IAA-AAS-CU-17-06-07

Development of a Flexible Nanosatellite Mission Control System Using Agile Development Methodology

Richard Duke, Brian Stewart, Ben Taylor, Christopher P. Bridges, Simon Fellowes and Guglielmo Aglietti

r.duke@surrey.ac.uk

4th IAA Conference on University Satellite Missions and Cubesat Workshop
4th December – 7th December 2017
Rome, Italy

Introduction

Previous University of Surrey Satellites based around a custom design for each individual satellite.

In 2015 a new system was required for multiple missions. It was an ideal opportunity to consolidate support for;

- Spacecraft operations
- S/C development and testing
- Communications research
- University and amateur cubesat community
- Teaching

Requirements

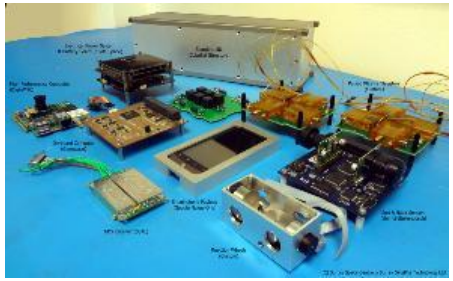
Summary of main requirements:

- Multiple simultaneous connections needed
- Easy to adapt to different spacecraft
- Multiple users; operators; AIT engineers, System Engineers; Students
- Mission critical operations vs accessibility for students
- Varying levels of automation (teaching vs lower operations cost)
- Expandable for the future

Data Connections

Dual VHF / UHF

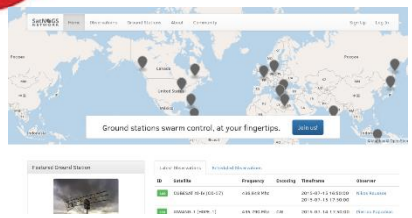
Software Testbeds



3 Metre Dish

Mission Operations Centre

Amsat Community



Partner Groundstations

AIT

Development Strategy

Agile Development strategy used

- It aims to produce good results by promoting close teamwork, over fixed processes.
- It values working software (especially early prototypes) over heavy documentation.
- It integrates the customer or end user during the development phase and welcomes input and change requests even late in the development to make sure that the output works well.
- It makes the assumption that that requirements and plans will change and therefore this can be expected and planned for.

Especially suitable when full requirements may not be fully described at the start of the project

Empowers developers and concentrates focus on good engineering rather than process

