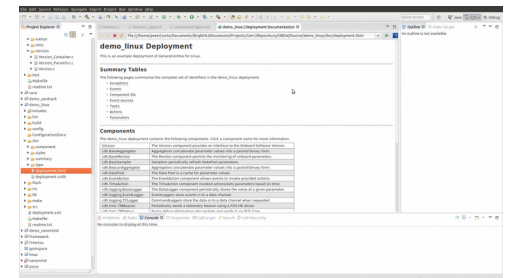
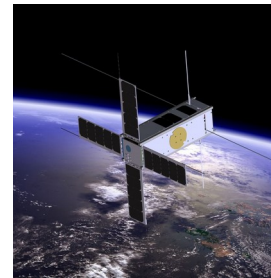
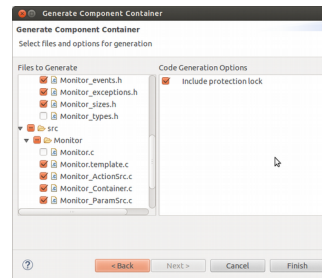
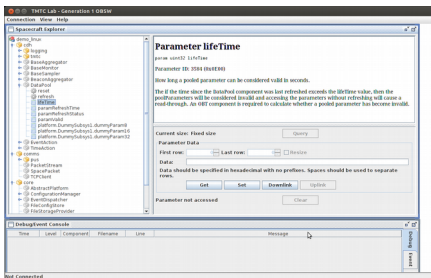




Integrated Flight-Ground Software for Rapid Mission Development

Mark McCrum, Alex Mason, Peter Mendham



generationone

About us

- **Space technology company** specialising in software
 - Based in Scotland, UK
- Founded in 2011
- Experience with a wide range of **upstream software**
 - Flight
 - Mission control/operations
 - Simulation
- Mixture of software **products and services**
 - Supplying products to nano-satellite missions worldwide
 - Partnered with spacecraft integrators offering full software service
 - Ongoing R&D projects and consultancy with ESA, UKSA and others

Space
Segment



TM Packets

TC Packets



Ground
Segment

Typical Space System

- Interface specified at the level of telecommand and telemetry packets
 - Impoverished view of the functionality provided
 - High level patterns of interaction not readily captured
 - Labour-intensive
 - Error-prone
 - Hard to change (particularly across organisational + contractual boundaries)
 - Complicates operation + automation
- Problems
 - Doesn't scale well
 - Doesn't support rapid, agile development

Some alternative approaches ...



CSP



The Consultative Committee for Space Data Systems



PUS



GenerationOne

- **GenerationOne** is a core software technology and reference architecture
 - Applied to onboard software through a **Flight Software Development Kit**
 - Applied to operations through **Mission Control Software**
- GenerationOne addresses the key challenges by combining
 - **Model-based** software engineering
 - Model captures software at an architectural level
 - **Component-based** software engineering
 - Software is built from regularly structured modules (components)
 - A **service-oriented** architecture
 - Interactions between components conducted using services
- GenerationOne was not solely designed up-front
 - Started simple with features introduced gradually
 - Has been iterated and improved upon
 - Uses experience from many projects and missions
 - Features tested in a practical environment
 - Continues to be extended and improved

