Constellation

A Flexible Control and Data Acquisition Framework for Beamlines and Beyond

Stephan Lachnit
for the EDDA collaboration
11th Annual MT Meeting, Darmstadt

HELMHOLTZ

Prevas

LUND UNIVERSITY

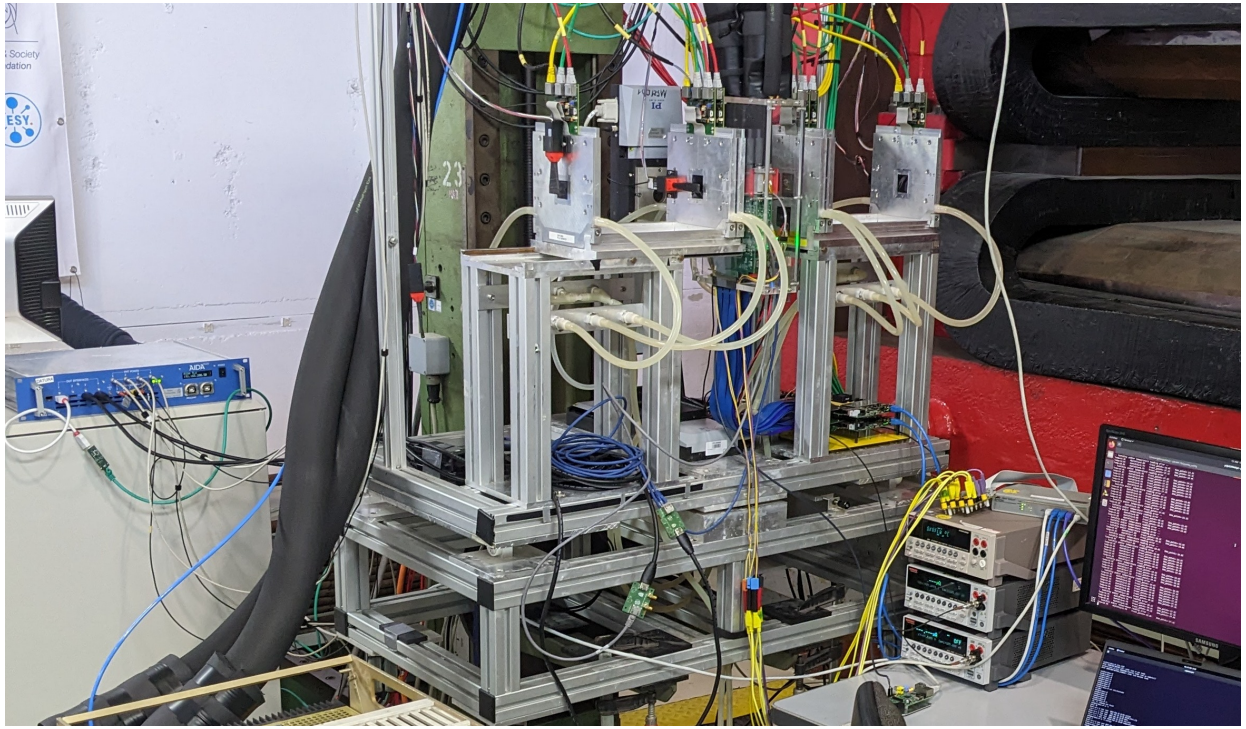
UH

BONN

DESY.

Beamline Example: Testbeam at DESY

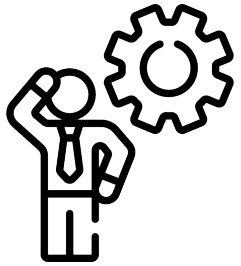
- Multiple devices required:
  - Device Under Test (**DUT**)
  - Beam **telescope** for tracking
  - Trigger Logic Unit (**TLU**) for synchronization
  - Timing Layer
  - Power supplies
  - Temperature sensors
  - ...
- Data Acquisition (DAQ) software spread across multiple computers



Example: Setup for the dSiPM test beam at DESY

Control Software at Beamline Setups

Challenges

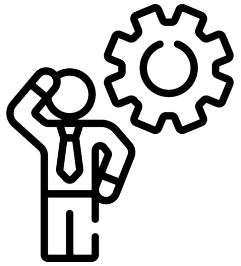


- Setup often deployed and used by students
- Different DAQ systems for different devices
- Frequent setup changes
- Integration depends on the facility
- Limited time for measurement (~ weeks)

=> Requires control software for synchronous operation

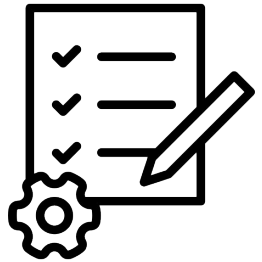
Control Software at Beamline Setups

Challenges



- Setup often deployed and used by students
- Different DAQ systems for different devices
- Frequent setup changes
- Integration depends on the facility
- Limited time for measurement (~ weeks)

