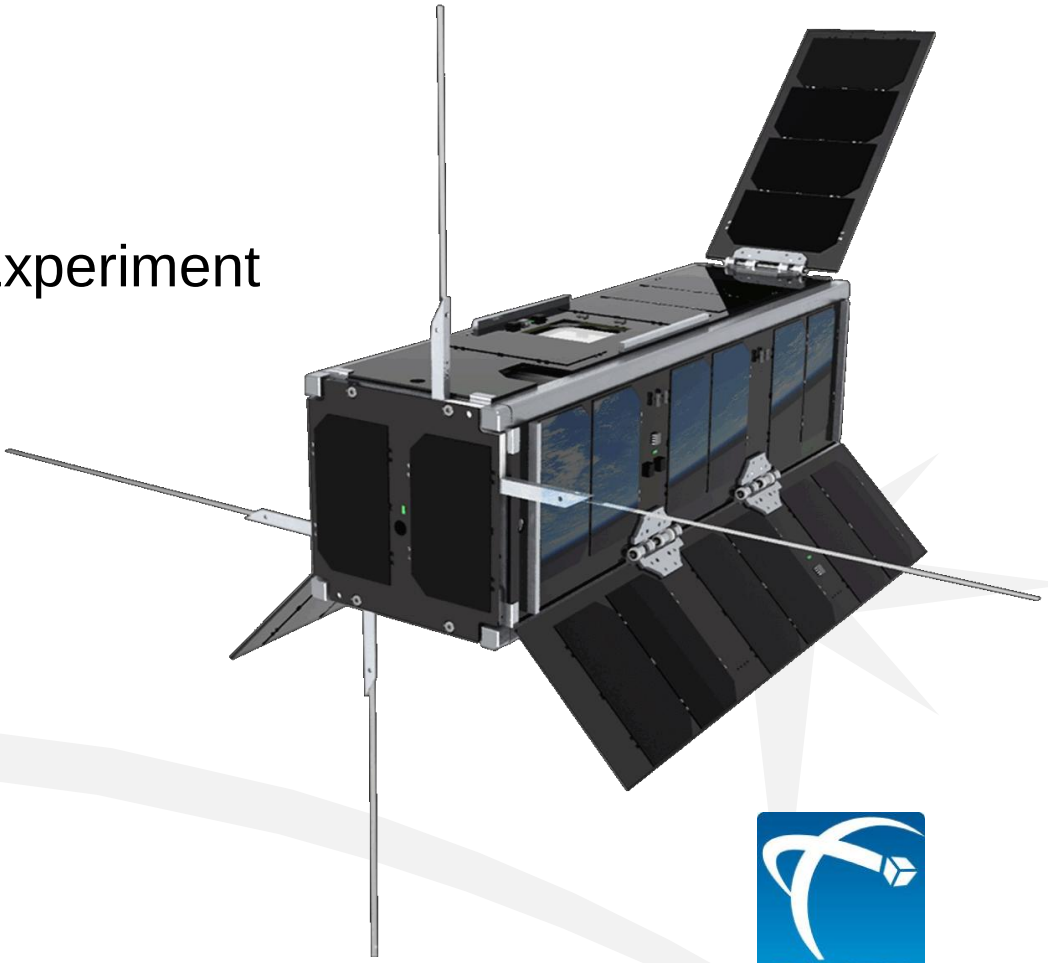
Developing onboard software 400 miles from the cleanroom

Component-based reusable software for UKube-1

Peter Mendham and Mark McCrum

UKube-1

- United Kingdom Universal Bus Experiment
- 3U CubeSat
- Five payloads
  - C3D – imager
  - JANUS – radiation
  - MIC – FPGA-based processor
  - TOPCAT – GPS occultation
  - FUNTRX – AMSAT transceiver
- Experimental platform (e.g. S-Band Transmitter)



CLYDE SPACE

bright ascension

Software for UKube-1

- The software challenges
- The development context
  - Organisational
  - Physical and geographical
- Working in a changing environment
- Software to solve the problems
- The future...

bright ascension

We'll just do it in software

- Many challenges common to CubeSats
- Low link budget
  - One (primary) ground station
  - Low up/downlink bandwidth
  - Short passes
- Lots of data
  - Large number of experimental systems
  - Multi-MB of payload data
- Limited operations
  - As “hands-off” as possible
  - Need flexible control over low-level aspects of system
  - Need lots of automation: time-based, orbit based, event-based, onboard scripting