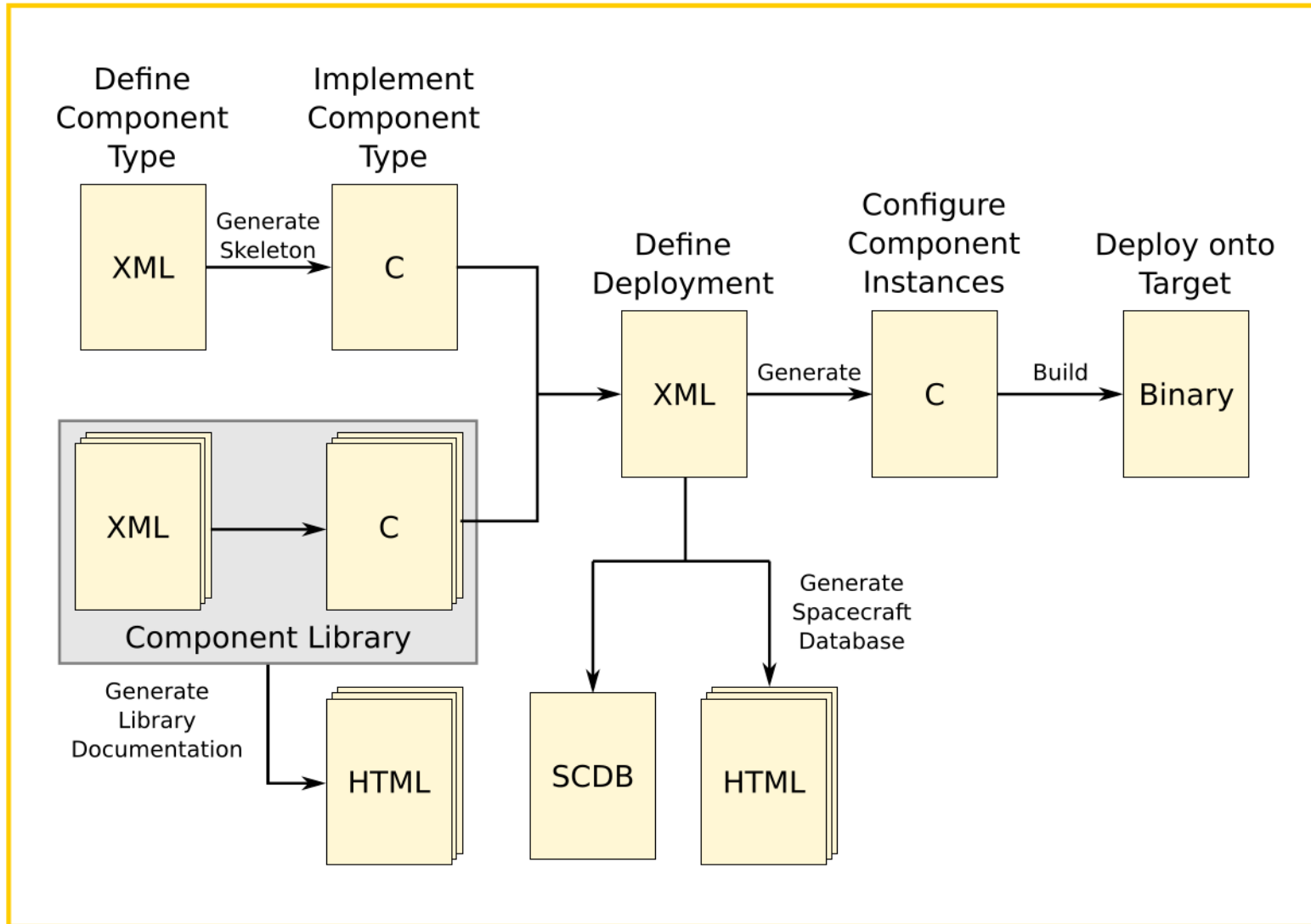
Characteristics of GenerationOne

- **Model-based** software engineering
 - Permits machine comprehension of software architecture
 - Enables tooling to assist with software development
 - Tools can also be used to assist with product/quality assurance
 - Model describes architecture across flight and ground
 - Model used across life-cycle from early development to end of operations
- **Component-based** software engineering
 - Key enabler for re-use
 - Each component includes implementation, tests and documentation
 - Complete system built from components
 - Lightweight underlying framework connects and supports components
 - Many (most) components portable across platforms and operating systems
- **Service-oriented** architecture
 - Provides consistent semantics for component interactions
 - Enables low level parts of software to be expressed as components
 - Raises the semantic level of operations
 - Separates interaction semantics from implementation protocol