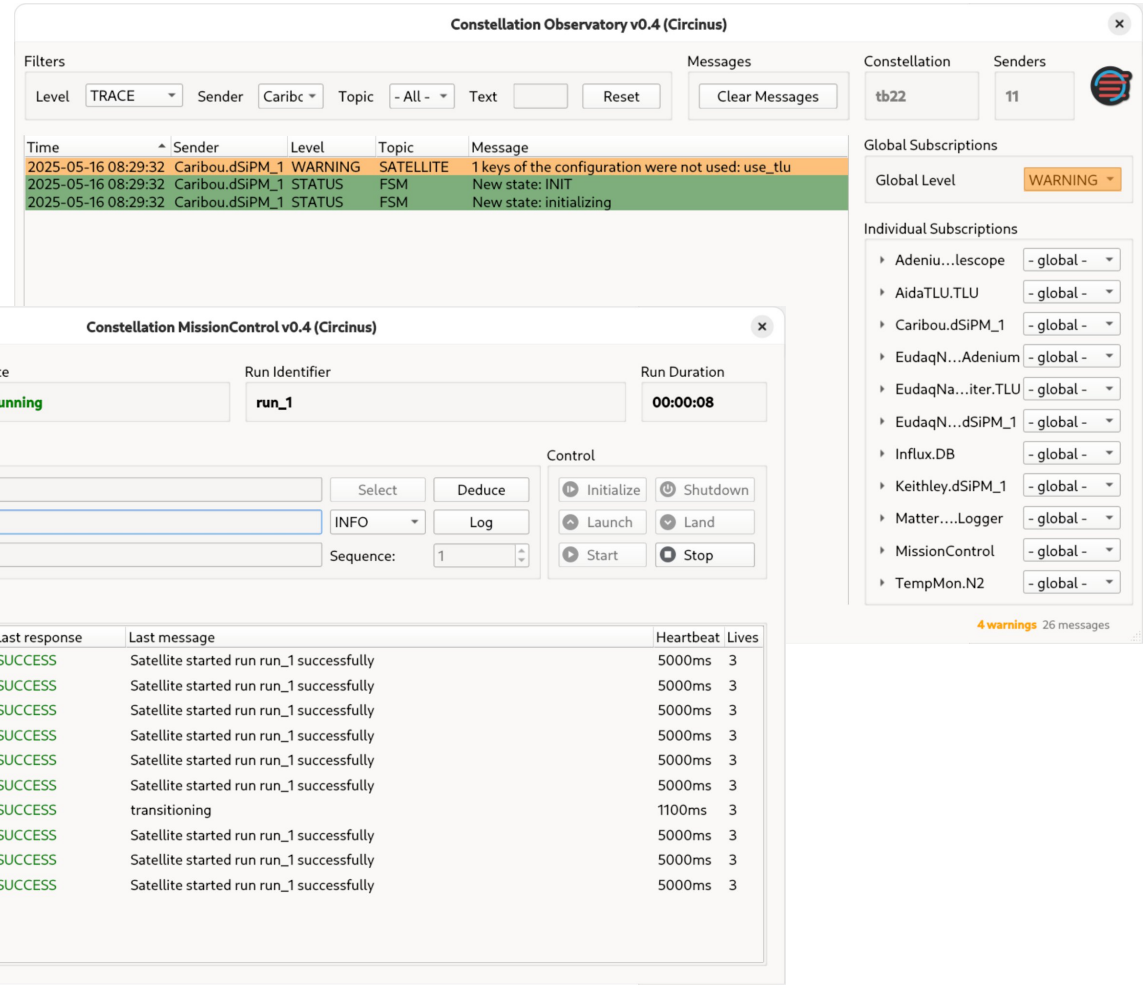
Requirements



- **Easy** to use
- **Fast integration** of new devices
- **Flexible** for many use-cases
- **Stable** operation and reliable error handling

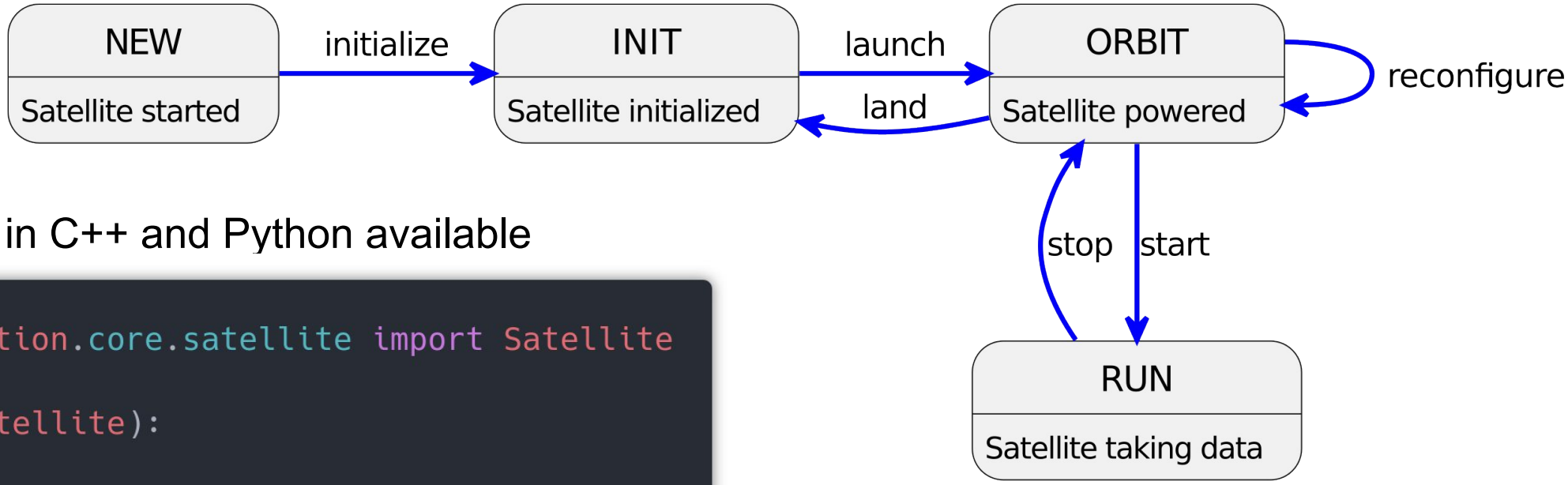
Constellation

- Control and Data Acquisition Framework for dynamic small-scale experimental setups
- Comes with *batteries included*:
  - Control & configuration
  - Logging & telemetry
  - Failure handling
  - Network discovery
  - Data transmission
  - Hot-swappable components
- Extensive documentation & tutorials



Satellites

- Participants in the data acquisition are called **satellites**
- Operate **autonomously** (no central point of failure)
- Operation governed by a Finite State Machine (**FSM**)



- Implementation in C++ and Python available

```python
from constellation.core.satellite import Satellite

class Dummy(Satellite):
    pass
```

