Message Abstraction Layer (MAL)</td>
</tr>
<tr>
<td>5 Session</td>
<td>Message Abstraction Layer Space Packets Binding</td>
</tr>
<tr>
<td>4 Transport</td>
<td>Space Packets</td>
</tr>
<tr>
<td>3 Network</td>
<td>Unified Space Data Link</td>
</tr>
<tr>
<td>2 Data Link</td>
<td>Telemetry Synchronisation and Channel Coding</td>
</tr>
<tr>
<td>1 Physical</td>
<td>Radio Frequency and Modulation System</td>
</tr>
</tbody>
</table>
Active Prototypes

Hey all,

I am Milenko from TU Darmstadt Space Technology, a student group from Darmstadt, Germany.

We are currently beginning to develop a CubeSat and a Rocketry programme, which both should be made open source from the very beginning.

Before setting up the KTS and so on, we want to make sure that the licensing is correct.

It would be interesting to us to know how you are managing intellectual property that is produced for the Libre Space Foundation.

Which licenses do you prefer for software and for hardware and why?

An other question is, under which name is the software/hardware published?

Is it enough to set up a repository with the license in it, with our group's name on it, and anyone contributing to it is accepting it that way?

Greetings from Darmstadt,

Milenko

---

Some basic stuff. The larger a project gets it is more common to use several licenses. There are many open-source licenses that a project can pick according to its needs and licensing can be an issue of debate (although we had consensus early on when we started). We wanted to provide our users with a license that will cover the following liberties for our users:

0. freedom to run
1. freedom to study
2. freedom to share
3. freedom to modify

We wanted a license that has a strong copyleft, meaning that if you do modify our source code and distribute it (or use it on your website for the LGPL 3.0 parts) you should share your modifications with the same license.

GNU GPL-3 for software. This license is a copyleft. Keep in mind that the v3 has several provisions regarding to against software patents, ODF, and more. Also keep in mind that GPL-3 is recognized as a legal document in several jurisdictions.

GNU AGPL-3 for websites actually is a special version of the GPL that covers the application layer's licence.

LGPL is a "weaker copyleft" license designed for libraries.

Hardware is licensed as CERN Open Hardware Licence v1.2 which provides a strong copyleft and is in my opinion closer to the copyleft paradigm on GPL.

Hope that helped.