Example software components

- **Subsystem components**, represent hardware
 - EPS, battery, ADCS, payload
 - Support for many off-the-shelf hardware subsystems and OBCs
 - AAC Microtec, Clyde Space, GOMspace, ISIS, Pumpkin, Vorago and many more
- **Data handling and monitoring components**
 - Sampling, data pool, aggregation, logging, monitoring, statistics
 - Support for most common onboard monitoring functions
- **Communications components**
 - Packet handling, telemetry reporting
 - Support for a number of different communications protocols
 - Includes support for ECSS PUS, CFDP
- **Automation components**
 - Absolute and relative time scheduling, orbit-based scheduling
 - Event-based automation
 - Onboard scripting
- **Mission-specific custom components**
 - Mode management, deployment sequencing, orbit counting

Typical workflow



MCS View

COAST MCS - LEOP - Orbit view

File Data Tools Help

LEOP Housekeeping Beacon

LEOP

- Layouts
 - CFDP Testing
 - Default Layout
 - Diagnostics
 - Orbit view
 - Pass DLink
 - Housekeeping
- cdh
- comms
- core
 - ConfigurationManager
 - EventDispatcher
 - FileConfigStore
 - FileStorageProvider
 - FileSystem
 - FileSystemManager
- OBT
 - reset
 - update
 - time
 - uptime
 - Storage
 - Time
 - platform
 - Version

Link Monitor

Time	Direction	Type	APID	Seq	Len	Service	Subservice	Description
Mon 12:25:32	Received	TM	2047	0	860	0	25	Housekeeping
Mon 12:25:37	Received	TM	2047	0	860	0	0	IDLE
Mon 12:25:42	Received	TM	2	139	12	3	25	Housekeeping
Mon 12:25:42	Received	TM	2047	0	860	0	0	IDLE
Mon 12:25:47	Received	TM	2	140	12	3	25	Housekeeping
Mon 12:25:47	Received	TM	2047	0	860	0	0	IDLE

Packet Type : Telemetry
 APID : 2
 Sequence Count : 141
 Length : 27

Reply to Query with transaction ID = 0
 Maximum size = 32768 bytes
 Minimum unused bits = 0
 Type = raw
 Read-only = false
 Fixed-size = false
 Maximum rows = 65535
 Current size = 72 bytes
 Current unused bits = 0
 Current row size = 12 bytes

Parameter Table: 0

Timestamp	cdh.DataPool.p...	cdh.DataPool.platf...	cdh.DataPool.plat...
Mon Jan 30 12:34:27 GMT 2017	Eleven	4.001	-
Mon Jan 30 12:34:32 GMT 2017	Eleven	4.001	-
Mon Jan 30 12:34:37 GMT 2017	Eleven	4.001	-
Mon Jan 30 12:34:42 GMT 2017	Eleven	4.001	-
Mon Jan 30 12:34:47 GMT 2017	Eleven	4.001	-
Mon Jan 30 12:34:52 GMT 2017	Eleven	4.001	-
Mon Jan 30 12:34:57 GMT 2017	Eleven	4.001	-
Mon Jan 30 12:35:02 GMT 2017	Eleven	4.001	-
Mon Jan 30 12:35:07 GMT 2017	Eleven	4.001	-
Mon Jan 30 12:35:12 GMT 2017	Eleven	4.001	-
Mon Jan 30 12:35:17 GMT 2017	Eleven	4.001	-
Mon Jan 30 12:35:22 GMT 2017	Eleven	4.001	-

Parameter: cdh.tmtc.TMBeacon.enable

Data: B 1

Get Set Downlink Uplink

Invoke action: core.Storage.wipe

Argument: H

Invoke

Parameter block: core.Storage.channelContent

Parameter index in block: 1

First r... 0 Last r... 5 Res... 6 r... ?