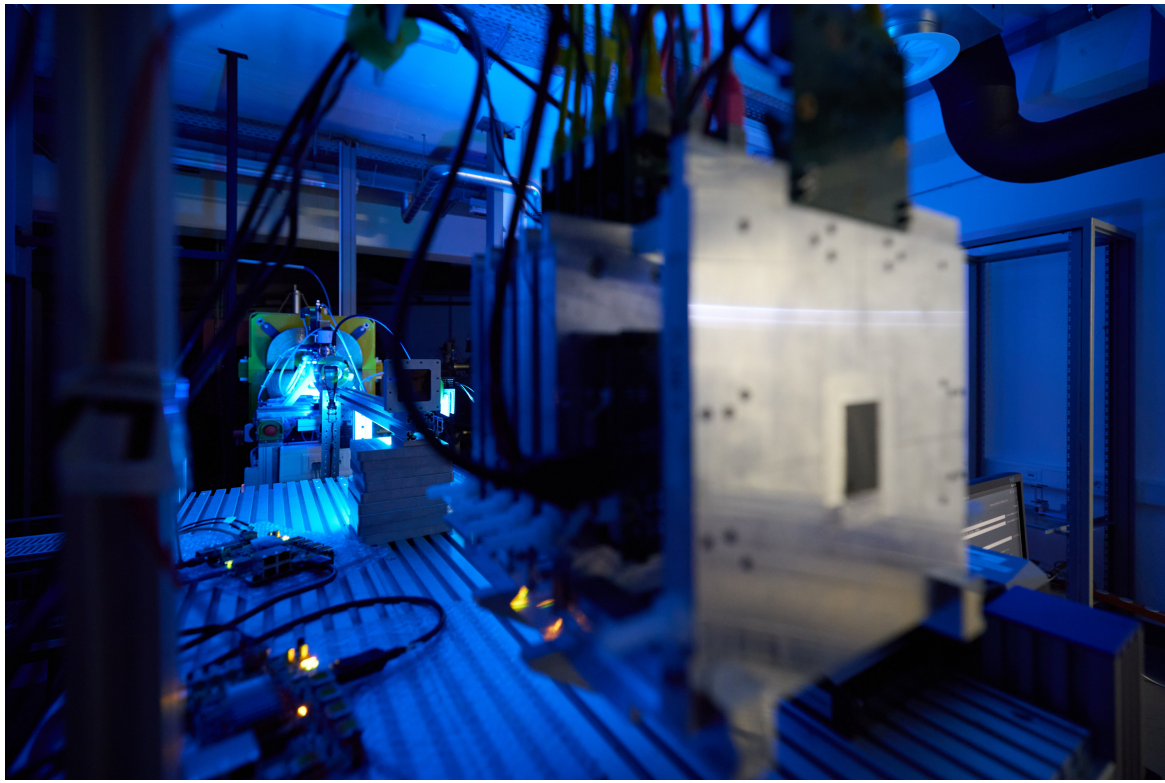
Application: Sensor Testing @ DESY II

- Testing of monolithic pixel sensor (e.g. H2M)
- Particle tracking with DESY-provided beam telescope
- Constellation highlights:
  - Backwards-compatible output to EUDAQ2 files
  - Mattermost notifications in case of failure
  - Online trigger rate monitoring via InfluxDB/Grafana

Constellation Mattermost notification on an Apple Watch (Lennart Huth)

Application: Beamline for Schools @ ELSA

- Beamlines for Schools (BL4S) at the ELSA accelerator (University of Bonn)
- Worldwide competition for students to carry out small experiments at a beamline
- One team @ ELSA with 8 days of beam
- Integration provided by University of Bonn
- Setup operated by high school students
- Constellation highlights:
  - Easy-to-use graphical user interfaces
  - Fast integration of the setup



BL4S setup at ELSA (Rasmus Partzsch)

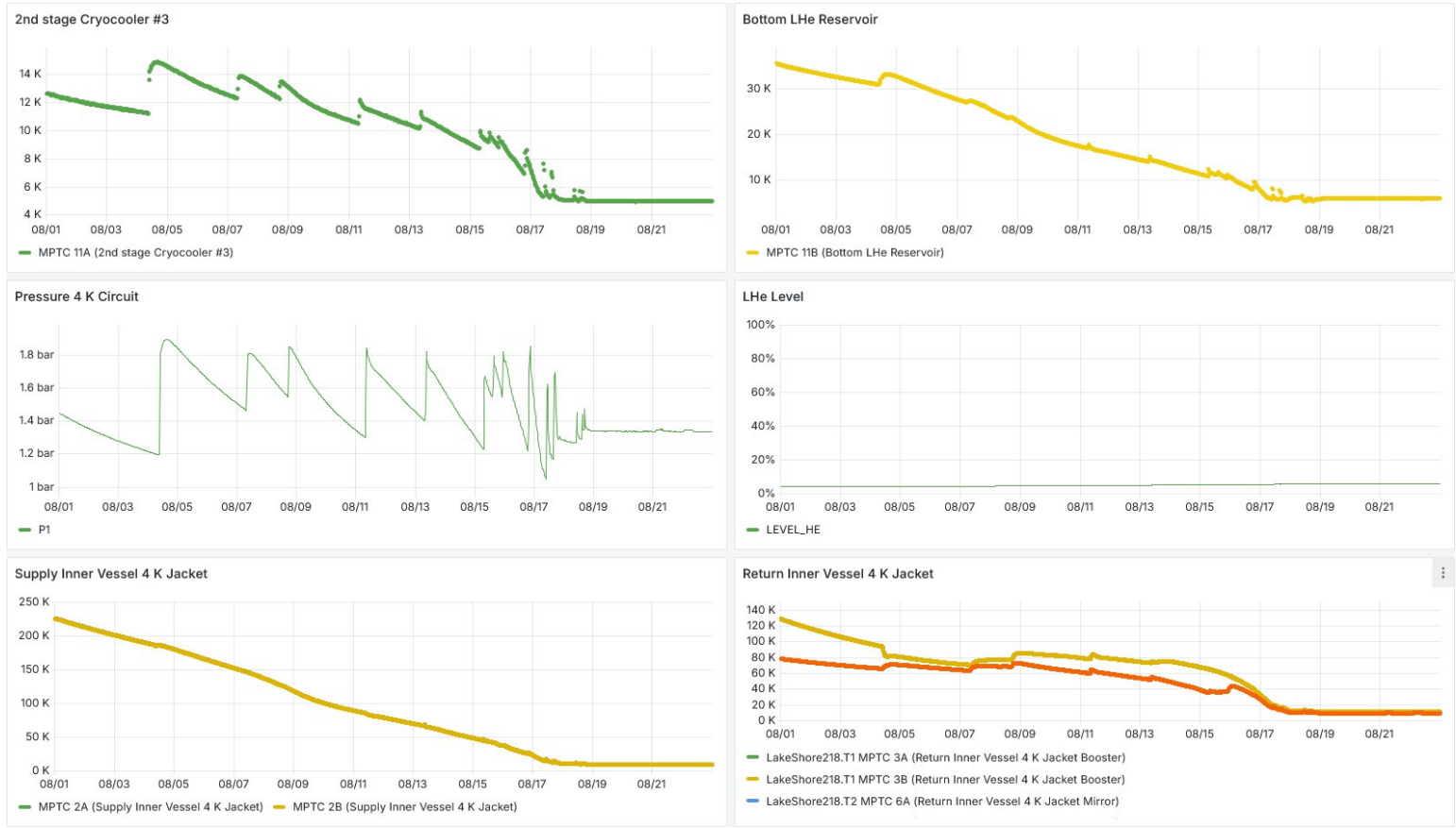
Application: Bent Crystal Characterization @ SPS

