OPEN SOURCE MISSION CONTROL SOFTWARE FOR SMALL SPACE PROJECTS

M. Barschke[1], Ş. Özkan[1], M. Johnson[2]
[1]Cranfield University, Cranfield, Bedfordshire MK 43 0AL, United Kingdom
[2]JA, Oxford, United Kingdom

Abstract

Many small space projects such as CubeSats, PocketQubs, and sounding rocket experiments are run by student, university or hobbyist groups. Most of these projects require a mission control application to command and monitor the platform and its payload, but these are often developed from a clean sheet at the last minute and the source code of these systems is rarely made available to others to use and build on. We describe the Open Mission Control (OMC) platform that we have developed and open sourced for our own and others' projects. The software is developed as a modular system and provides a software framework, which can easily be customised and adapted for a wide range of projects.

The first mission that will use the Open Mission Control platform is the myPocketQub 442 payload for the United Kingdom Space Agency's first CubeSat mission: UKube-1. myPocketQub 442 is one of the four payloads selected to fly on this mission, and has been developed by the United Kingdom Students for the Exploration and Development of Space (UKSEDS) student organisation as a spare time project. It consists of a myPocketQub IQEA in payload adaptor configuration with five PocketPayload format subpayloads. The PocketPayload is a compact payload format designed to allow very low cost experimental access to space.

The primary objective of the mission is to demonstrate the utility of the myPocketQub IQEA and PocketPayload formats for small space missions and gain flight heritage for their reference implementations. A secondary objective is to provide a free, or very low cost, platform that can be used by schools to integrate real hands on space missions into the curriculum to inspire school pupils about science and engineering. The number of payloads and the diverse objectives of the myPocketQub 442 project allow Open Mission Control to demonstrate that it is a comprehensive and flexible solution.

The software is designed in such a manner that it can be used at different levels of complexity depending on the type of user operating it. These levels range from showing data in an appealing way for a school class or popular science television programs, to acting as a professional grade mission command, control and monitoring system to be used by student or professional mission controllers. As well as controlling flight hardware, the software will also be used to interact with engineering models connected to the Internet, which will allow monitoring, commanding, and analysing data from this hardware under realistic boundary conditions. The system also has a robust method of providing historical and real time data to thousands of simultaneous clients from the flight hardware and provides the ability to interactively review or playback any previous period of the mission in real time and fast forward modes.

The software, as is the myPocketQub project, is open sourced under the standard Open Source Initiative BSD license and has been developed and tested in parallel for Microsoft Windows, Mac OS X and Linux. We hope that by making the software, its supporting infrastructure and the myPocketQub 442 project available to anyone who is interested, our work will encourage many groups to use the software for their projects, to make improvements and mission data available to all and to provide a valuable and constantly improving building block for the community.

1. INTRODUCTION

In recent years the number of CubeSat sized space missions increased significantly [1]. These projects establish new perspectives to integrate real space missions into university education. "The opportunity to perform on-orbit operations for one's satellite as a student is a very rare but extremely rewarding experience [2]."

Such on-orbit operations require monitoring and commanding software to be used as an interface for the operator to communicate with the platform or the payload. Typically, this software is developed in form of several separate tools, which are tailored to the specific mission [3, 4].

There is some basic functionality for this kind of software, which is similar for most missions. With the Open Mission Toolbox, Open Mission Control will

provide a framework that already contains the required basic functionality and can be customised for a specific project with minimal effort.

Events like the *NASA Open Source Summit* and the *ESA Summer of Code in Space* promote the use of open source software for space applications [5, 6]. This is especially important for projects with limited resources, since they can save money and time by using open source software within their projects. Open Mission Control is open-sourced under the BSD 2 clause license that permits commercial and non-commercial use with minimal restrictions [7].