Data: H 5888B19F408400040006593

Get Set Downlink Uplink

Transfers

- CFDP Ack Uplink Completed successfully
- CFDP Ack Uplink Completed successfully
- CFDP Ack Uplink File transfer cancelled
- CFDP Ack Uplink Completed successfully
- CFDP Ack Uplink Completed successfully
- CFDP Ack Uplink Completed successfully
- CFDP Ack Uplink Completed successfully
- CFDP Ack Uplink Completed successfully

Remove successful transfers on completion

Suspend Resume Abort Clear

Aggregation builder

Parameter	First Row	Last Row	Width
comms.CFDP.nakLimit	0	0	0
comms.CFDP.ackLimit	0	0	0
comms.CFDP.ackTimeout	0	0	0
comms.CFDP.nakTimeout	0	0	0
comms.CFDP.inactivityTimeout	0	0	0
comms.PacketStreamY.spacelinkTxEnable	0	0	0
comms.PacketStreamY.spacelinkRxEnable	0	0	0
comms.PacketStreamY.umbilicalTxEnable	0	0	0
comms.PacketStreamY.umbilicalRxEnable	0	0	0
comms.PacketStreamY.umbilicalTimeout	0	0	0
comms.TmSpaceDataLink.isOIDEnabled	0	0	0

CFDP Uplink

Destination: 0

Source file: ..\..\TMTCLab\CFDP\chn1.dat Choose

Destination file: /chn1.dat

Put

CFDP Uplink

Destination: 0

Source file: ..\..\TMTCLab\CFDP\out3.dat Choose

Destination file: /out3.dat

Put

Instance of component type: subsys.OBT

component OBT

Component ID: 4 (0x0004)

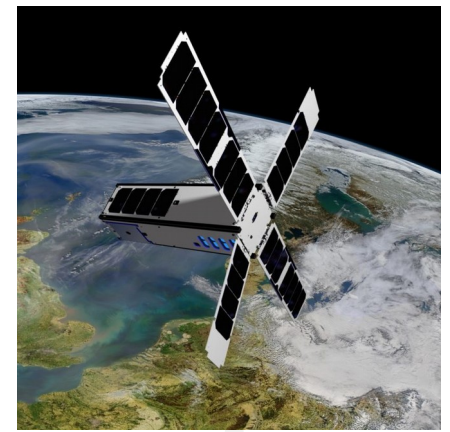
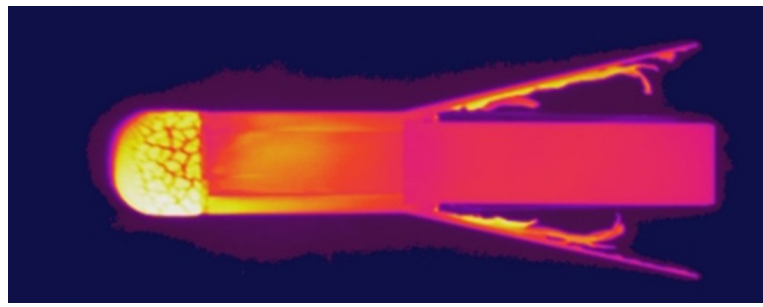
The OBT component provides an interface to the onboard

Time	Level	Source	Message
2017-01-30 12:35:22	DEBUG	gen1_protocol.spacepacket.SpacePacketProtocolEntity	received indication
2017-01-30 12:35:22	DEBUG	gen1_protocol.tmspacedatalink.TmSpaceDataLinkProtocolEntity	Packet complete
2017-01-30 12:35:22	INFO	gen1_protocol.tmspacedatalink.TmSpaceDataLinkProtocolEntity	Space packet length: 867
2017-01-30 12:35:22	DEBUG	gen1_protocol.tmspacedatalink.TmSpaceDataLinkProtocolEntity	Got space packet header
2017-01-30 12:35:22	DEBUG	gen1_gui.hk.HkChecks	Logging SID
2017-01-30 12:35:22	ERROR	gen1_gui.hk.HkChecks	Parameter cdh.DataPool.platform.DummySubsys1.dummyParam16 violated check 1 - Value of 16 is above HARD limit of 10