- Characterization of bent crystals for beam manipulation
- Timepix3 beam telescope @ SPS North Area
- Up to 14 motion stages for crystal movement and beam telescope alignment
- Constellation highlights:
  - Fast integration of the setup
  - InfluxDB/Grafana integration
  - Stable operation as user facility

BIPXL setup at SPS (Justus Braach)

Application: MADMAX Cryostat @ UHH

- Upcoming Dark Matter Experiment @ University of Hamburg
- Cryostat with liquid helium
- Constellation highlights:
  - InfluxDB/Grafana integration
  - Mattermost notifications
  - Stable operation



Online Monitoring of the MADMAX cryostat (David Leppla-Weber)

Project Status

- Fully open source under EUPL-1.2
- Available on DESY GitLab
- Currently version 0.6.1 (*Triangulum Australe*)
  - Mostly feature complete
  - Version 1.0 planned for spring 2026
- Continuously tested (~85% code coverage)
- Continuously deployed on PyPI, FlatHub and as Docker image

Patch Release 0.6.1 for Constellation Triangulum Australe

2025-10-08 Simon Spannagel , Stephan Lachnit , Hanno Perrey
Releases

This patch release contains a series of import bug fixes and minor improvements for our last release.

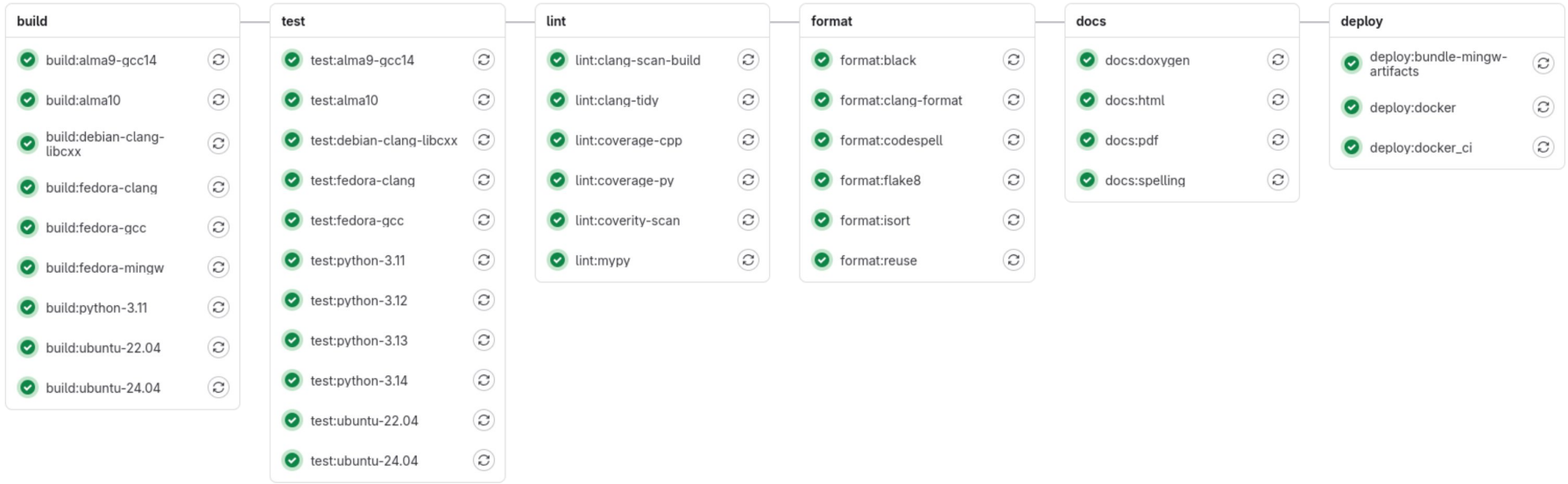
Read more ...

Constellation Preview Release 0.6

2025-09-24 Simon Spannagel , Stephan Lachnit
Releases

We are happy to announce the release of **Constellation version 0.6**, named **Triangulum Australe**.

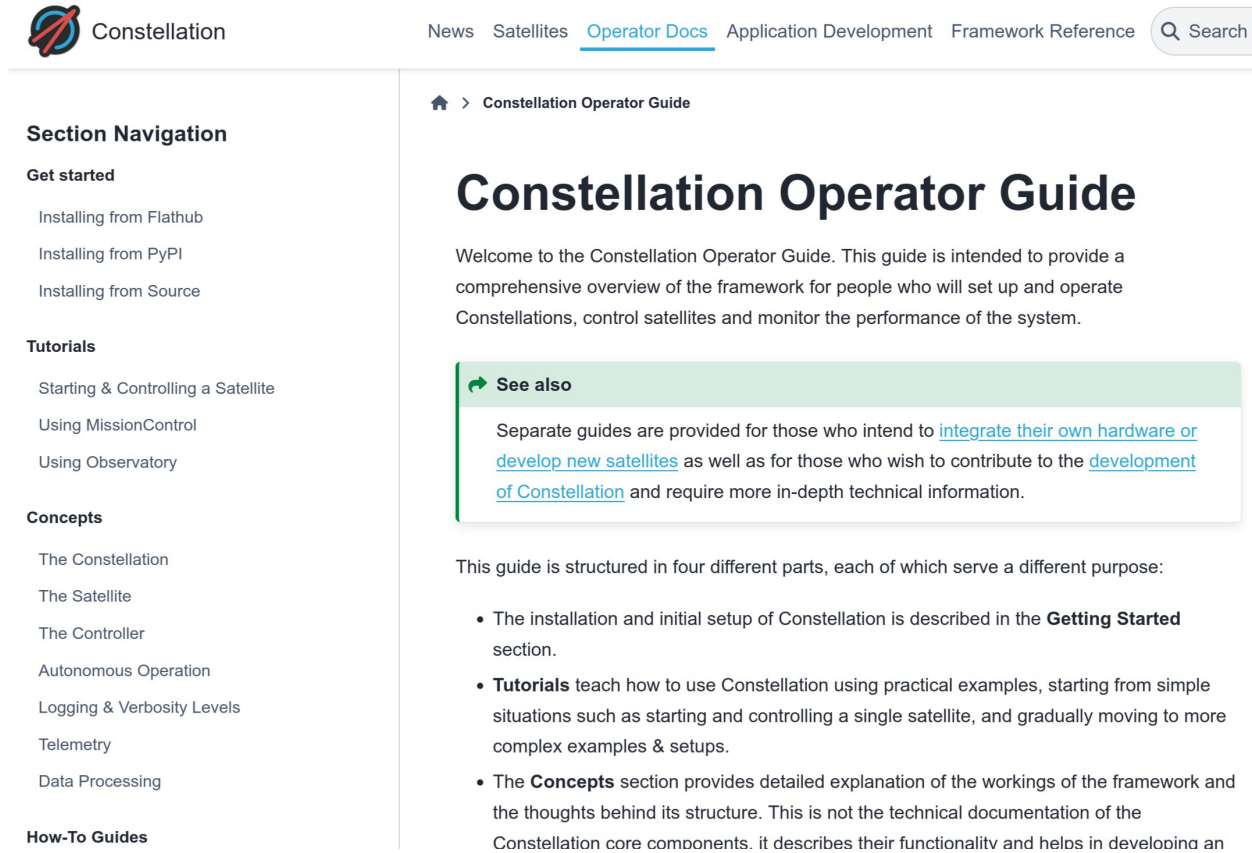
Read more ...



