Creating a Setup to Assess the Use of Virtual Reality for Mission Control

Felix Sittner, Cedric Liman, Gino Schulze, Hans Schülein, Jan Schmieder,
Jan Tischhöfer, Marlene Busch & Sergio Montenegro

The Team

InnoCube

- Info8-ILR @JMU Würzburg

  → Software Team
  - Onboard Software
  - Control Room Software
- IRAS @TU ~~Brunswick~~ Berlin
- DLR Bremen

“Serious Mission”

Virtual Control Room (VCR)

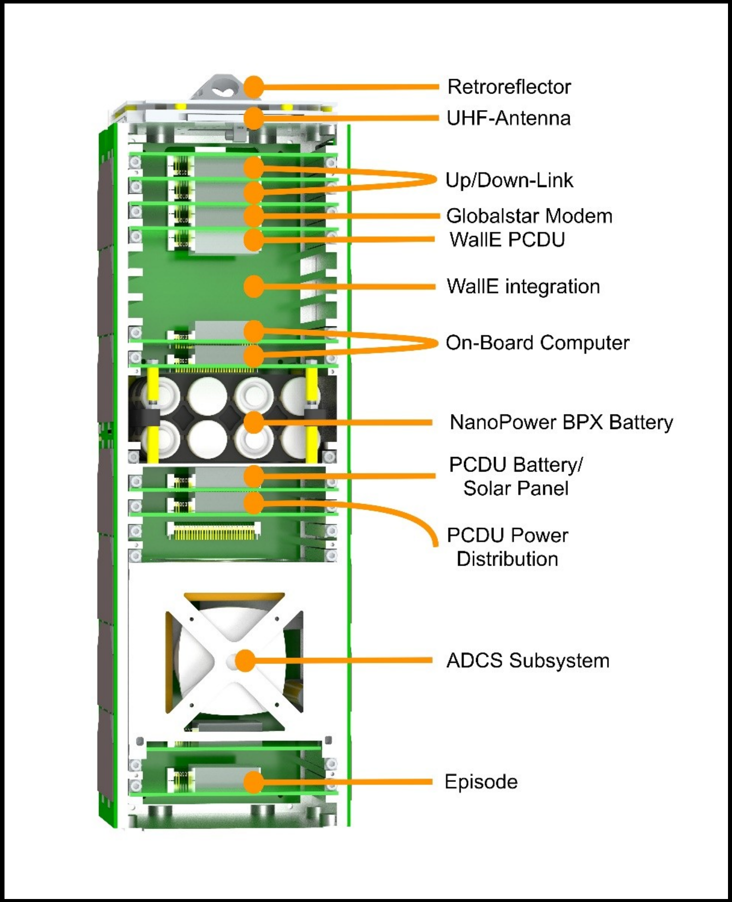
- Student Thesis Projects
- Student Assistants
- Internships

“Fun educational project”

The Mission

InnoCube

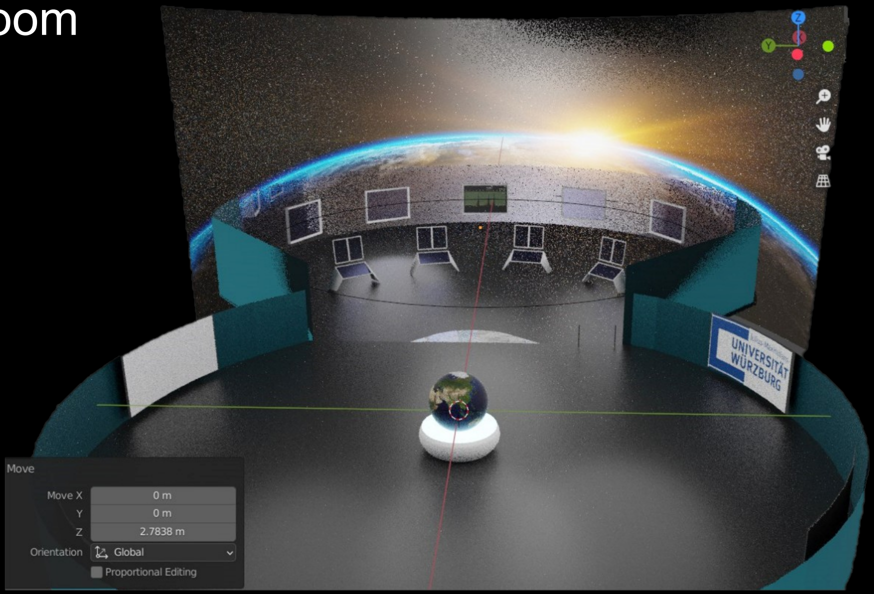
- 3U Cubesat
- Technology Demonstration:
  - **SKITH** wireless satellite bus
  - **WALL#E** structural battery
- Scientific Payload:
  - **EPISODE** SDR GNSS
- Launches Spring 2023



Motivation

Why build a Virtual Control Room (VCR) ?

