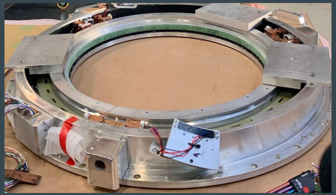
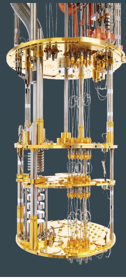
Data Acquisition and Control with the Observatory Control System Software

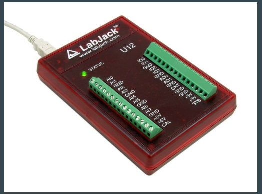
•••

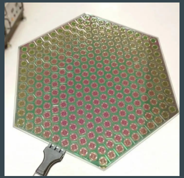
Lauren Saunders
CMB-S4 Meeting - August 2022

Hardware needed to do CMB science





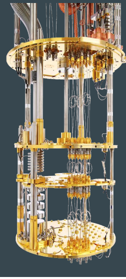


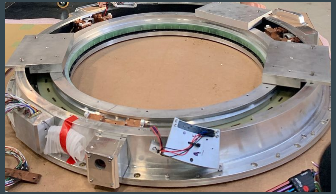


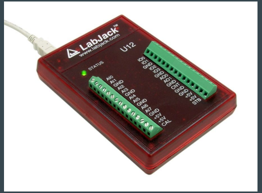
x a lot

...and more!!

Hardware needed to do CMB science







x a lot

...and more!!

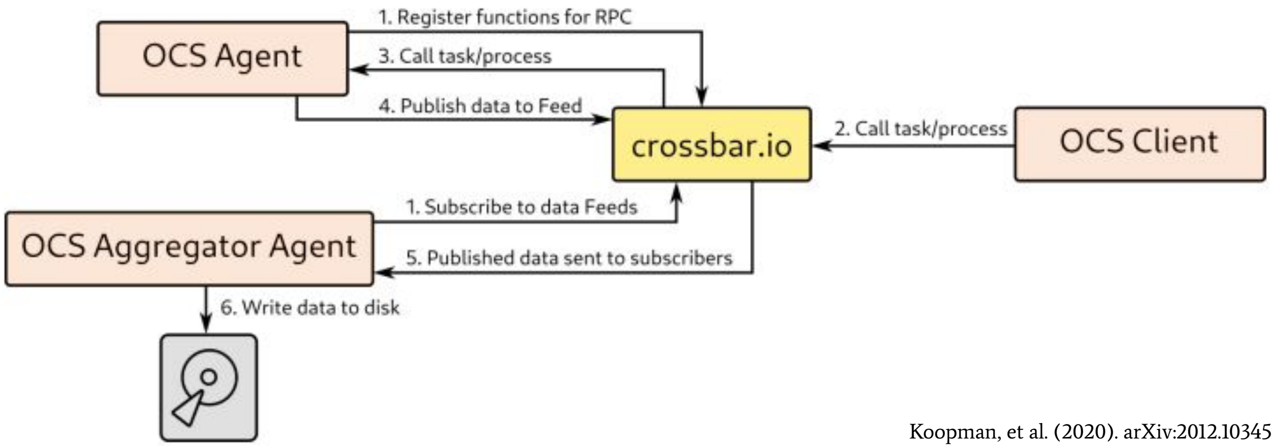
This is a lot of data to collect, and a lot of instruments to control!

What do we need from our software?

- Handles many different objects
- Takes data asynchronously
- Easy to install and use
- Easy for hardware experts to participate in development

We built the Observatory Control System (OCS) to meet these needs!