Summary

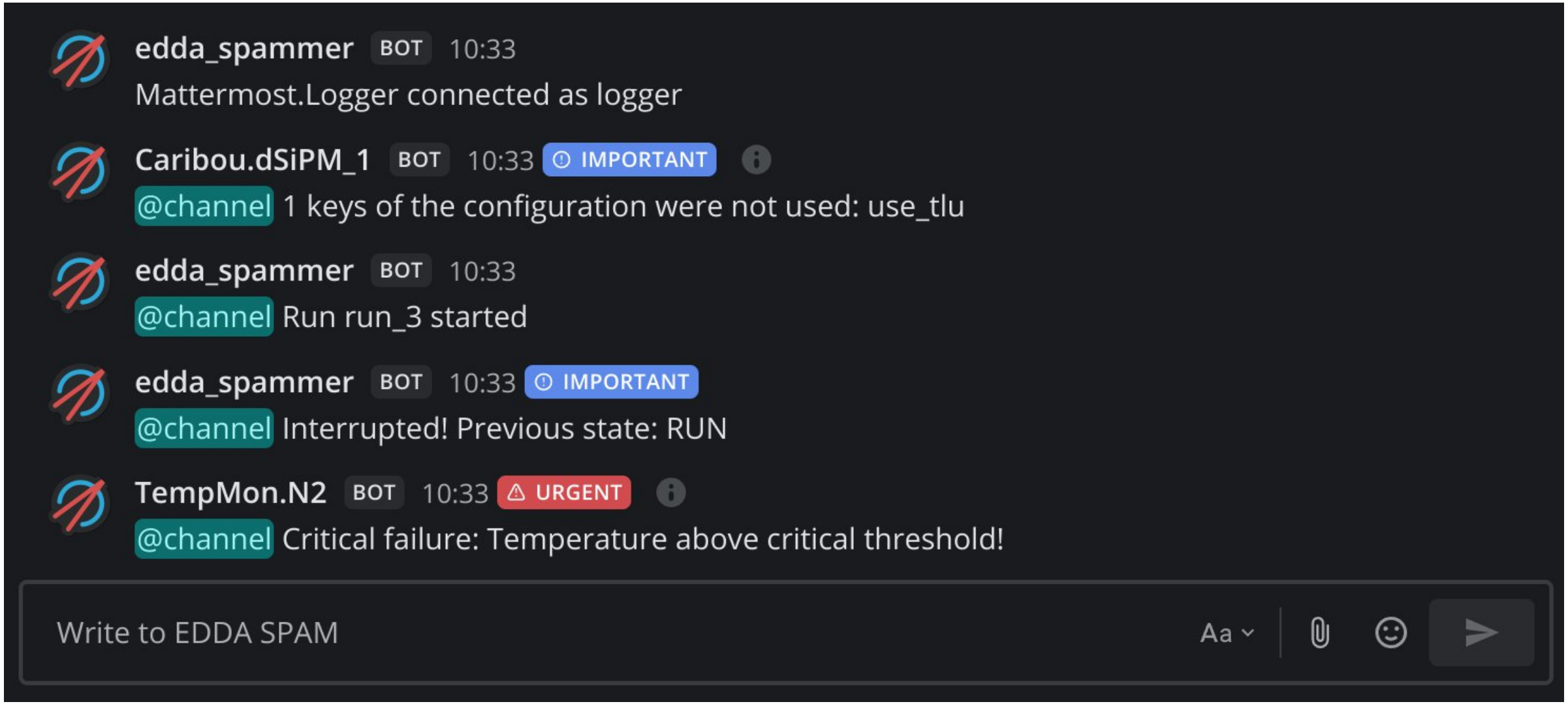
- Constellation is a control and data acquisition framework tailored towards beamline and lab setups
- Framework with *batteries included*
- Extensive documentation
- Used in several beamlines and setups