- Fun project for students!
- Less hardware needed: One computer, one VR headset
- Virtual environment (“scene”) reconfiguration is easy & cheap
- We could not use our real control room anymore due to pandemic



Move

Move X 0 m
Y 0 m
Z 2.7838 m
Orientation Global
□ Proportional Editing

The Idea

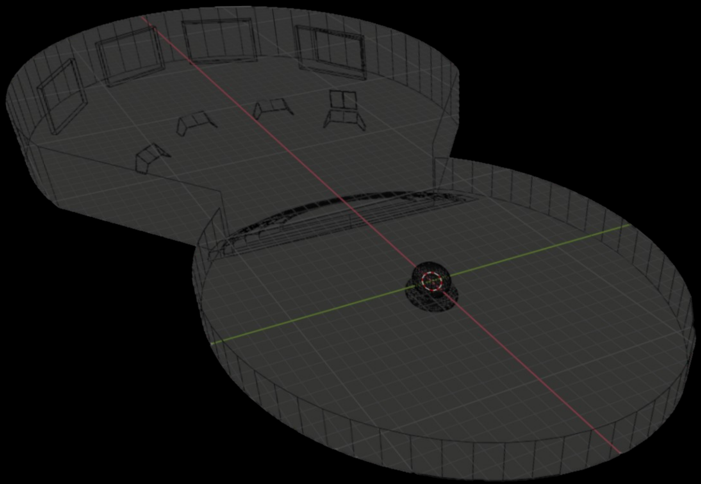
Coordinate InnoCube GS SW and VCR development and find out what can be achieved using the available technology!

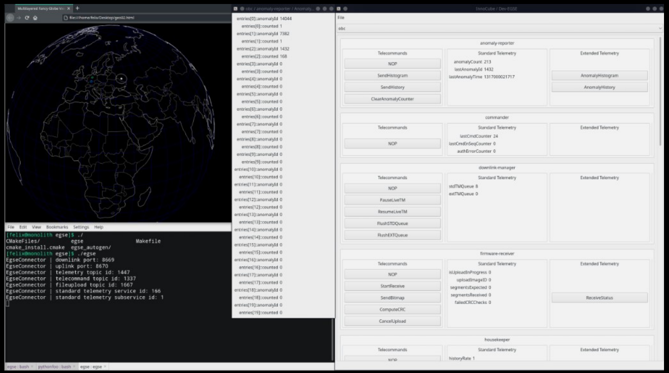
Requirements:

1) Create GUIs that can be used normally and within VR
2) One GS backend software framework
3) Can connect either to dev/simulation or real satellite
4) Integrate with our exiting setup:
   - Satellite / simulation: *Rodos* OS
   - Exchange data using the *Corfu* framework
   - Use Corfu configuration files to create GUI views
5) Test, evaluate & adapt VR environment to support ground crew (and reduce simulation sickness)