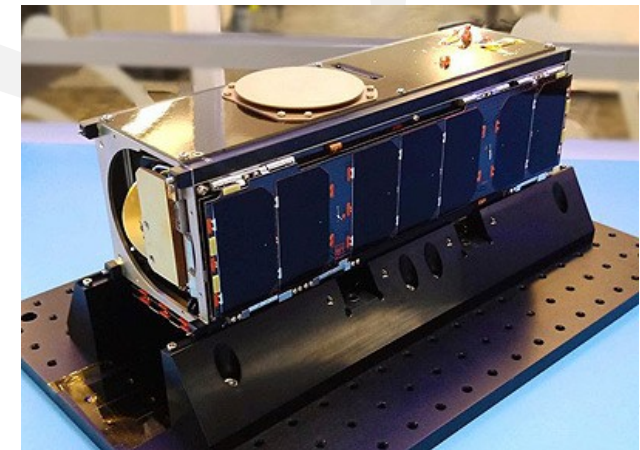
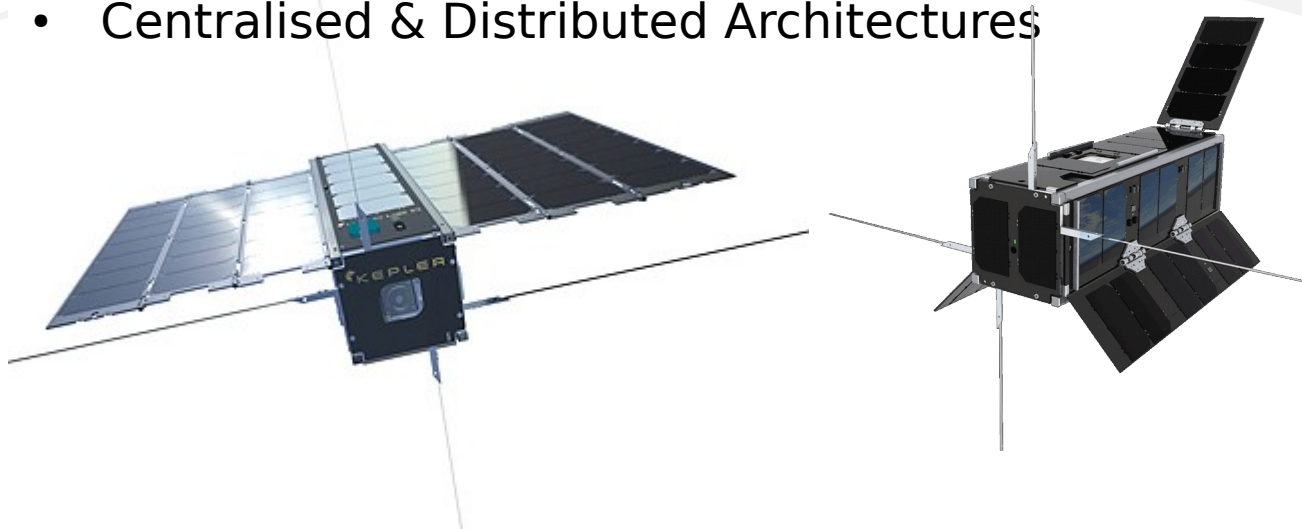
TCP/IP client connected



Results



- 10 Diverse Cubesat missions within the company
 - Small team, rapid development
- 8 Satellites currently operating with gen1 technology
- Application Diversity
 - Commercial, Science, Education
- Platform & Technology Diversity
 - AAC-Clyde, Pumpkin, GOM Space, Custom
 - Linux, RTEMS, FreeRTOS, Bare metal
 - ARM, x86, Leon-2, SPARC, MSP430
- Centralised & Distributed Architectures