https://constellation.pages.desy.de



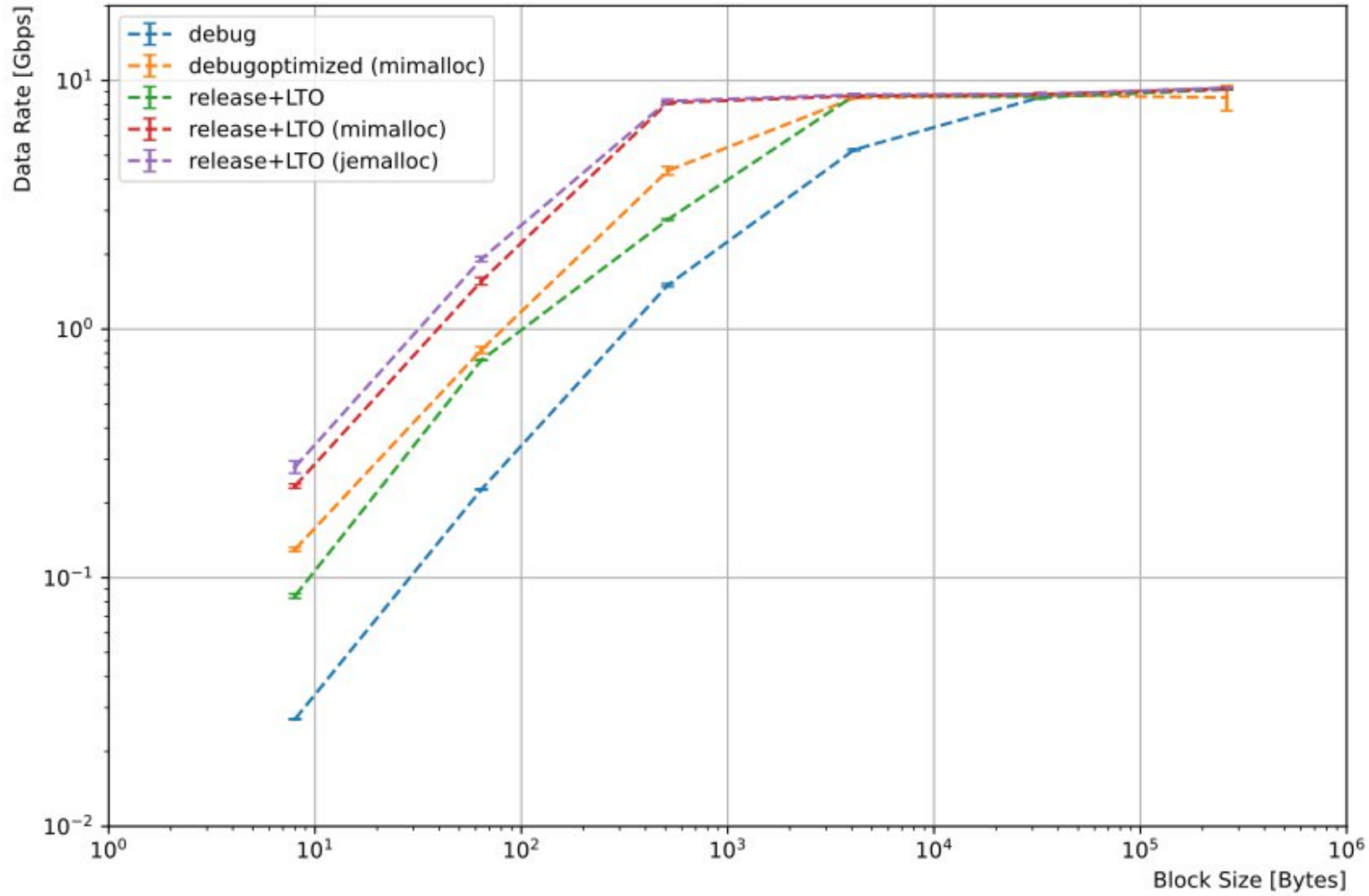
Backup

Mattermost Notifications



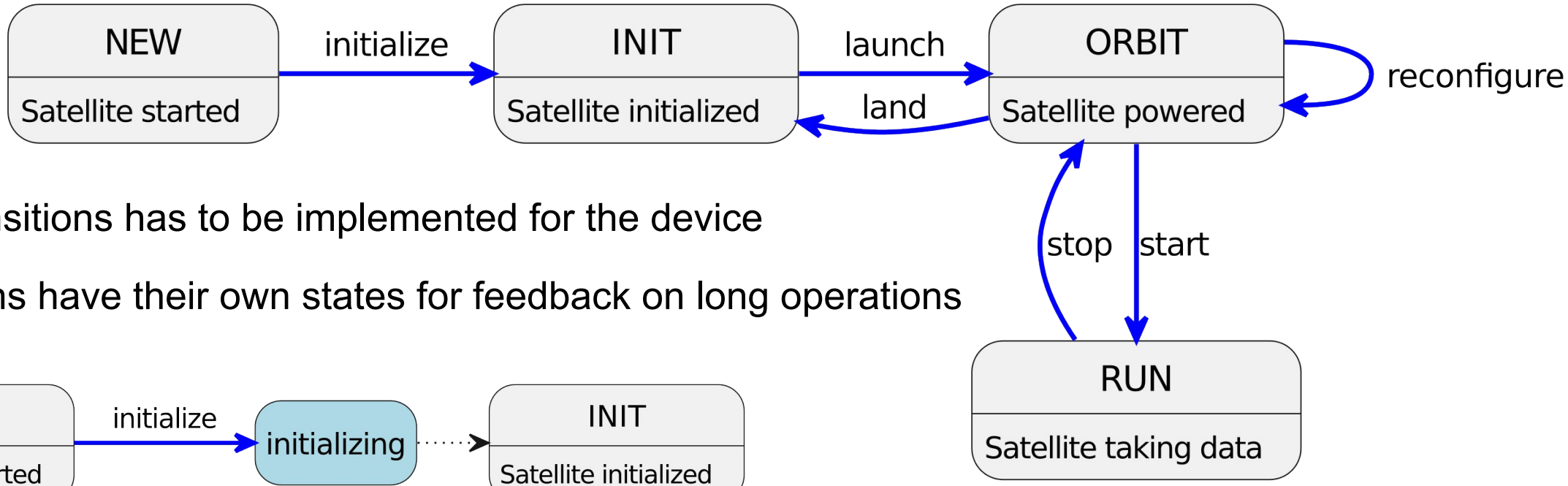
Grafana Online Monitoring

Data Transmission Speed



Satellite Finite State Machine

- Device is always in well-defined state

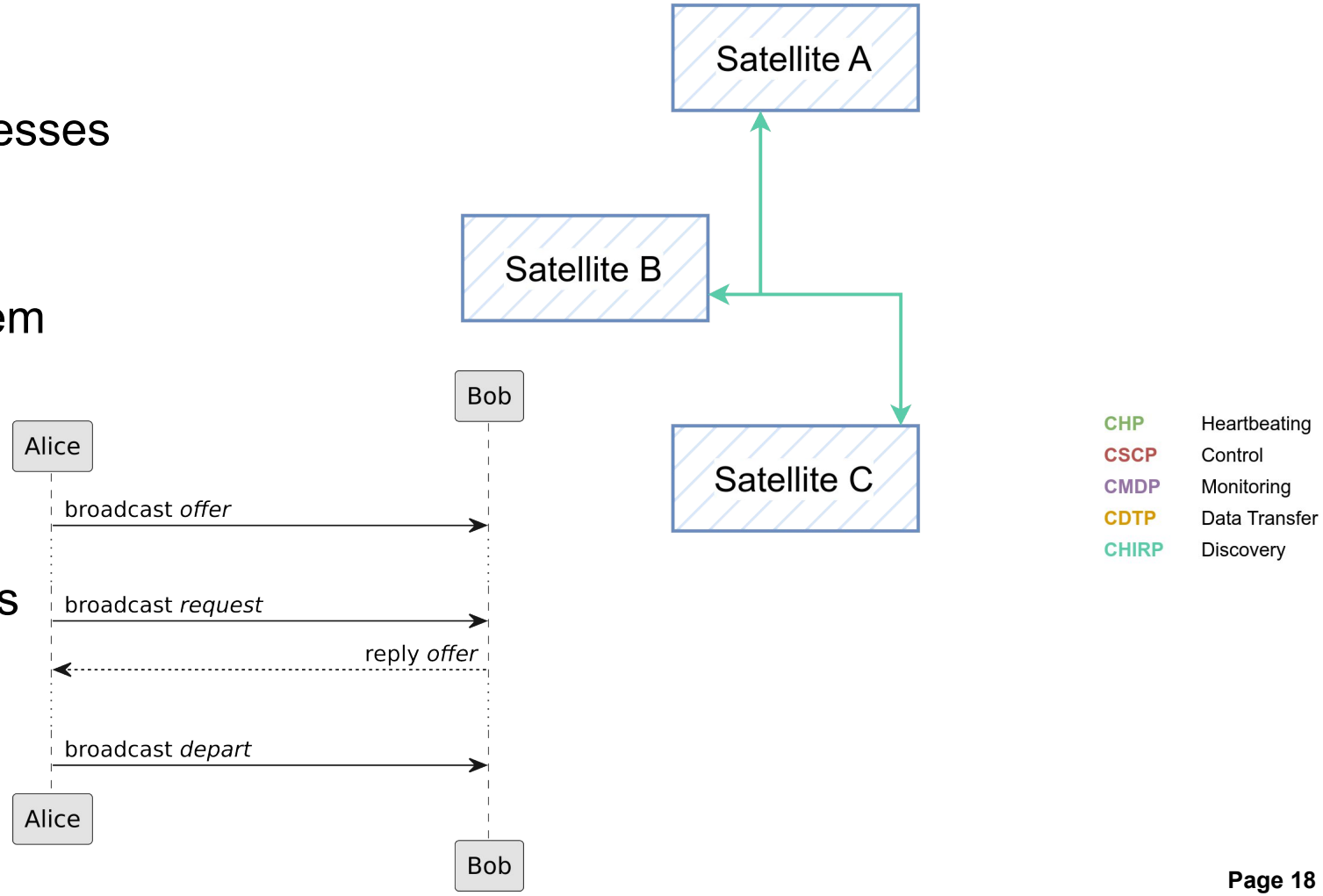


- Each transitions has to be implemented for the device
- Transitions have their own states for feedback on long operations