The Tasks

- Create the VCR scene
- Develop control room software
- Plan & furnish real control room
- Assign software views to displays
- Configure for individual operators
- Test with users
- Improve Software & VCR design





```
[felix@monolith egse]$ ./
CMakeFiles/          egse                     Makefile
cmake_install.cmake  egse_autogen/
[felix@monolith egse]$ ./egse
EgseConnector | downlink port: 8669
EgseConnector | uplink port: 8670
EgseConnector | telemetry topic id: 1447
EgseConnector | telecommand topic id: 1337
EgseConnector | fileupload topic id: 1667
EgseConnector | standard telemetry service id: 166
EgseConnector | standard telemetry subservice id: 1
```

```
entries[0]::anomalyId 14044
  entries[0]::counted 1
entries[1]::anomalyId 7382
  entries[1]::counted 1
entries[2]::anomalyId 1432
  entries[2]::counted 168
entries[3]::anomalyId 0
  entries[3]::counted 0
entries[4]::anomalyId 0
  entries[4]::counted 0
entries[5]::anomalyId 0
  entries[5]::counted 0
entries[6]::anomalyId 0
  entries[6]::counted 0
entries[7]::anomalyId 0
  entries[7]::counted 0
entries[8]::anomalyId 0
  entries[8]::counted 0
entries[9]::anomalyId 0
  entries[9]::counted 0
entries[10]::anomalyId 0
  entries[10]::counted 0
entries[11]::anomalyId 0
  entries[11]::counted 0
entries[12]::anomalyId 0
  entries[12]::counted 0
entries[13]::anomalyId 0
  entries[13]::counted 0
entries[14]::anomalyId 0
  entries[14]::counted 0
entries[15]::anomalyId 0
  entries[15]::counted 0
entries[16]::anomalyId 0
  entries[16]::counted 0
entries[17]::anomalyId 0
  entries[17]::counted 0
entries[18]::anomalyId 0
  entries[18]::counted 0
entries[19]::anomalyId 0
  entries[19]::counted 0
```

anomaly-reporter

Telecommands	Standard Telemetry	Extended Telemetry
NOP SendHistogram SendHistory ClearAnomalyCounter	anomalyCount 213 lastAnomalyId 1432 lastAnomalyTime 1317000021717	AnomalyHistogram AnomalyHistory

commander

Telecommands	Standard Telemetry	Extended Telemetry
NOP	lastCmdCounter 24 lastCmdInSeqCounter 0 authErrorCounter 0	

downlink-manager

Telecommands	Standard Telemetry	Extended Telemetry
NOP PauseLiveTM ResumeLiveTM FlushSTDQueue FlushEXTQueue	stdTMQueue 8 extTMQueue 0	

firmware-receiver

Telecommands	Standard Telemetry	Extended Telemetry
NOP StartReceive SendBitmap ComputeCRC CancelUpload	isUploadInProgress 0 uploadImageID 0 segmentsExpected 0 segmentsReceived 0 failedCRCChecks 0	ReceiveStatus

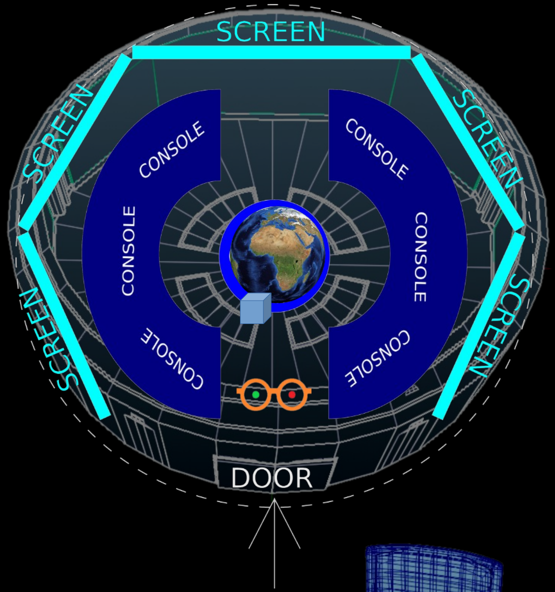
housekeeper

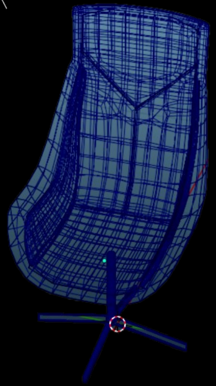
Telecommands	Standard Telemetry	Extended Telemetry
NOP	historyRate 1	

Creating the
Virtual Reality Scene

Creating the VR Scene

- Design & floor plan
- Model 3D objects & room in Blender
- Add textures
- Create Scene in Unity3D with the objects
- Add XR-rig to represent the user in the scene
- Add functionality / interactivity, e.g.,
    - Interactions (add triggers & link scripts to objects)
    - Set up browser windows to display control room software GUI
    - Update object properties with external data