Architecture

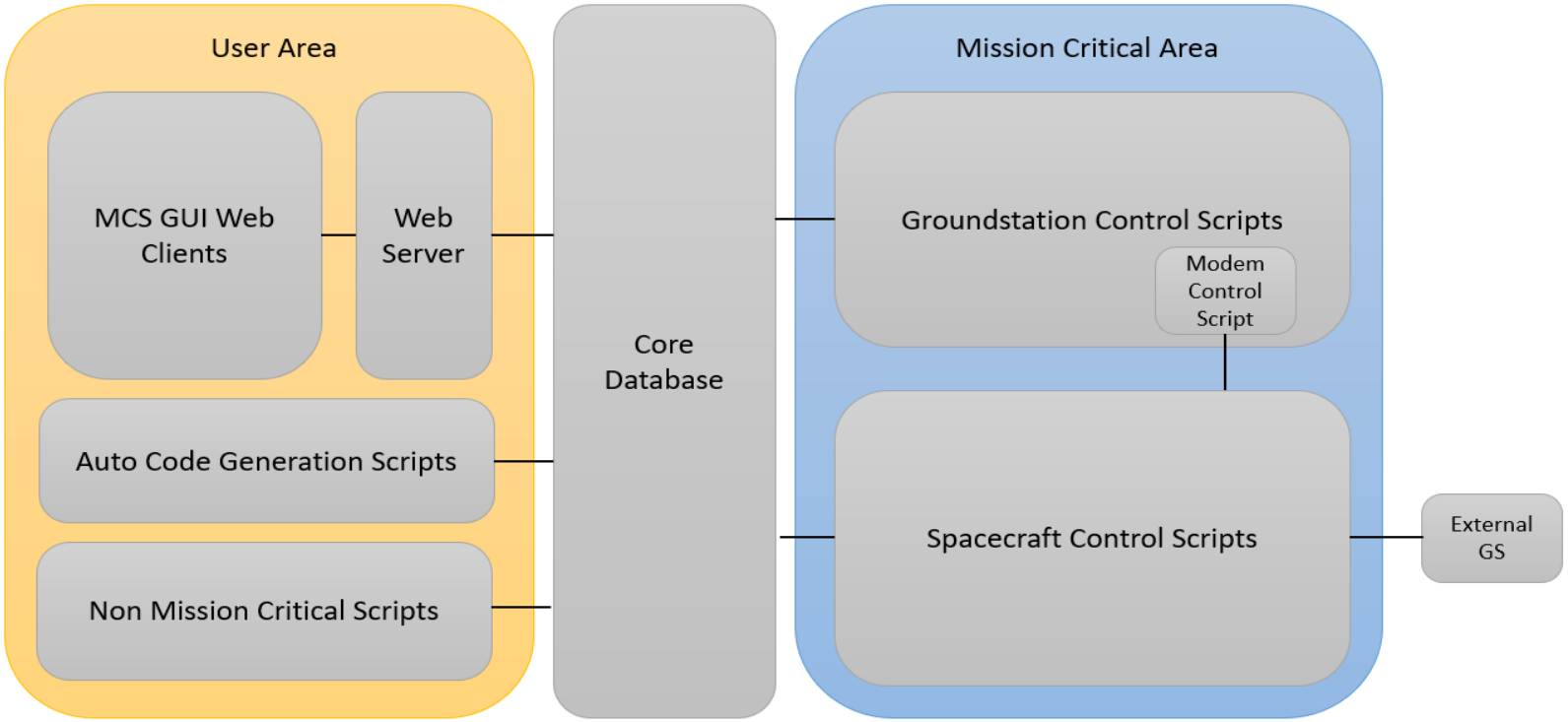
With agile development the architecture of the system is critical to make sure it can cope with changing requirements.

We placed a relational database (PostgreSQL) at the core of the system with standard SQL language used to access data.

- Based on industry standard technology.
- Databases are designed to be accessed simultaneously and connected to multiple different types of systems.
- Everything is logged by default. Traceability of commands from first development all the way to end of life.

Architecture

‘Mission Critical’ and ‘User’ programs are separated.