- Additional **SAFE** and **ERROR** states for error handling

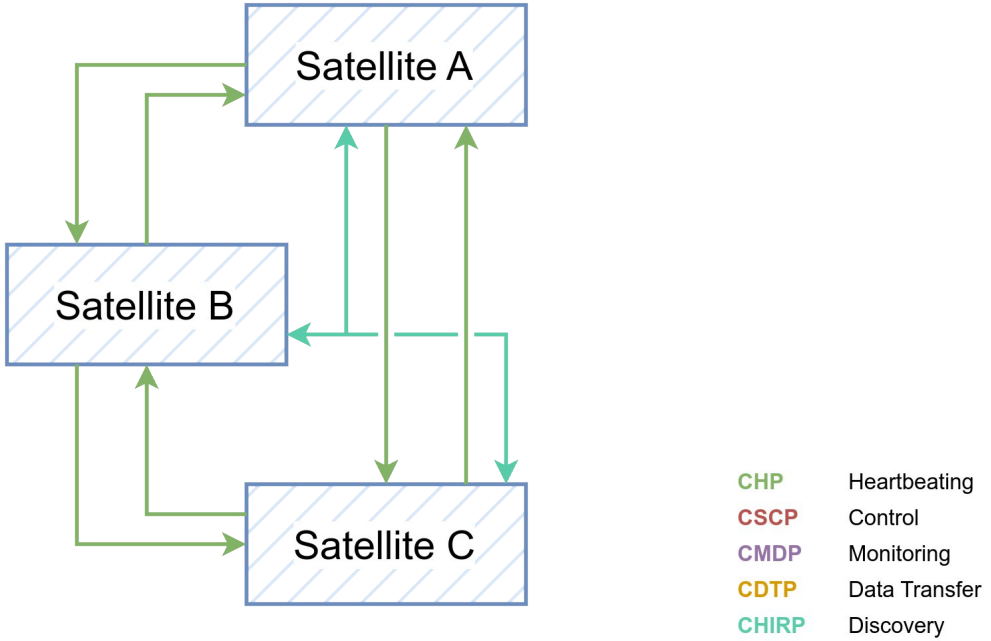
Network Discovery

- Satellites in the local network are discovered automatically
- Advantages:
  - No need to assign fixed IP addresses
  - Simpler setup procedure
  - Allows for autonomy in the system
- Uses UDP broadcasts:
  - OFFER for provided services
  - REQUEST to search for services
- Ports assigned dynamically