2. myPocketQub 442

The myPocketQub 442 mission will be the first mission to be controlled by Open Mission Control. It was used as a reference to define the requirements for Open Mission Control and it will be the first demonstration of the capabilities of the software. The myPocketQub 442 hardware consists of a myPocketQub consisting of the myPocketQub IQEA bus and five PocketPayloads. A myPocketQub *In-plane Qub with Experiment Array* (IQEA) is a free flying myPocketQub (one eight the volume of a 1U CubeSat) flat packed on a CubeSat PC/104 card with multiple positions for mounting PocketPayloads [8].

Due to their small dimensions and simple interfaces, PocketPayloads provide very low cost experimental access to space and can therefore be used for projects that would not be possible on a larger scale, due to resource constraints. The five PocketPayloads on myPocketQub 442 range from science and engineering experiments to an educational platform for software payloads of schools and universities:

- OpenSpace365 is an educational payload, which consists of an Arduino microcontroller, a CCD camera and a set of several sensors. It will allow school pupils, university students and hobbyists to run their own software experiments in space.
- OrbitView is an optical payload, which creates interactive panoramas of UKube-1 and the view from UKube-1.
- Qubduino is an engineering payload that demonstrates a low cost Field Programmable gate Array FPGA running self-repairing virtual payloads.
- SuperLab is a science payload that characterises superconducting materials.
- SuperSprite is based on the Cornell University Space Systems Design Studio Sprite and demonstrates a spacecraft on a chip as a proof of concept.

This wide variety of applications provides an ideal opportunity to demonstrate the capabilities of Open Mission Control. Furthermore, the distribution of the software to thousands of participating schools and universities will provide an ideal evaluation of the general software layout as well as an extensive testing of the functionality of the software.

2.1. The myPocketQub 442 use cases

A number of different use cases were considered while deriving the requirements for Open Mission Control for myPocketQub 442. These use cases represent different capabilities that the software is supposed to provide in order to ensure that it can be adopted to a variety of different small space missions. Four main use cases are described here to give insight into the different fields of application of Open Mission Control for myPocketQub 442:

- The first use case is the use as a command and monitoring system for spacecraft on orbit. This will demonstrate the ability of Open Mission Control to be used as a professional ground station software, addressing all the needs of such a task. In the case of UKube-1 this includes the use of Open Mission Control as monitoring and commanding software for myPocketQub 442 and the UKube-1 spacecraft.
- The second use case involves schools and universities, using the software to follow a real space mission and potentially to interact with the spacecraft. In the case of myPocketQub 442, schools and universities will use the software to upload their virtual payloads for OpenSpace365, monitor their experiments as they run and download the produced data for analysis.
- The third use case involves the use of Open Mission Control as a command and monitoring system for scientific and engineering experiments running on a spacecraft. Apart from visualising the data produced by these experiments, it must also support downloading these data for further usage.
- Finally, Open Mission Control can be used to communicate with engineering models of the real spacecraft connected to the Internet. These engineering models allow users to command and monitor the spacecraft and its payloads in real time and to simulate different critical mission phases under real conditions. The use of virtual spacecraft models will increase the utility of Open Mission Control in education and general development. The hardware of myPocketQub 442 is selected in such a manner that anyone can, after some space certified components are exchanged for comparable low priced components, buy these components and copy the spacecraft for a very low price. This allows

universities, schools, and hobbyists to combine their own hardware with the Open Mission Control software for the use as engineering models and training material. Internet based software reference models can be used to verify the hardware models or as training objects in their own right. For myPocketQub 442, an automated value generator will be used as a virtual model of the spacecraft. This allows to produce real time test data to test the software under circumstances as realistic as possible.

3. OPEN MISSION CONTROL

Open Mission Control is developed using National Instruments LabVIEW™ Professional Development System (PDS). A graphical programming language was regarded as important for this task to allow simple customisation by users without a software background. In addition, LabVIEW provides a number of functional blocks, such as ready made GUI graphs, that are suitable for immediate use in applications such as Open Mission Control. The LabVIEW development environment runs on Windows, Mac OS X, and Linux and the PDS permits the royalty free distribution of the completed application, runtimes and source code allowing anyone to use the application free of charge [9].