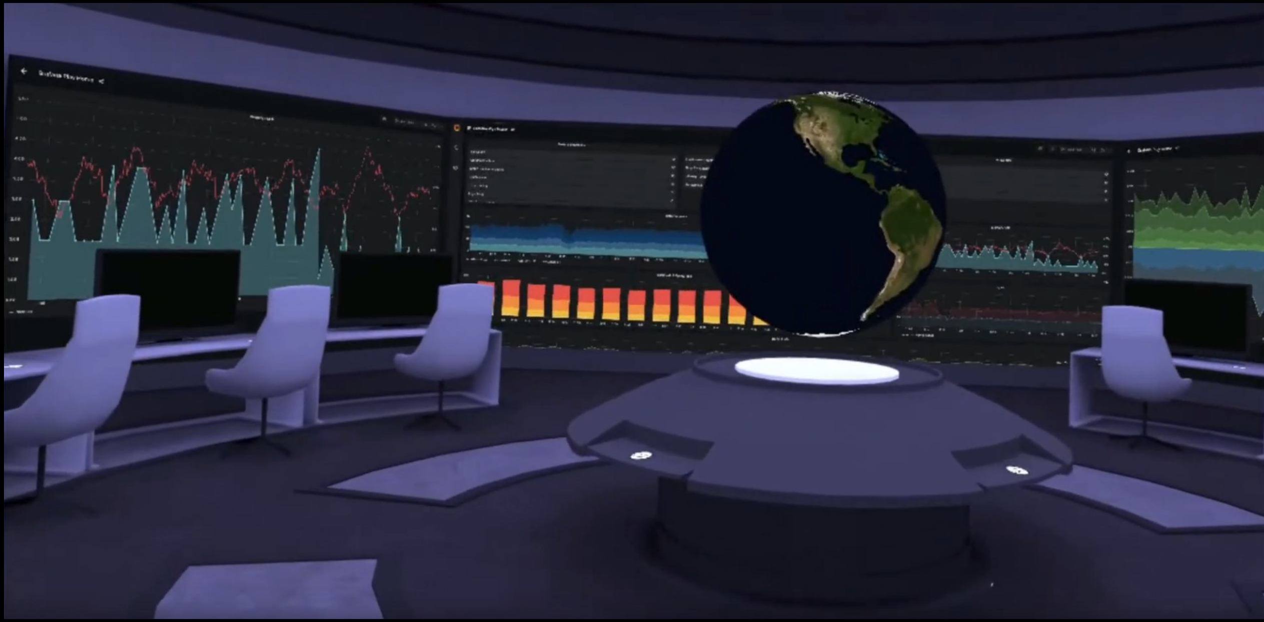




First VCR Design

Attempt to create a futuristic virtual representation of a normal control room:

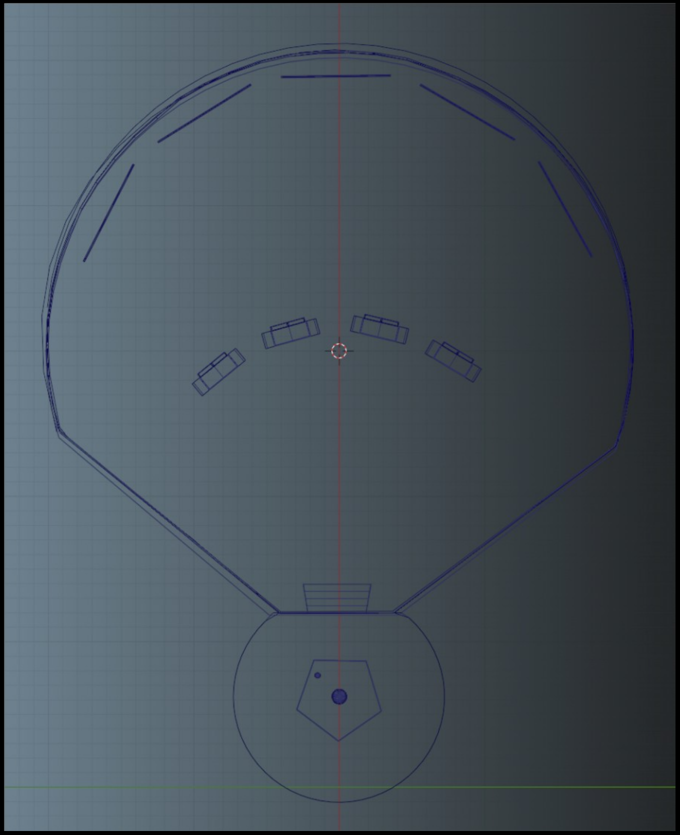
- Intended to be used by multiple operators working in teams
- Round room with two rows of consoles
- One console / row per operator
- Huge displays on walls
- Central globe showing the satellite position
- Calm color scheme
- Operators may walk, but usually sit