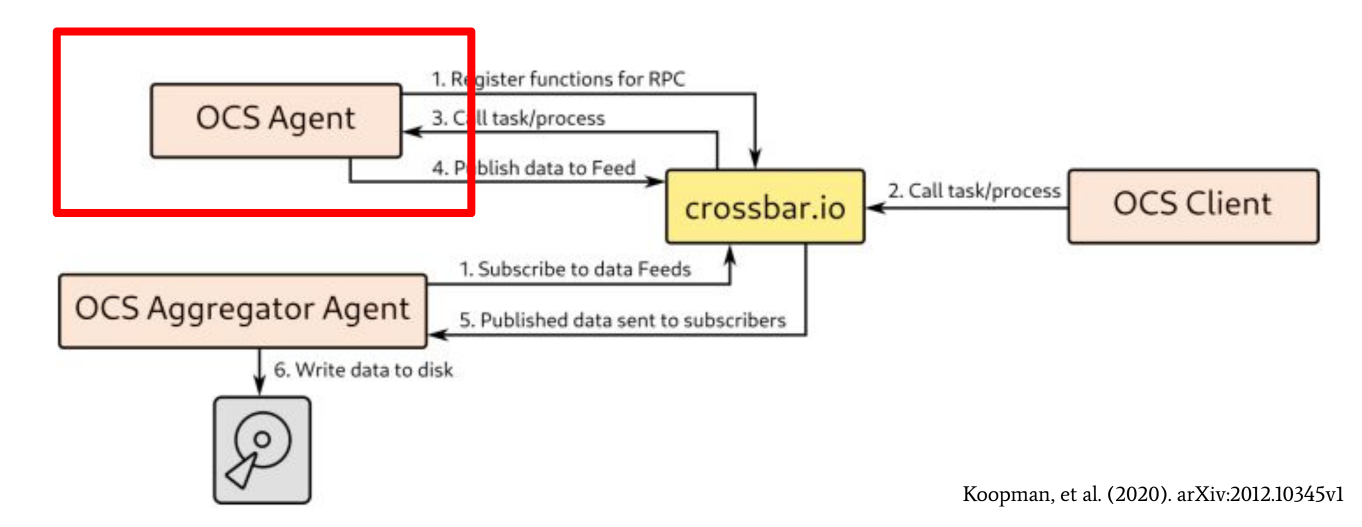
OCS Structure



Koopman, et al. (2020). arXiv:2012.10345v1

OCS Structure

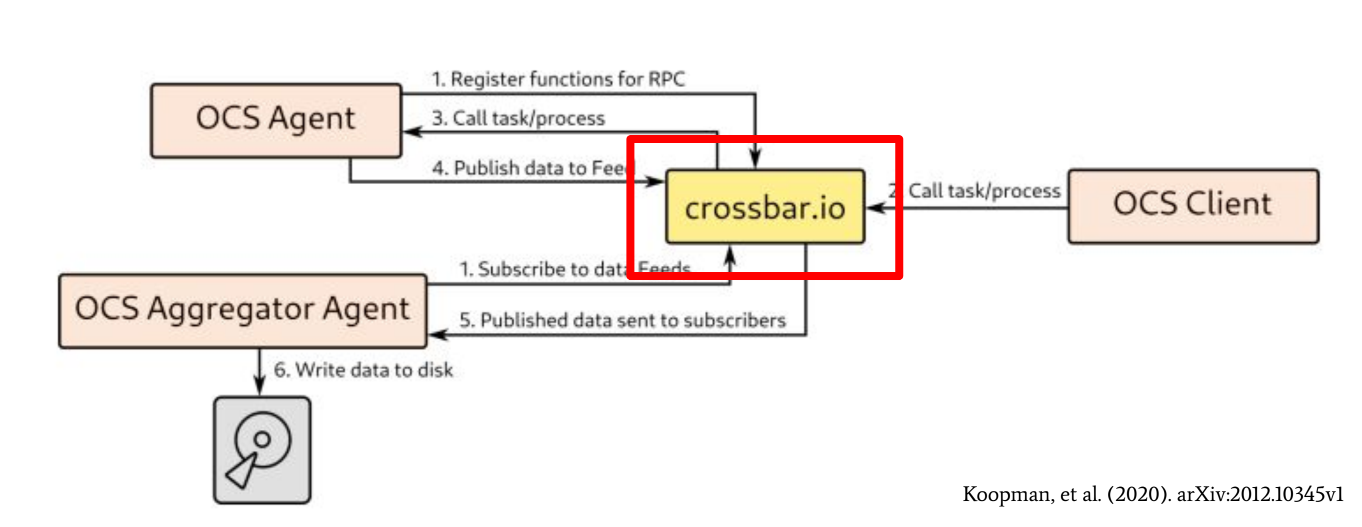
Agent = interface with hardware