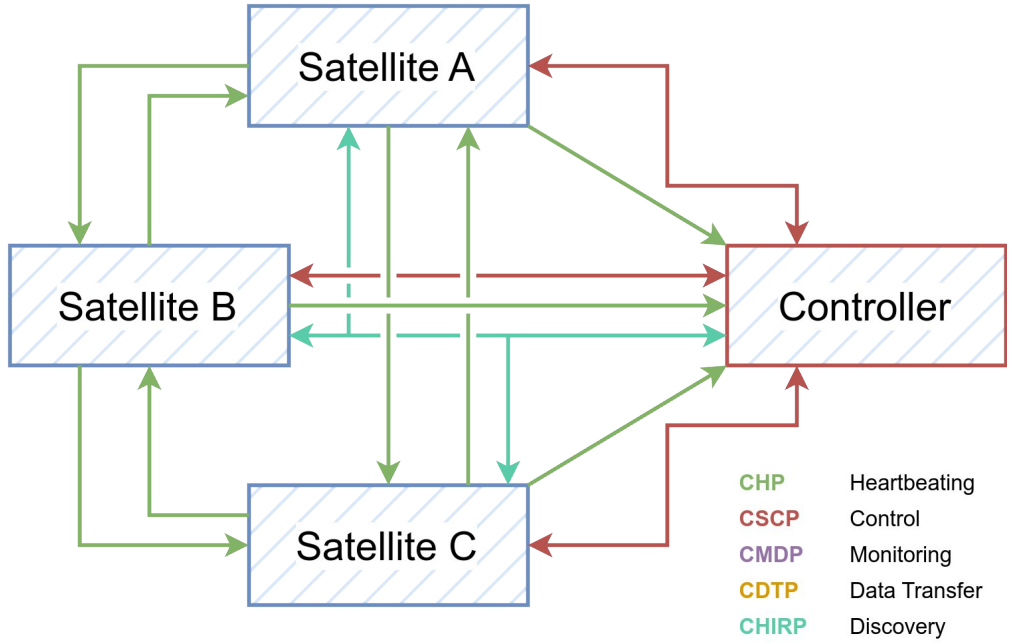
Autonomy

- Satellites emit regular “**heartbeats**”
- Heartbeats contain the current state of the satellite
- Each satellite connects to the heartbeats of the other satellites
- When another satellite sends a heartbeat with the **ERROR** state, satellites go to **SAFE** state
- When a satellite stops sending heartbeats (e.g. due to a crash), satellites go to **SAFE** state
- Heartbeats can also be used to monitor the states of the satellites with a stateless software



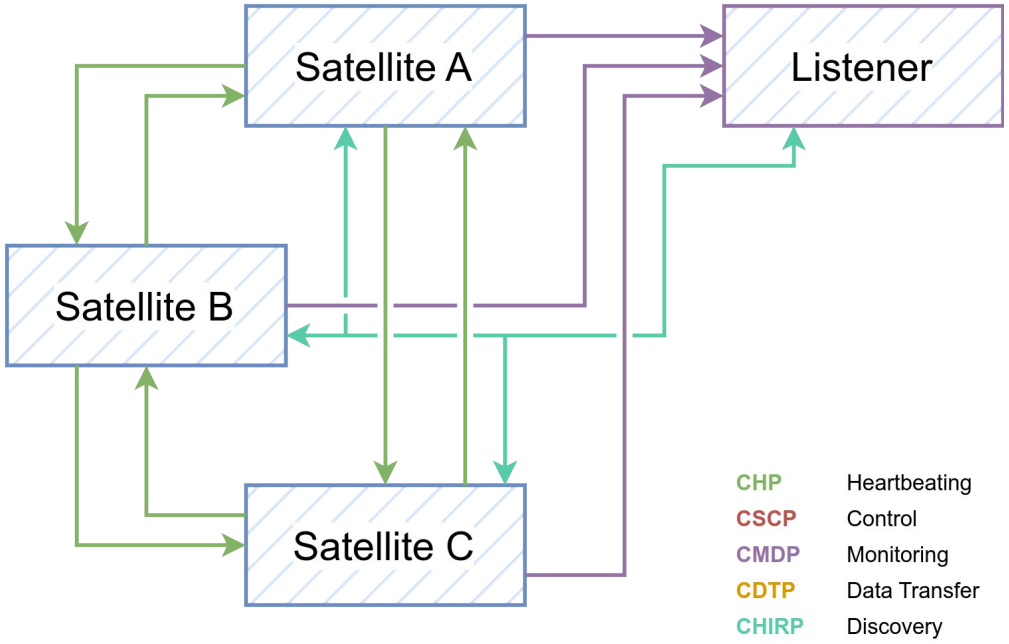
Control

- A controller joins the network:
  - Uses network discovery to find satellites
  - Connects to receive heartbeats
  - Uses heartbeats to monitor the state
  - Sends commands to a single or all satellites
- Client-server pattern for commands
- Controllers are stateless
- Custom commands are possible
  e.g. read register values



Monitoring

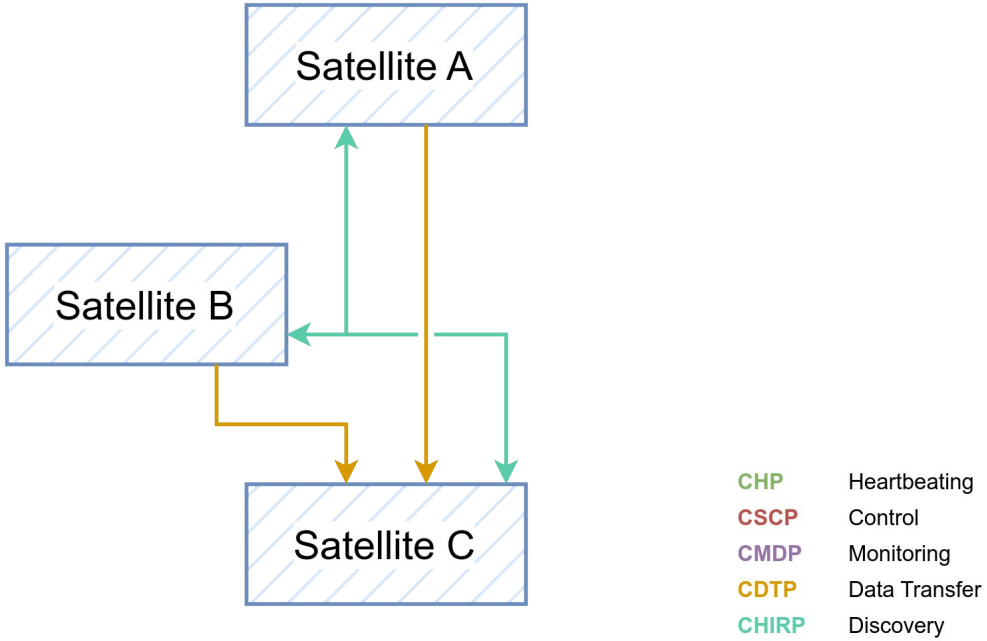
- Satellites can publish telemetry and log messages
- A monitoring listener joins the network:
  - Uses network discovery to find satellites
  - Subscribes to relevant log messages or telemetry
- Publish-subscribe pattern
- Only subscribed monitoring data is actually transmitted
- Simple integration into third-party software
  like InfluxDB/Grafana or Mattermost



Data Transmission

- Sending data over the network can be useful:
  - Embedded system (e.g. Caribou)
  - Store all data in one place
  - Flexible Support for file format
    - EUDAQ2
    - HDF5

Caribou DAQ system



- Capable of >10 Gbps transmission throughput