Second VCR Design

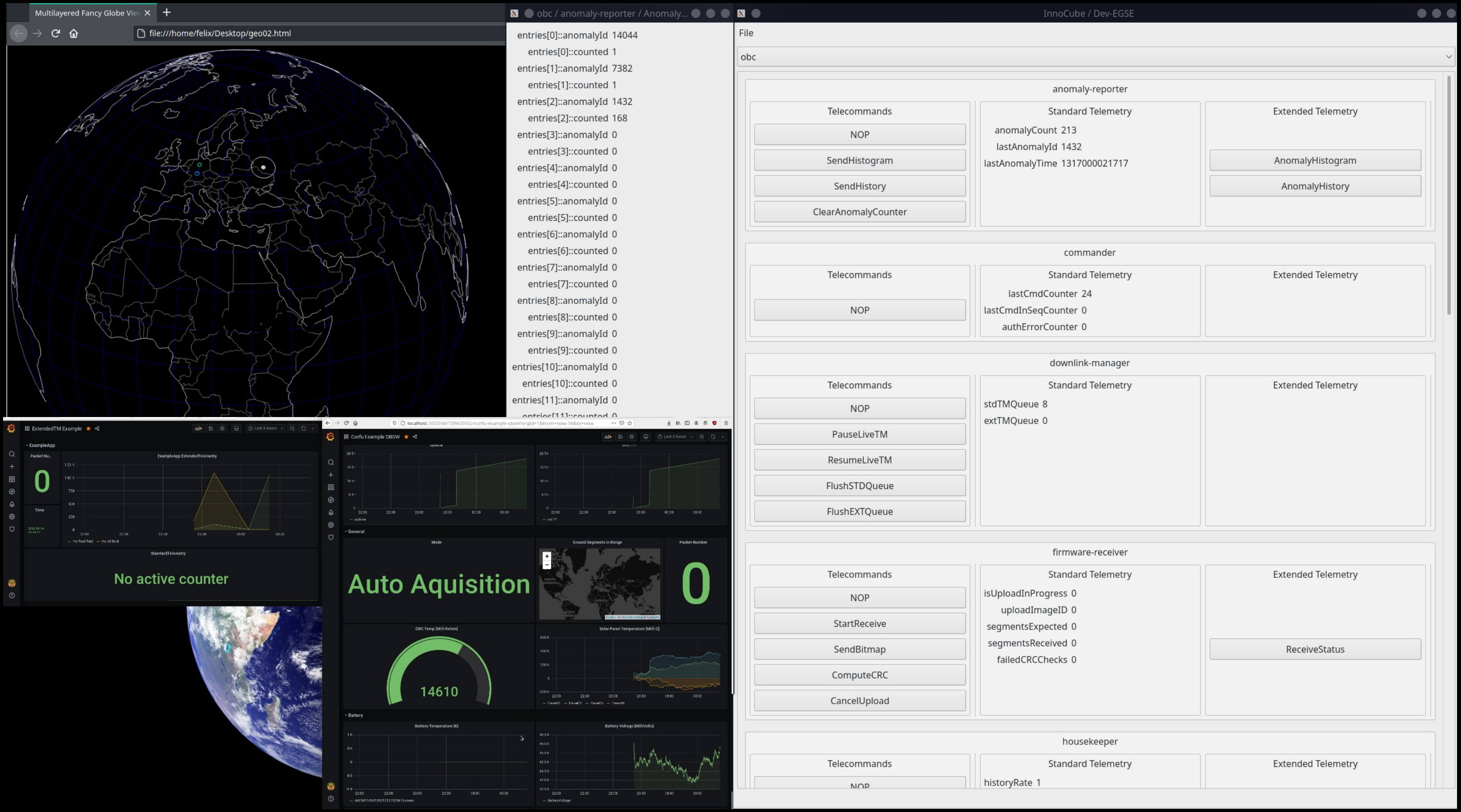
Improving upon the first VCR design.