Koopman, et al. (2020). arXiv:2012.10345v1

OCS Structure

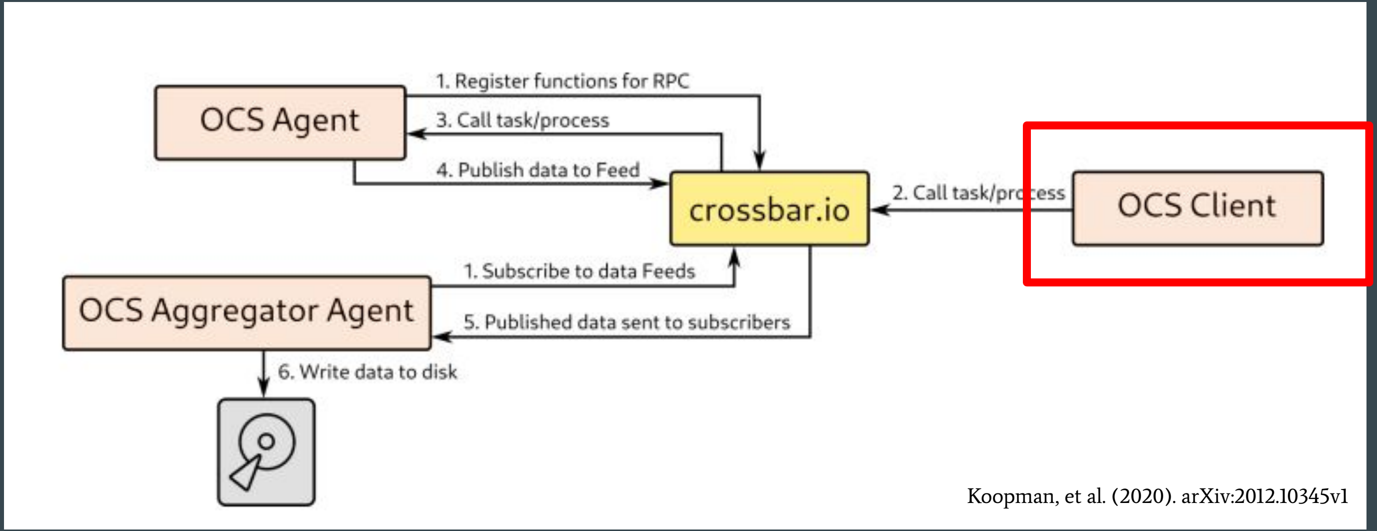
Crossbar = router



Koopman, et al. (2020). arXiv:2012.10345v1

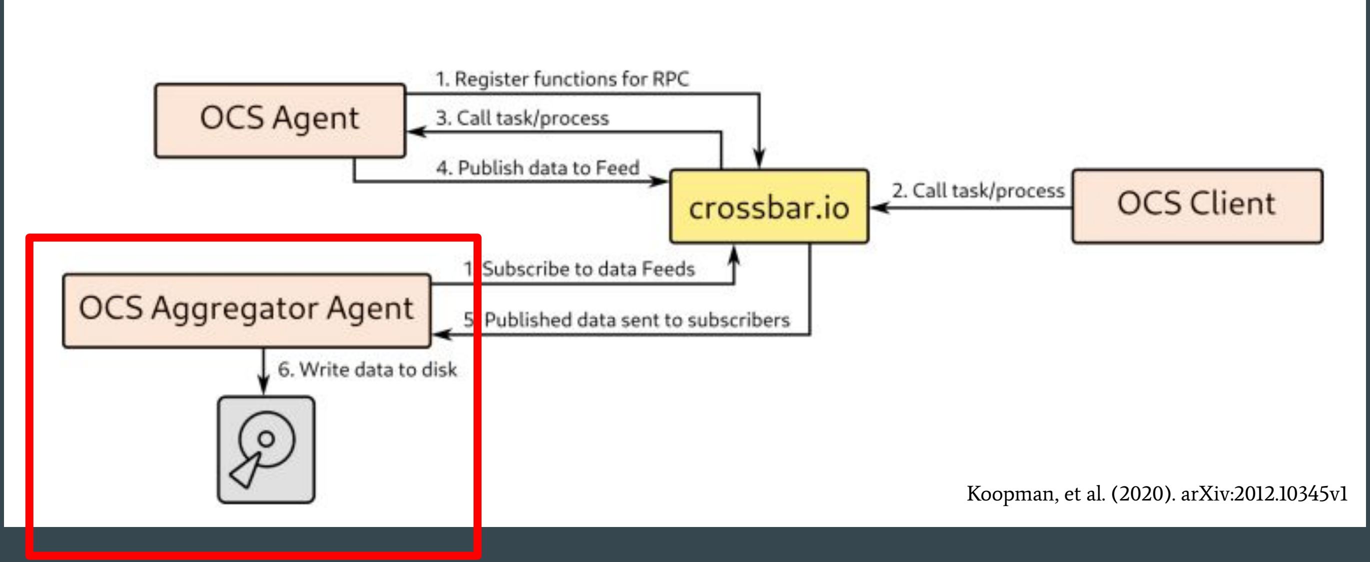
OCS Structure