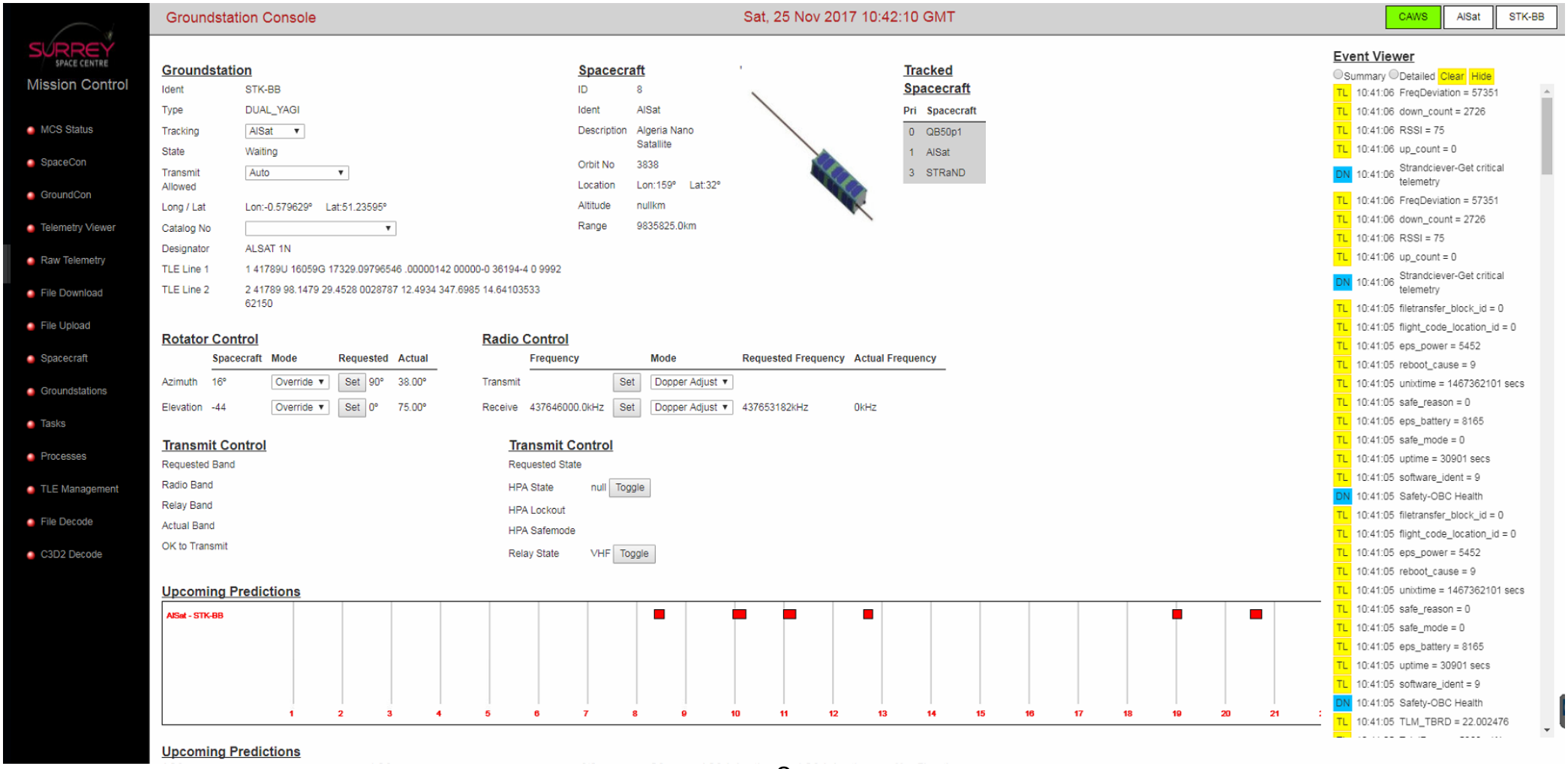


The mission critical system programs can be tightly controlled, while allowing rapid development for the rest of the system.

Easy to set up safe access via database to allow students to develop new systems without risk to key operations.

User Interface

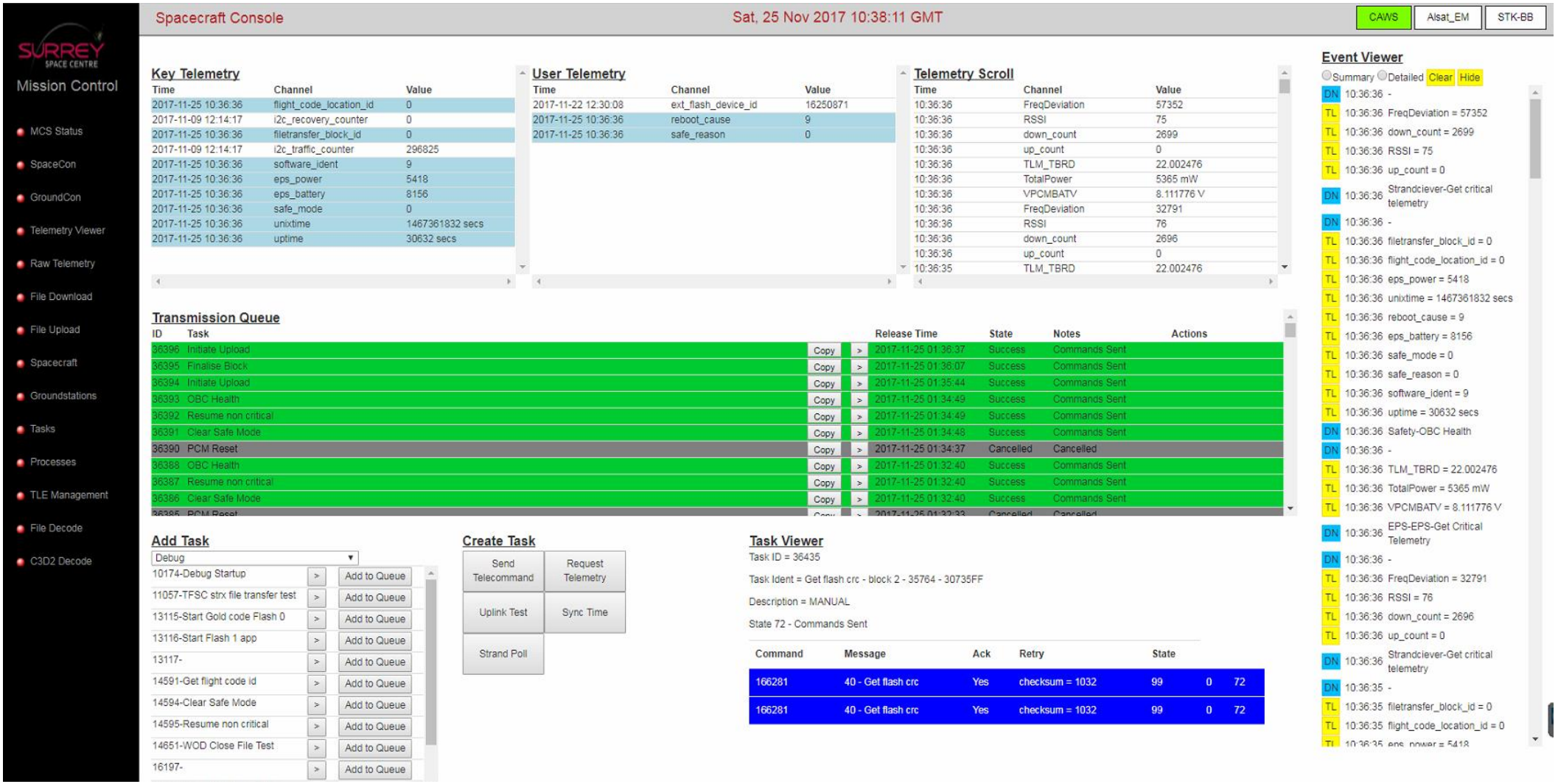
All groundstations and spacecraft are accessed by the user though the same interface