- Intended to be used by a single operator
- User not globe at the center of the room
- Two connected rooms:
  - One main work area
  - One room with the 3D globe
- 4 consoles in a half circle
- User can grab / move large screens
- User can stand or sit



Creating the
Control Room Software

The early GUIs – Dev-EGSE, D3-Geo & Grafana



```
entries[0]::anomalyId 14044
  entries[0]::counted 1
entries[1]::anomalyId 7382
  entries[1]::counted 1
entries[2]::anomalyId 1432
  entries[2]::counted 168
entries[3]::anomalyId 0
  entries[3]::counted 0
entries[4]::anomalyId 0
  entries[4]::counted 0
entries[5]::anomalyId 0
  entries[5]::counted 0
entries[6]::anomalyId 0
  entries[6]::counted 0
entries[7]::anomalyId 0
  entries[7]::counted 0
entries[8]::anomalyId 0
  entries[8]::counted 0
entries[9]::anomalyId 0
  entries[9]::counted 0
entries[10]::anomalyId 0
  entries[10]::counted 0
entries[11]::anomalyId 0
  entries[11]::counted 0
```

anomaly-reporter

Telecommands	Standard Telemetry	Extended Telemetry
NOP	anomalyCount 213	
SendHistogram	lastAnomalyId 1432	AnomalyHistogram
SendHistory	lastAnomalyTime 1317000021717	AnomalyHistory
ClearAnomalyCounter		

commander

Telecommands	Standard Telemetry	Extended Telemetry
	lastCmdCounter 24	
NOP	lastCmdInSeqCounter 0	
	authErrorCounter 0	

downlink-manager

Telecommands	Standard Telemetry	Extended Telemetry
NOP	stdTMQueue 8	
PauseLiveTM	extTMQueue 0	
ResumeLiveTM		
FlushSTDQueue		
FlushEXTQueue		

firmware-receiver

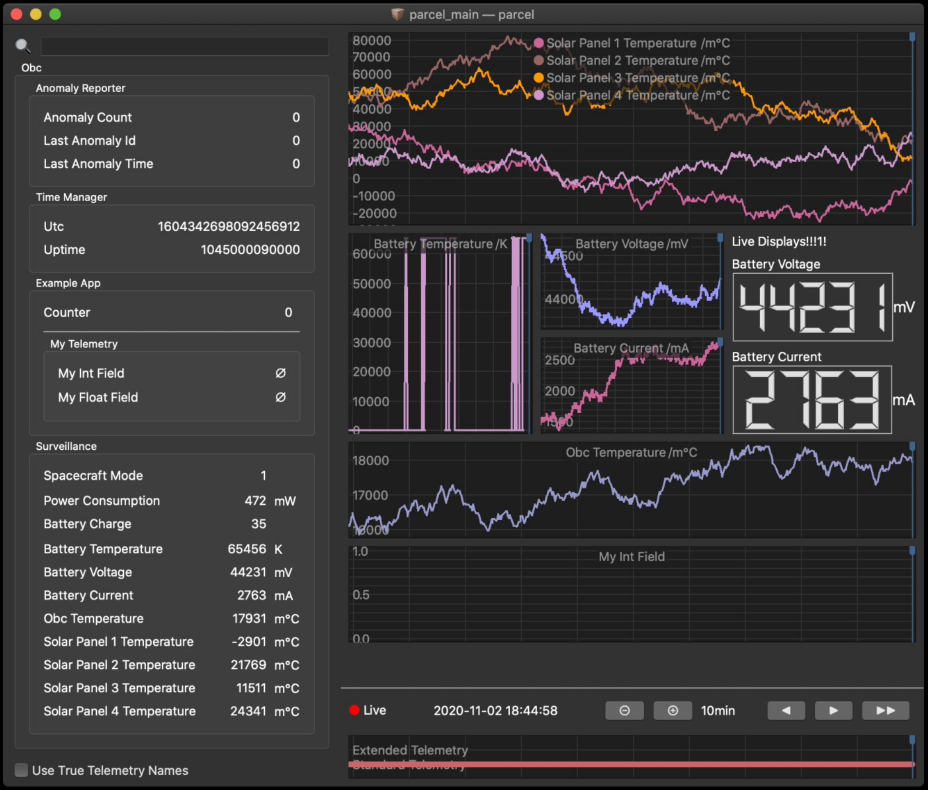
Telecommands	Standard Telemetry	Extended Telemetry
NOP	isUploadInProgress 0	
StartReceive	uploadImageID 0	
SendBitmap	segmentsExpected 0	ReceiveStatus
ComputeCRC	segmentsReceived 0	
CancelUpload	failedCRCChecks 0	

housekeeper

Telecommands	Standard Telemetry	Extended Telemetry
NOP	historyRate 1	

Improving the GUI – BA-Thesis: Parcel

- Python Application for Recording a Corfu EGSE Link
- Works with native Corfu telemetry messages (as sent by InnoCube)
- Creates GUI elements from config
- Adds telemetry visualizations
- Adds search function
- Allows to replay telemetry
- GUI configuration in python script, which builds Parcel executable