There are some main functionalities which are required to monitor and control any small space mission. These main functionalities are identified by Cooper et al. [2] as:

- Acquire incoming telemetry packages
- Process, display and store the data
- Allow to compose and send commands

Figure 3.1 shows a schematic of an extended set of main requirements, which are the main tasks for Open Mission Control.

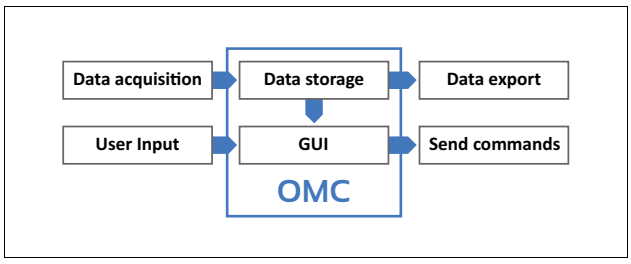

FIGURE 3.1: The main functionalities of Open Mission Control.

The *data acquisition* functionality includes the download of the data from the Internet or directly from hardware components. The data will then be converted from the incoming protocol to the Open Mission Control database format. The Open Mission Control Toolbox allows a number of standard blocks to be selected to support standard protocols such as X.25.

The *data storage* functionality includes a database and a data manager that provides the data for display in the Open Mission Control graphical user interface and also for the data export. The data provider is capable of delivering multi sample rate data for display purposes. Within Open Mission Control the data is stored in a database based on CSV-files to keep the data management simple and multi-platform conform.

The *data export* functionality is provided by a module from the Open Mission Control Toolbox, which permits the export data in convenient formats and time ranges.

The *command* functionality sends commands to the spacecraft. Command selection and transmission is based on the arm/release double action procedures used in professional spacecraft operations [10]. The user has to unlock the command window and then select a command from the *routine commands* or *crucial commands*, or enter a command manually and then add a timestamp to the command. When the command is selected, the user can load the command into the command window. When the required commands are gathered in the command window they can be sent to the spacecraft by pushing the *send command* button which will lead to a pop-up window, asking whether the command should be send as another level of security. Once uploaded to the spacecraft, the commands will be executed at the time defined by the timestamp. As long as the command is not yet uploaded to the spacecraft, the user can still stop the execution and delete selected commands from the command window.

3.1. Open Mission Control for myPocketQub 442

In addition to the basic requirements, a few custom functions will be implemented to support myPocketQub 442. For example, a software payload upload component that allows schools and universities to upload their OpenSpace365 payload to the OpemSpace365 team for review before onward transmission to the spacecraft.

Once the user starts Open Mission Control for myPocketQub 442, the mission selection screen is shown, where the user can select the mission and the language of the user interface. Currently English and German are available, but the support for more languages is planned. When the myPocketQub 442 mission patch is selected, the user is directed to the mission's home screen.

The *home screen* of Open Mission Control for myPocketQub 442 shows the ESA control room at ESOC, Darmstadt, Germany. The home screen serves as an overview over the entire mission,

where the most important data are shown embedded in the picture of the mission control room (Figure 3.2). Mission data is superimposed on the screens to give the user the feeling that they are at ESOC. The mission data displayed includes a model of the spacecraft, the ground track on a world map, and a number of status boards, which indicate the status of the sub-systems.