User Interface

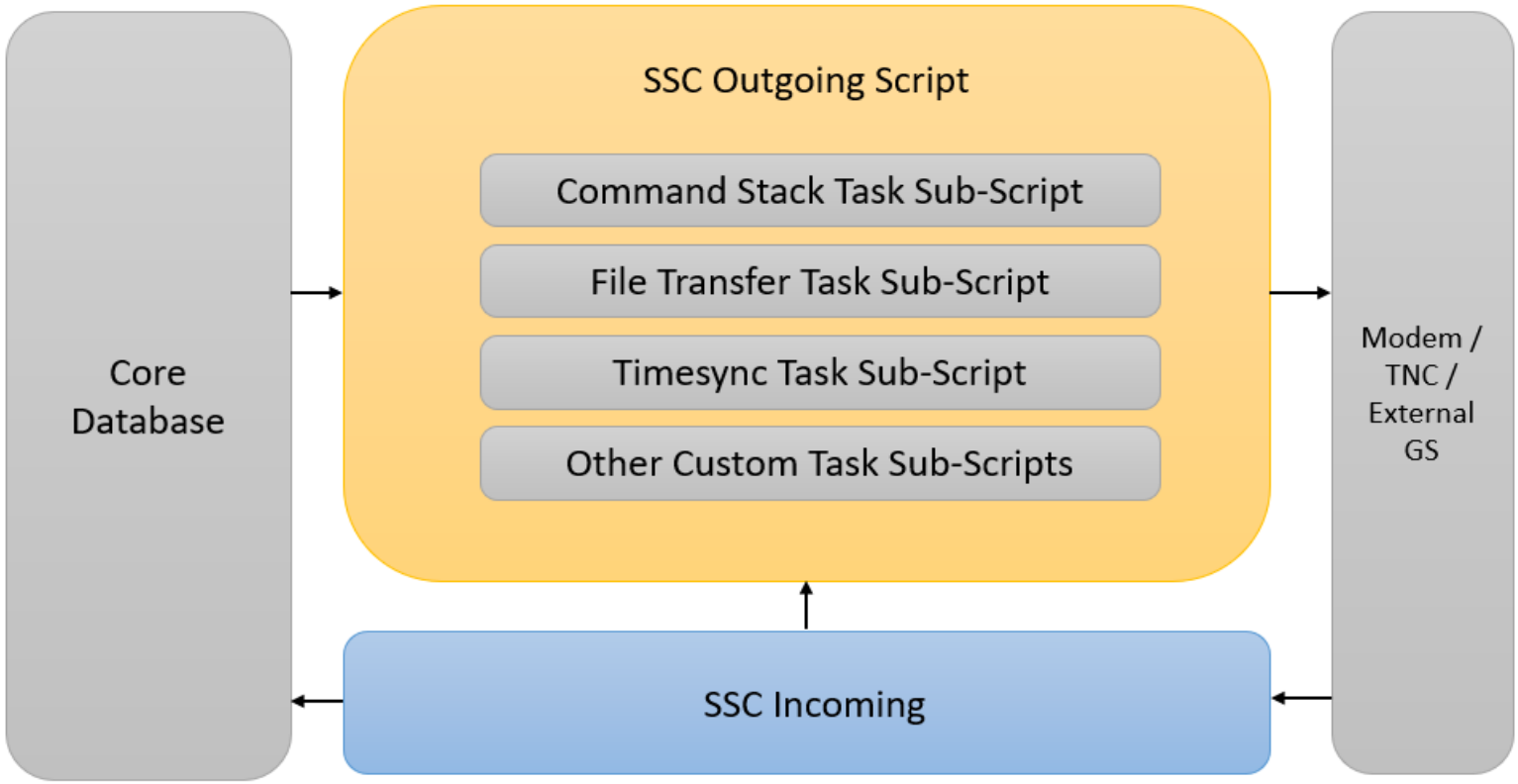
The user interface is based on a standard web interface.