Obc

Anomaly Reporter

Anomaly Count	0
Last Anomaly Id	0
Last Anomaly Time	0

Time Manager

Utc	1604342698092456912
Uptime	1045000090000

Example App

Counter	0

My Telemetry

My Int Field	∅
My Float Field	∅

Surveillance

Spacecraft Mode	1	
Power Consumption	472	mW
Battery Charge	35	
Battery Temperature	65456	K
Battery Voltage	44231	mV
Battery Current	2763	mA
Obc Temperature	17931	m°C
Solar Panel 1 Temperature	-2901	m°C
Solar Panel 2 Temperature	21769	m°C
Solar Panel 3 Temperature	11511	m°C
Solar Panel 4 Temperature	24341	m°C

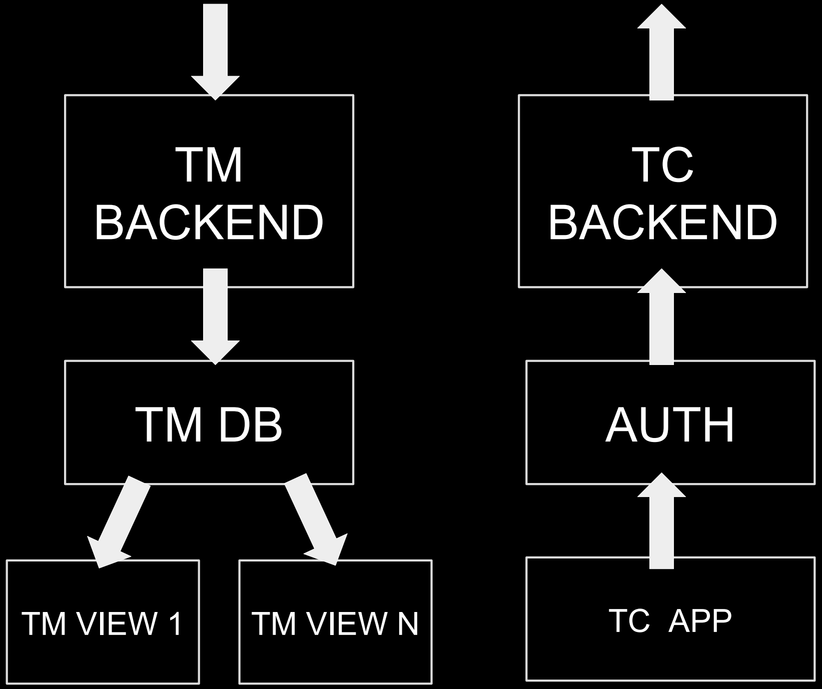
□ Use True Telemetry Names

The Redesign

Handle TM and TC separately!

Divide into frontend and backend!

- Backend:
  - Run natively on GS computer
  - Communicates with satellite/simulation
  - Stable / Reliable
- Frontend:
  - Apps packaged into containers
  - Fast setup & development