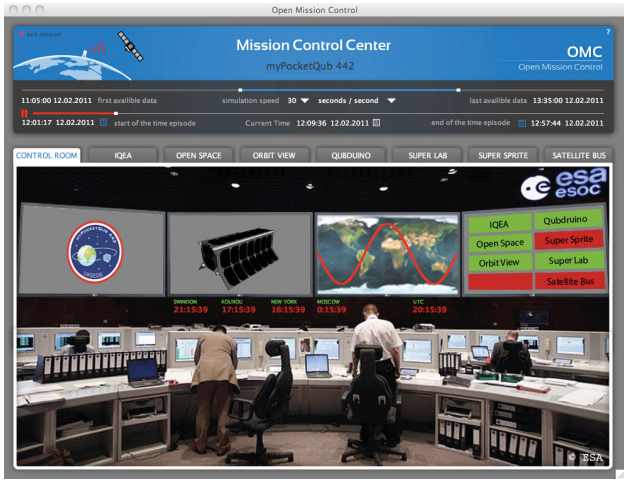

FIGURE 3.2: The home screen of Open Mission Control for myPocketQub 442

The header of Open Mission Control provides extended time control. As most satellite missions will retrieve their data not in real time, but during down-link windows, there cannot be a real time display of the data in most cases. Therefore the header allows the user to choose any period within the available mission data and replay the data in real time or faster or slower than real time. This allows for simulated 'real time' simulations even if the data is not retrieved in real time.

Due to the development of remotely accessible ground stations and ground station networks such as myGroundStations.com, small space projects have access to many more ground stations than previously and therefore have longer downlink windows [11, 12, 13]. In the future, continuous access to such spacecraft may be possible [14]. Open Mission Control provides a real time mode for such missions. However, even for missions with continuous real time data, playback of previously recorded data is useful for reviewing critical phases of the mission or for an overview of the mission.

The OrbitView tab is shown in Figure 3.3 as an example for one of the payload tabs implemented for myPocketQub 442. A header containing general mission and time information and controls, is fixed at the top of the interface. Below the header, a panel containing raw numeric data is displayed with graphical interpretations arranged below.

FIGURE 3.3: The OrbitView tab of Open Mission Control for myPocketQub 442

3.2. The Open Mission Toolbox

Open Mission Control is designed to provide a framework that can be adapted quickly and easily to support a variety of small space projects such as CubeSat, myPocketQub, and sounding rocket experiments.

The Open Mission Control Toolbox is a subset of the source code consisting of the graphical user interface, the basic structure of the program and a number of ready to use functions typically required to implement a mission control application.

The Open Mission Control graphical user interface can be adapted for a project quickly and easily by populating it with elements from the Open Mission Control Toolbox and standard LabVIEW elements. This approach allows users with limited programming experience to create a sophisticated mission control system starting from a solid base. Users with experience in graphical programming can quickly extend the functionality and share new elements of the toolbox with other users to allow work developed for a broad range of missions to be available to all.

To maintain the software and to develop new features for the Open Mission Control Toolbox, support for additional missions may be implemented directly by the Open Mission Control project including:

- UKube-1, the first British CubeSat which will carry payloads provided by AMSAT-UK, Astrium Ltd., the Open University, UKSEDS, and the University of Bath.
- FUNcube-1, the AMSAT-UK amateur radio payload on UKube-1.
- CubeSat-on-Demand 443
- FUNcube-2

- myPocketQub 391
- Spacecraft-on-Demand 444

Any mission that is interested in using the software is welcome to work with the Open Mission Control team or download the source code from the SVN servers and work independently.

4. CONCLUSION AND FUTURE WORK

With Open Mission Control for myPocketQub 442, we developed sophisticated monitoring and control software for a PocketQub space mission with five PocketPayloads requiring a range of different functionalities. The software is compatible with Windows, Mac OS X and Linux platforms. Open Mission Control is designed in such a way that it can be used as a professional mission control application, as software for scientists to retrieve data from their experiments and for educational purposes.

The first step in the future development of Open Mission Control is to evaluate the lessons learned from using Open Mission Control as monitoring and commanding software for the myPocketQub 442 mission and use the results to improve the Open Mission Control Toolbox to make it ready for distribution.

Furthermore, the Open Mission Control Toolbox can then be expanded in functionality to support as many project requirements as possible. For example, a basic version of the mission planning functionality described in Lee and Kim [15] could be implemented in the future. This functionality would predict the spacecraft's position in the future and would therefore allow predicting the ground-station contact and planning the execution of certain commands at a certain point in orbit.