Conclusion

- Move towards a more unified treatment of space and ground through a shared functional architectural model
 - Facilitates **rapid iteration**
 - Raises the **semantic level** of the space / ground interface for improved **operability, autonomy** and **scalability**

Bright Ascension Ltd
www.brightascension.com
enquiries@brightascension.com
+44 (0) 1382 602041

Come and see us at
Stand **B6**

Backup slides



Interface to a component

- **Actions**
 - Commands/operations/methods
 - Can 'invoke' from onboard or ground
- **Parameters**
 - Data fields/attributes
 - Can 'get' or 'set' (if not read-only) from onboard or ground
- **Exceptions**
 - Status code
 - Returned by synchronous operations (e.g. get/set/invoke)
 - Indicates abnormal or unusual operation, usually an error
- **Events**
 - Issued asynchronously and usually logged onboard
 - Indicates abnormal or unusual operation, usually an error
- This is the interface that is “seen” from ground
- Components can be grouped together to form a logical hierarchy

The GenerationOne approach

- Provide common functions off-the-shelf as **software components**
 - Model captures the available component types
- Flight software rapidly **assembled** from these components
 - Within the “glue” of the GenerationOne framework
 - Model captures the architecture resulting from component assembly
- The software can therefore be **tailored for the mission**
 - Which components are used
 - How many of each type of component is used
 - The ways in which components are connected together
 - Custom components for the mission
- Component framework is relatively simple
 - Not flying a lot of unnecessary complexity
- Component interfaces follow regular structure
 - Limits issues caused by component interactions
- **Reduce** development **time, cost** and **risk**
 - Software **available earlier** to support AIT

Component interface from ground

The screenshot displays the TMTc Lab - GenerationOne OBSW interface. The main window is titled "Spacecraft Explorer" and shows a hierarchical tree of components. The tree is organized into folders: demo_linux, cdh, logging, tmtc, comms, pus, and core. The "lifeTime" parameter is selected under the "DataPool" component. The right-hand pane shows the configuration for "Parameter lifeTime", including a description of its function and a data entry field. The "Data" field contains the hexadecimal value "0000000C". Below the data field are buttons for "Get", "Set", "Downlink", and "Uplink". A status message at the bottom of the pane reads "Parameter accessed successfully".

Component

Action

Parameter

Parameter lifeTime

param uint32 lifeTime

Parameter ID: 3584 (0x0E00)

How long a pooled parameter can be considered valid in seconds.

The if the time since the DataPool component was last refreshed exceeds the lifeTime value, then the poolParameters will be considered invalid and accessing the parameters without refreshing will cause a read-through. An OBT component is required to calculate whether a pooled parameter has become invalid.

Current size: Fixed size Query

Parameter Data

First row: Last row: Resize

Data:

Data should be specified in hexadecimal with no prefixes. Spaces should be used to separate rows.

Get Set Downlink Uplink

Parameter accessed successfully Clear

Time	Level	Component	Filename	Line	Message
Wed 14:01:59	DEBUG	framework	LoggerCore.c	111	Channel has incorrect record size, reformatting
Wed 14:01:59	DEBUG	framework	FileStoragePr...	298	Overwriting
Wed 14:01:59	DEBUG	framework	FileStoragePr...	298	Overwriting
Wed 14:01:59	INFO	framework	Deployment.c	36	Deployment initialisation successful

Connected

Components and services

- A component interface and a service are on different semantic levels
- Component interfaces (as defined by the OSRA)
 - Describe *what*
 - For example
 - An attribute
 - An event
 - Bindings bind a *thing* to a *thing*
 - e.g. an attribute to an attribute
- Service interfaces (as defined by MO and, to a lesser extent, SOIS)
 - Describe *how*
 - For example
 - Parameter service
 - Event service
 - Bindings bind a *mechanism* to a *mechanism*