Data Flow

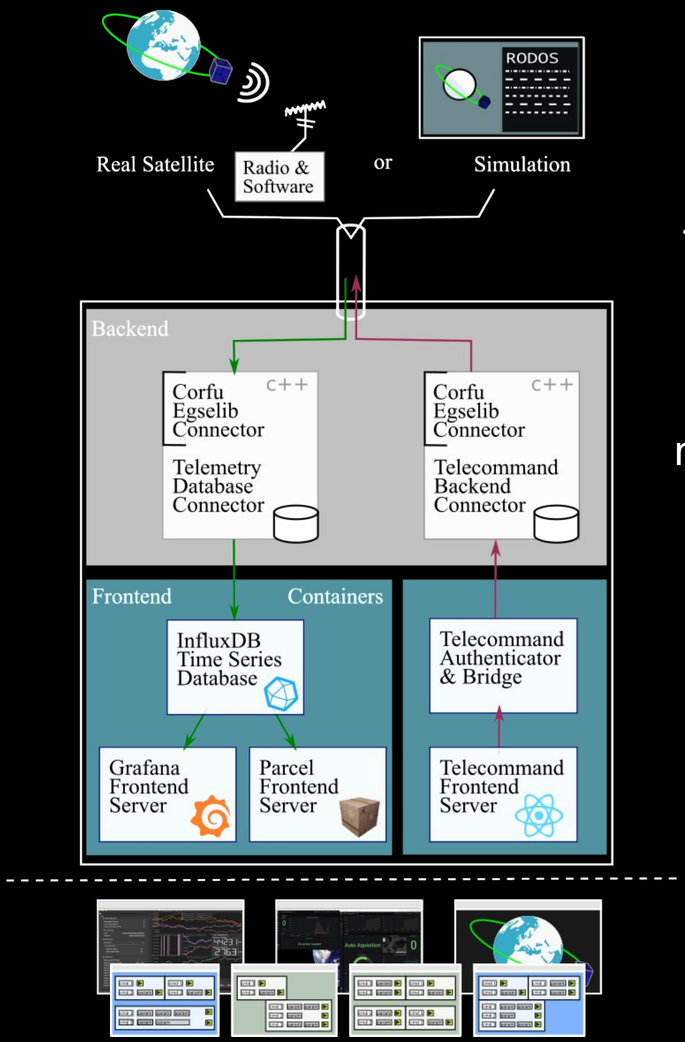
Telemetry

TM message is received & forwarded to control room SW

Backend stores raw data, extracts telemetry & pushes it into database

Frontend software pulls Data from db & serves visualizations

Operators access telemetry data displays via browser



Telecommands

TC message is sent to uplink

Backend assembles TC message object, stores a copy & forwards TC

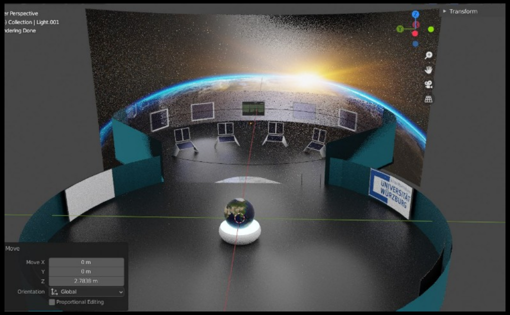
Frontend server checks credentials & forwards cmd data to the backend

Operator enters command in the web interface

The Components

Backend:

- EgseLib, DBs, Connectors
- Container Setup & Routing
- Secure connection

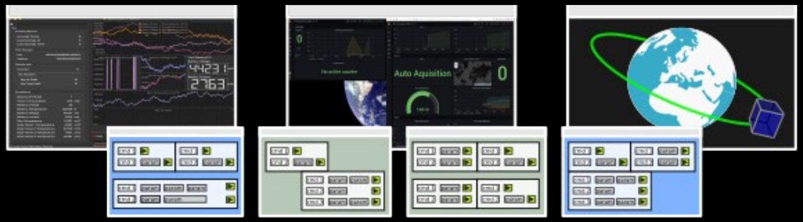


VR environment:

- Control room scene design
- VR interaction design
- VR-only assets

Frontend servers & software:

- Grafana
- Globe / Map view
- Parcel telemetry viewer
- Telecommand web GUI



Normal control room / office:

→ Views should work with various computers & display devices

The Resources

Web Frontends:

- Grafana
- D3.js & D3-geo
- InfluxDB Client JS
- React

Backend:

- Docker
- Ubuntu
- InfluxDB

Virtual Control Room

- Blender
- Unity3D
- UnityBrowser

Rodos

- Rodos on GitLab & RodosVM
- Rodos Tutorial on Youtube

For access to the software developed at our chair, such as Corfu, Parcel & the VCR project, send us an email.

The Fine Print

Contact:

Julius-Maximilians-Universität Würzburg
Department of Computer Science
Chair VIII: Aerospace Information Technology
Josef-Martin-Weg 52/2
D-97074 Würzburg

Phone: +49 931 3188786
Email: felix.sittner@uni-wuerzburg.de

Funding:

InnoCube is funded under support code 50RU2000 & 50RU2001
by the Federal Ministry for Economic Affairs and Energy
on the basis of a decision by the German Bundestag.

Gefördert durch:

Bundesministerium
für Wirtschaft
und Energie

aufgrund eines Beschlusses
des Deutschen Bundestages