Furthermore Open Mission Control could be extended in its capability to be used as an educational software tool, for example by programming different missions in the form of Web-accessible learning objects, for use in school and university education [16].

ACKNOWLEDGEMENTS

Open Mission Control receives financial support from ESA (Summer of Code in Space program), JA and the STFC and donations in kind from CodeBase and National Instruments. We also thank Clyde Space, UKSEDS and Vega for their advice and support.

REFERENCES

[1] R. Nugent, R. Munakata, A.Chin, R. Coelho, J. Puig-Suari (2008). The CubeSat: The Picosatellite Standard for Research and Education. *In: AIAA SPACE 2008 Conference*, September 9-11, San Diego, California, USA.

[2] C. Cooper, R. Fevig, J. Patel (2002). *The CubeSat Ground Station at the University of Arizona* [online]. [Accessed 2011-08-02]. Available from: <ftp://pirlftp.lpl.arizona.edu/pub/cubesat/cubesat_papers/gspaper.pdf>.

[3] Nihon University (2008). *About Groundstation software* [online]. [Accessed 2011-07-28]. Available from: <http://cubesat.aero.cst.nihon-u.ac.jp/english/software_e.html#01>.

[4] M. Ferrante, L. Petrozzi, A. Di Ciolo, G. Ortenzi, G. Torso (2004). Ground station software for receiving and handling IRECIN telemetry data. *In: The 4S Symposium Small Satellites, Systems and Services*, September 20-24, La Rochelle, France. p60.1.

[5] NASA (2011). *NASA Open Source Summit 2011* [online]. [Accessed 2011-07-20]. Available from: <http://www.nasa.gov/open/source/live.html>.

[6] ESA (2011). *ESA Summer of Code in Space 2011* [online]. [Accessed 2011-07-20]. Available from: <http://sophia.estec.esa.int/socis2011>.

[7] *Open Source Initiative OSI - The BSD License* [online]. (2011) [Accessed 2011-08-19]. Available from: <www.opensource.org/licenses/bsd-license.php>.

[8] M. Johnson (2011). PocketSpacecraft - The start of the personal space age? [Presentation]. *Presented at: UK Space Conference*, June 4-5, University of Warwick, Coventry, UK.

[9] National Instruments (2011). *Product Information: What is NI LabVIEW?* [online]. [Accessed 2011-07-28]. Available from: <http://www.ni.com/labview/whatis/>.

[10] R. Lowe (2011). *SPOPST – Simulator for space operations training* [User Manual]. VEGA Space Ltd.

[11] J. W. Cutler, C. A. Kitts (1999). Mercury: a satellite ground station control system. *In: Aerospace Conference*, March 6-13, Aspen, CO, USA . pp.51-58 vol. 2.

[12] M. Wilkinson, C. Swenson (1999). Design of a satellite tracking station for remote operation and multi-user observation. *In: Small Satellite Conference*, August 23-26, Logan, Utah, USA.

[13] JA (2011). *myGroundstations.com* [online]. [Accessed 2011-08-02]. Available from: <http://mygroundstations.com>.

[14] J. Cutler (2003). Ground station virtualization. *In: International Symposium on Reducing the Cost of Spacecraft Ground Systems and Operations*, July 8-11, Pasadena, CA, USA.

[15] B.-S. Lee, J.-H. Kim (2003). Design and implementation of the mission planning functions fort he KOMPSAT-2 control element. *Journal of Applied Systems Studies*. vol. 20, pp.227-238.

[16] Y. Baradaranshokouhi, J.A. Rossiter (2011). *Developing Remote and Virtual Laboratories with LabVIEW* [online]. [Accessed 2011-07-25]. Availible from: <http://sine.ni.com/cs/app/doc/p/id/cs-13030>.