Client = script that talks to the Agent



Koopman, et al. (2020). arXiv:2012.10345v1

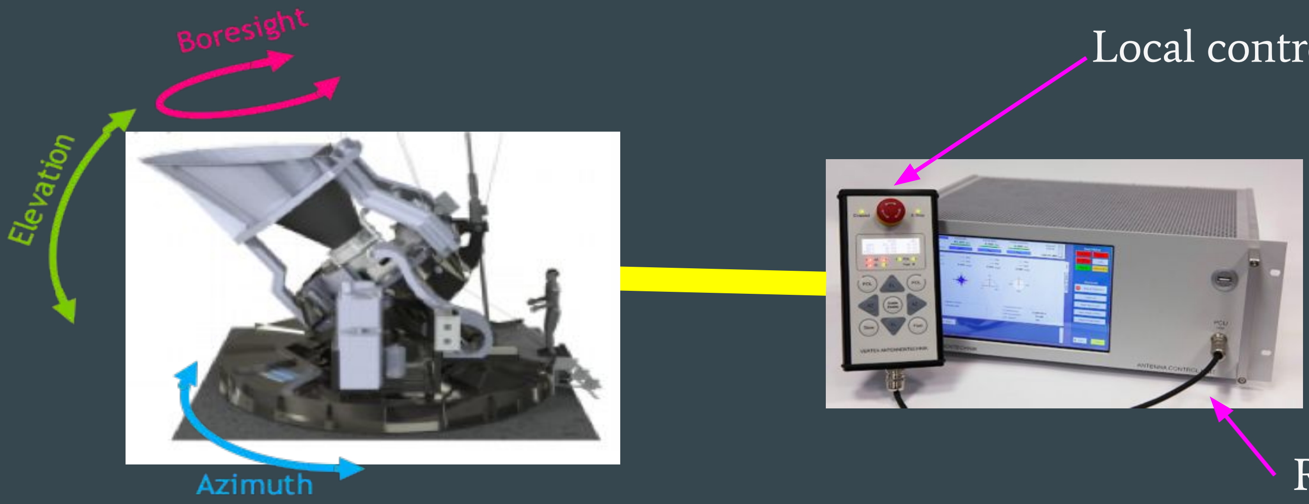
OCS Structure



Koopman, et al. (2020). arXiv:2012.10345v1

Write to disk!