Commanding is primarily done via command stacks.



Task

Each task is a self contained plugin script allowing custom scripting if required.



Three standard scripts are used to cover almost all requirements

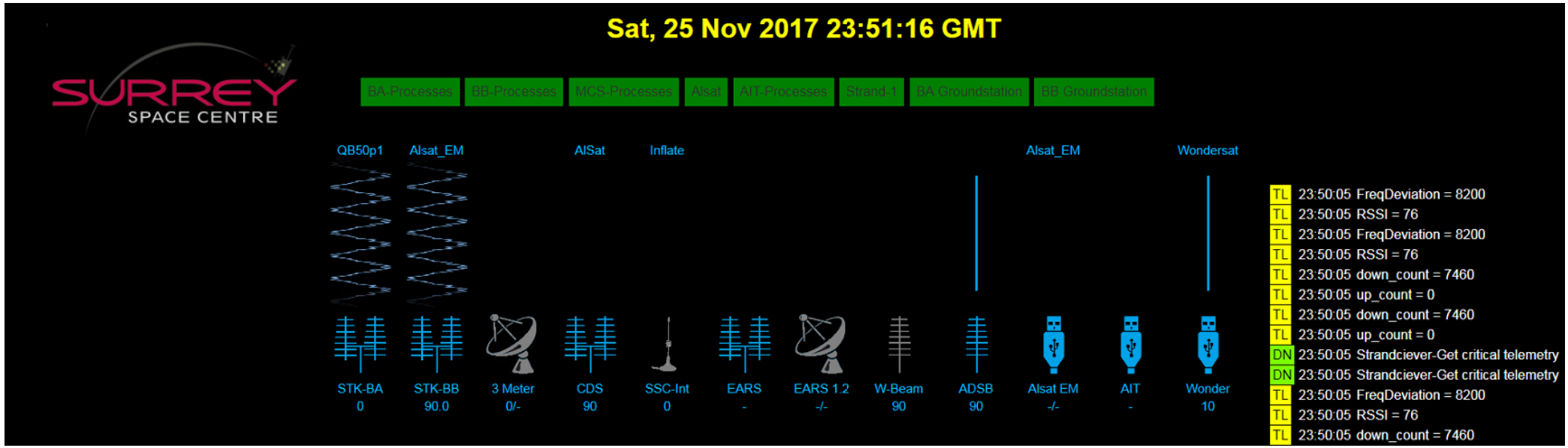
User Interface

Incoming data sets can be plotted graphically as soon as the data is received. Very useful during the testing phase!

User Interface

Custom displays can be built independently without affecting other systems.