bright ascension

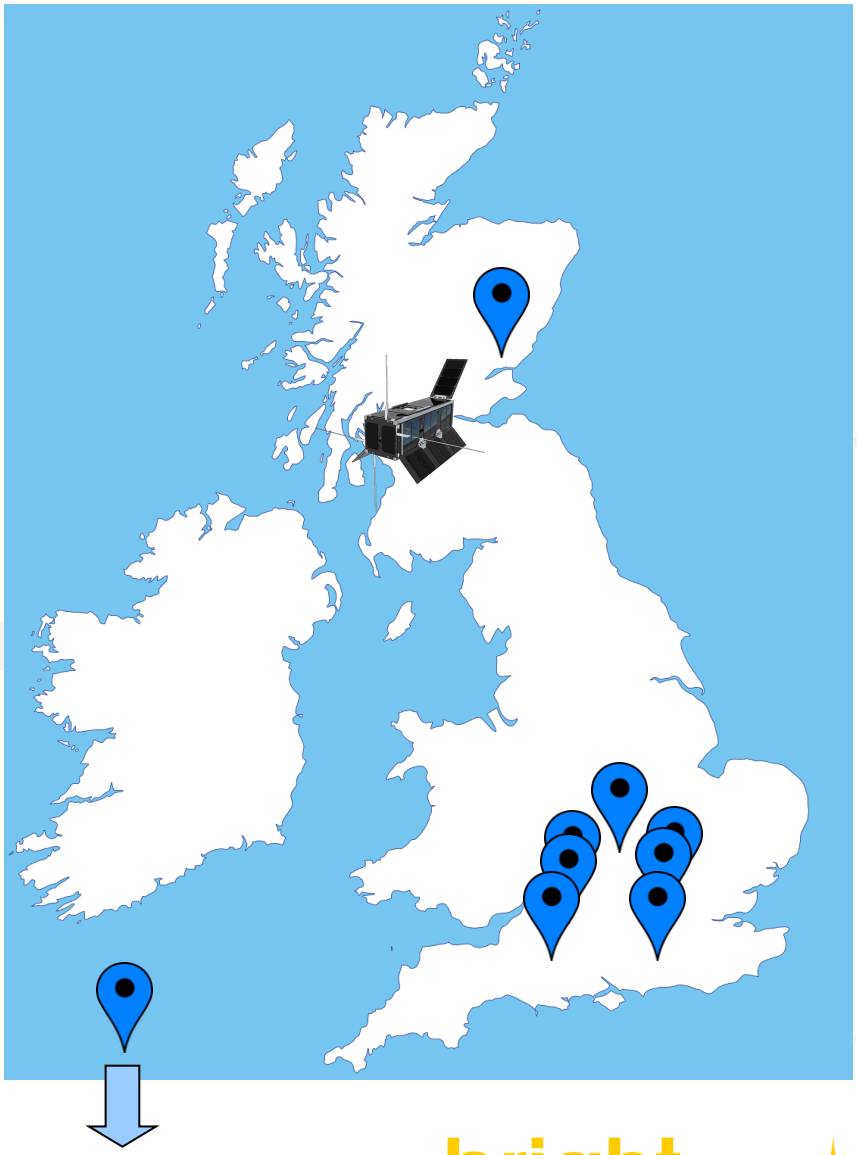
Organisational Challenges

- Some unusual challenges for a CubeSat project
- Large number of teams
  - Experimental platform components
  - Payloads
  - Ground segment
  - ~10 teams, not counting suppliers
- Loosely coupled
- Software subcontracted by platform supplier
  - Late in development process

bright ascension

Physical Challenges

- Teams are not next door to each other
- Software team distributed
- Only one set of hardware
  - No EMs
  - Highly contested
  - Often under development



bright ascension

Waiting for teleportation...

- Use of hardware via remote access
- Software built on abstraction framework
- All I/O abstracted
  - RF link interchangeable with umbilical serial link
- OS abstracted
  - Software can be rebuilt for Linux on a PC
  - Development and test without hardware

bright ascension

Continuous Refinement

- At software kick off
  - Little hardware definition, nothing final
  - Little operational definition
- Software necessary for hardware testing
- Software necessary for environment testing
- Change was a certainty!
- Adopted an iterative development approach
  - Two-week iterations, frequent deliveries
  - Adapt priorities to project needs
  - Many lessons learned...
- Use of unit testing framework
  - Automated testing possible, no hardware access

bright ascension

Designing for Change

- Design the software architecture to accommodate change
- Base software around components
- Each component is functionally self-contained and configurable
- A component is what is “seen” from ground
- Most change can now be accommodated in
  - Which components are deployed
  - How many are deployed
  - How they are connected together
- Built on abstraction library
- Based on CCSDS standards

bright ascension

A Component-Based Approach

- Example components
  - Subsystems: EPS, Battery, Switchboard, Transceiver
  - Monitoring: Sampler, DataPool, Aggregator, DataLogger, Monitor
  - Automation: EventAction, TimeAction, Script
  - System: ModeManager, DeploymentManager
- Standard interface to components
  - Parameters (get/set), actions (invoke)
  - These are the only TM/TC operations necessary
- Components highly configurable

A Bright Future

- The UKube-1 OBSW has been spun-out into a product
- Easily ported to new platforms and OSs
- Abstraction framework
- Component framework
- Library of components
- Tooling
  - Auto-code generation for components, deployments
  - Generation of documentation
- Customisable from the ground up
- Rapid development, lower risk and **more adaptable to change**
- Strong interest from a number of other missions and R&D projects

bright ascension

Lessons Learned

- The only certain thing is that nothing is certain
- Organisation issues can be as big as technical issues
- Use of remote access for software development is feasible
- Subcontracting or using a separate software team is feasible
  - A way of managing change is needed
- A software architecture and software tools that let you adapt to change quickly, easily and cheaply can be very important

Contact Us

Question, comments or suggestions

Bright Ascension Ltd

www.brightascension.com

enquiries@brightascension.com

+44 (0) 1382 602041

bright ascension