An OCS Example: Simons Observatory Antenna Control Unit (ACU)

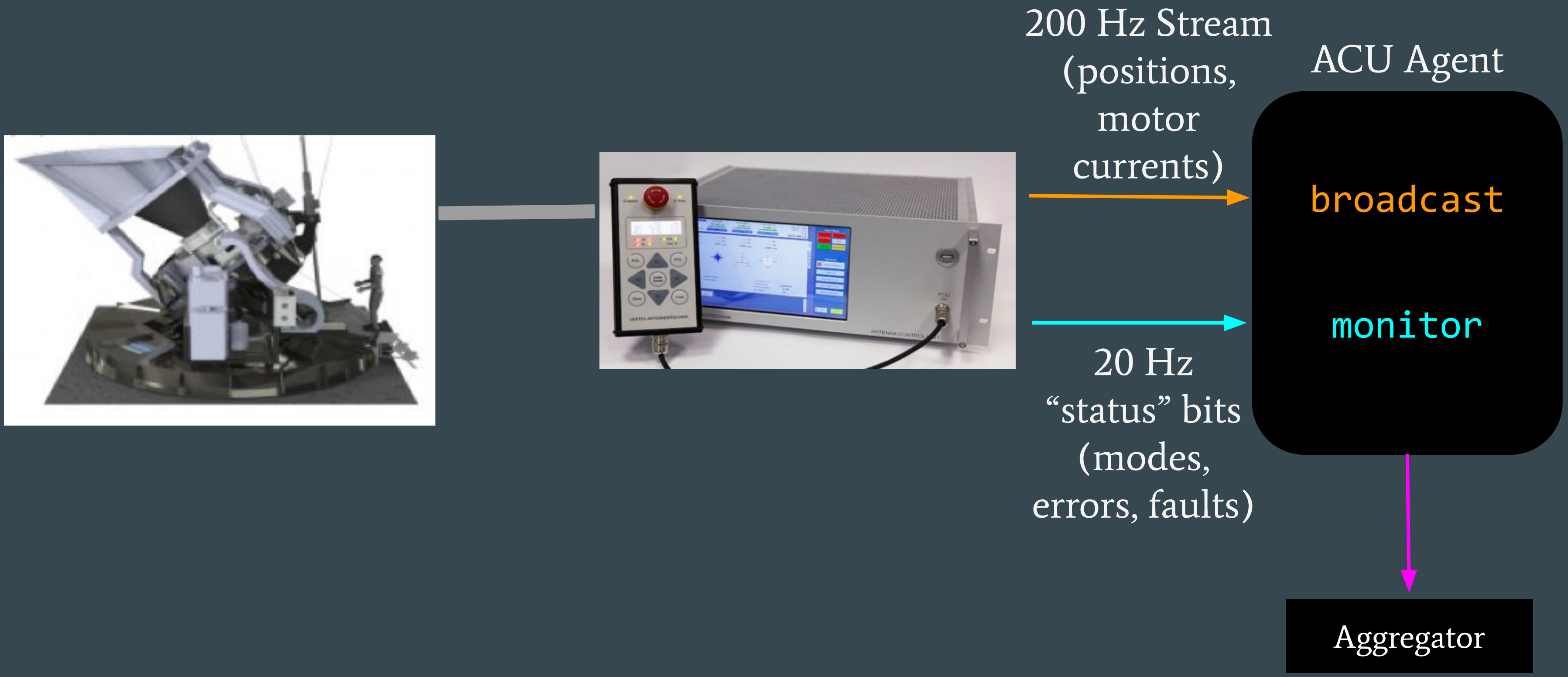


Local control

Remote control

- Hardware and basic control software built by Vertex Antennentechnik
- Software development with ACU in Emulator Mode