This is especially important for teaching and outreach, allowing screens that can demonstrate real-time status of both ground and space segments



Mission Heritage

Full Missions

- Alsat
- Inflatesail

Build and Test

- Remove Debris (VBN Cubesat)
- CubeSail
- SME Sat

Heritage Missions

- Strand
- DeorbitSail

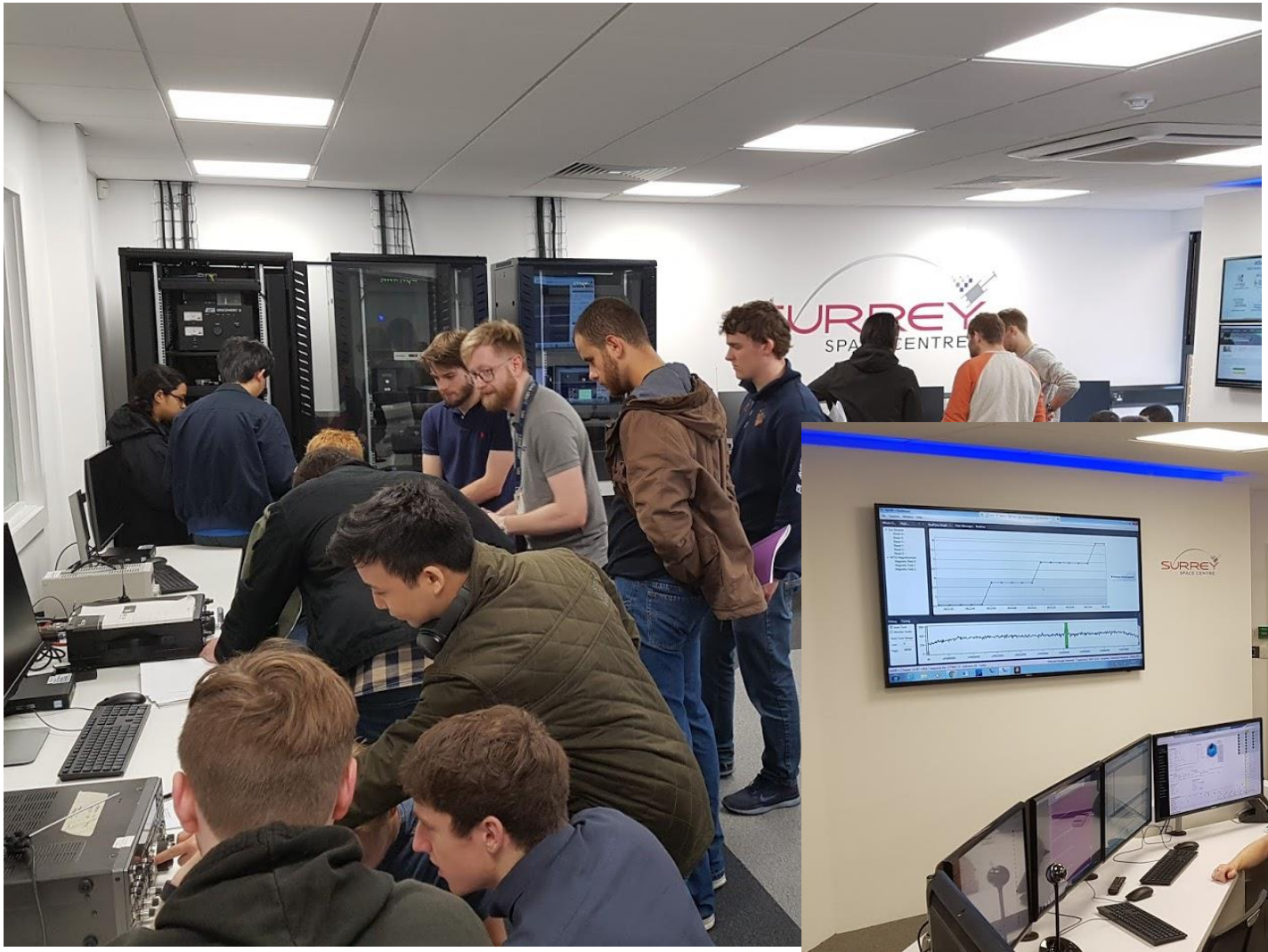
+ various amateur and university cubesat support