ACU Data Acquisition with OCS



200 Hz Stream (positions, motor currents)

ACU Agent

broadcast

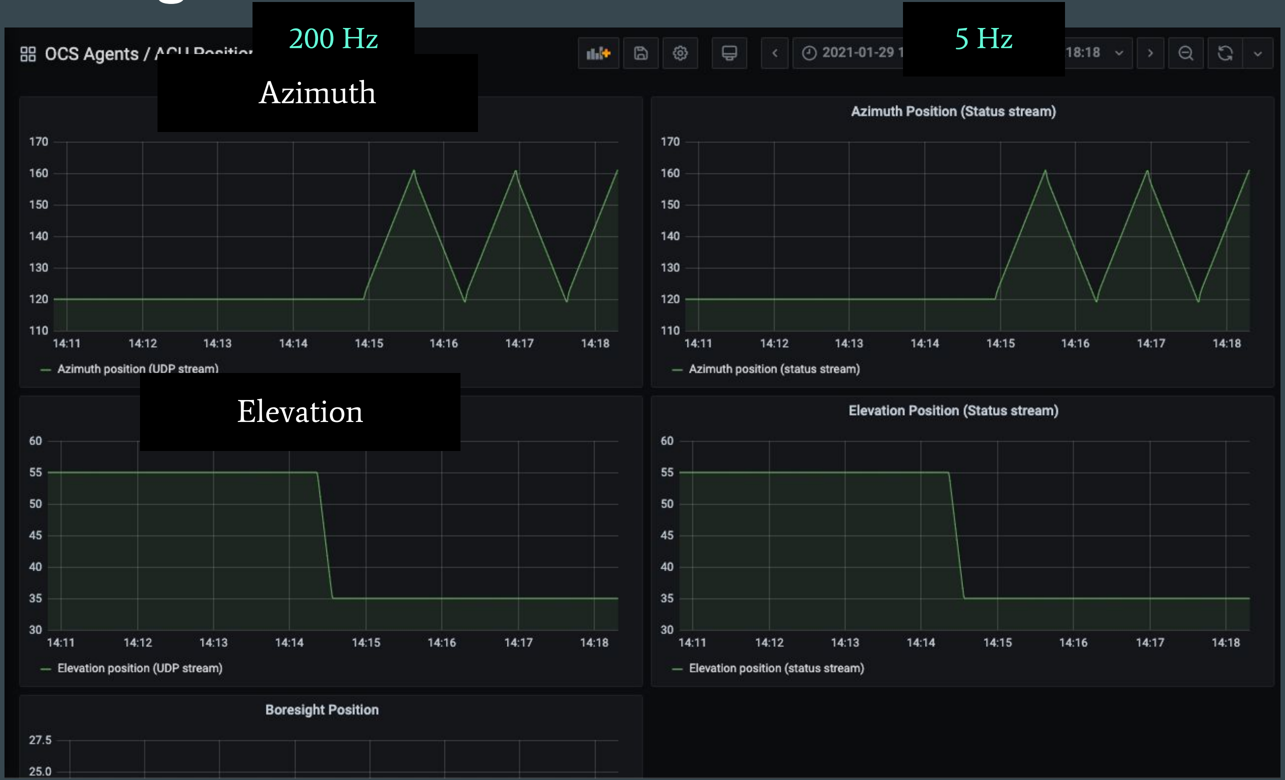
monitor

20 Hz "status" bits (modes, errors, faults)

Aggregator

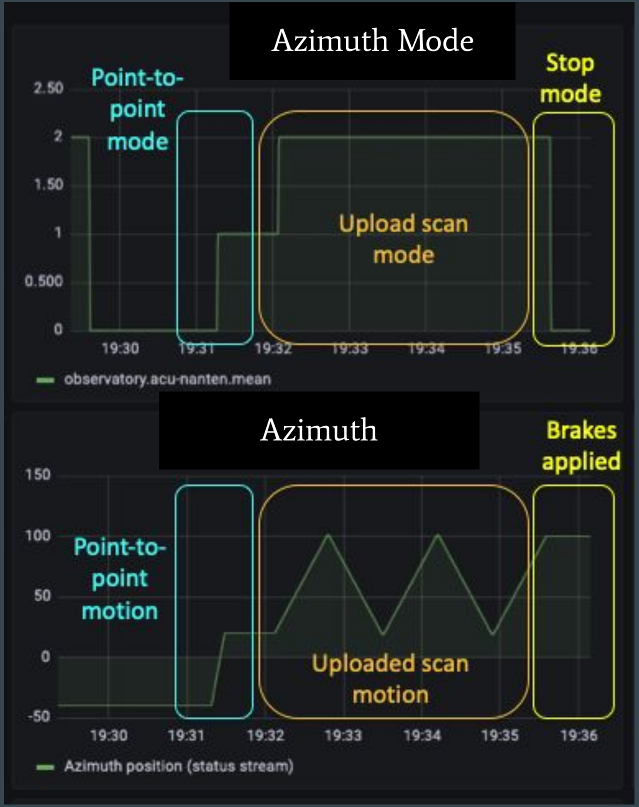
Live Monitoring with Grafana

Data from ACU in Emulator Mode

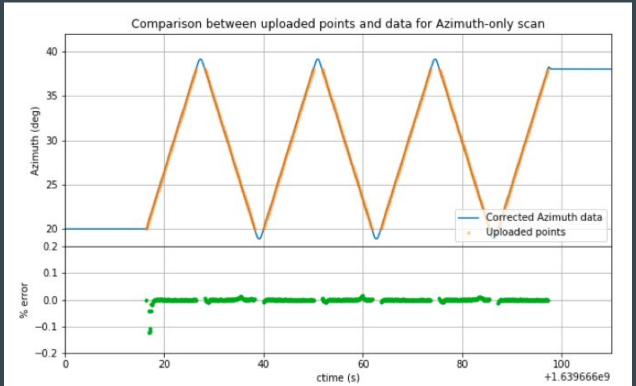

OCS Agents / ACU Positior
2021-01-29 1
18:18
200 Hz
Azimuth
5 Hz
Azimuth Position (Status stream)
170
160
150
140
130
120
110
14:11
14:12
14:13
14:14
14:15
14:16
14:17
14:18
Azimuth position (UDP stream)
Azimuth position (status stream)
Elevation
Elevation Position (Status stream)
60
55
50
45
40
35
30
Elevation position (UDP stream)
Elevation position (status stream)
Boresight Position
27.5
25.0