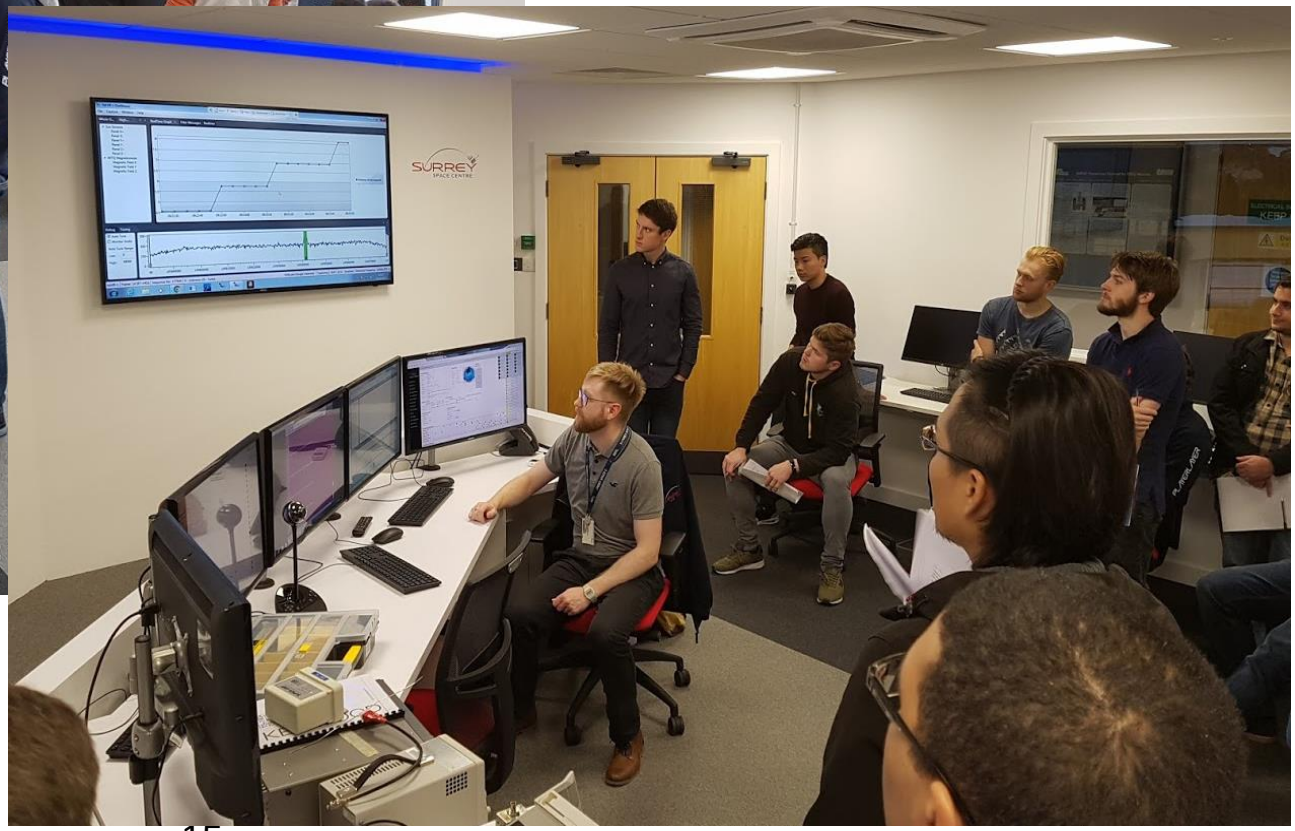
Alsat-1N

Used in Simulation Mode with engineering module to train operators

Installed at ASAL (CDS) in Algeria for primary groundstation

Mission Operations Training





Development Strategy

Recommendations from Development

- Agile development allowed us to have early initial capability, increasing the testing time with the spacecraft. Software issues were found early and fixed instead of late in the development.

- Initial architecture is critical. Make sure you have a flexible core that can cope with changing requirements. Use of a standard database / interface has helped

- Combining test software and on orbit control software provides a huge timesaving in training and provides confidence that the mission control system will work once in orbit.

- Agile helped with project schedule challenges; make sure the whole system is there, then improve.

Thank You! Questions?