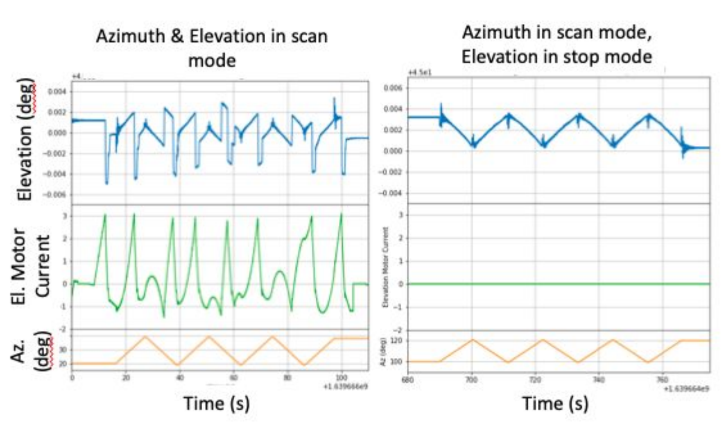
ACU Commanding with OCS

Data from ACU in Emulator Mode



OCS-ACU in Action: SO SAT Acceptance Testing

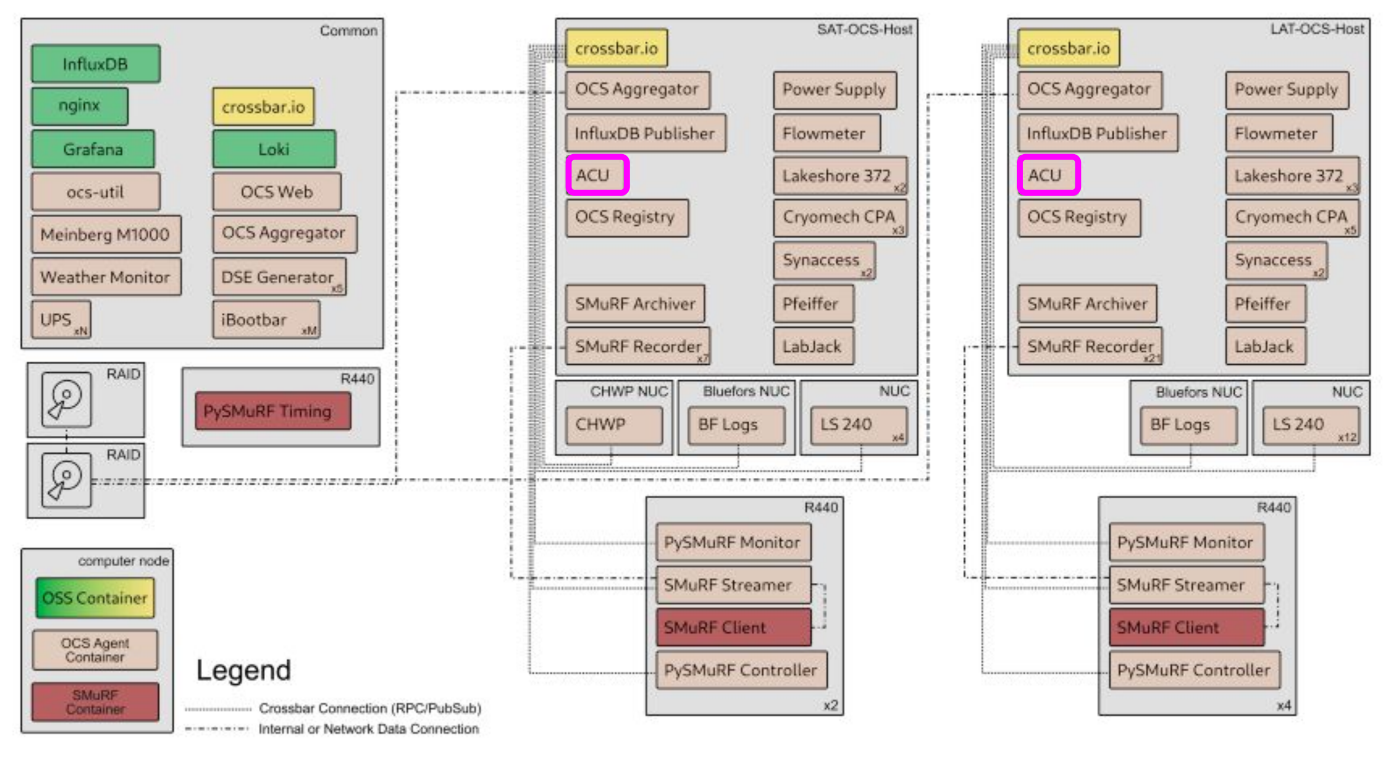




- We see a high level of agreement between points we command and encoder values (>99.98%).
- We can analyze small effects and develop software best practices

Data from the SAT1 platform at the Vertex factory in December 2021.

ACU Interactions with Other Agents



ACU needs to work together with the other components

- ➔ Sequencer, Scheduler
- ➔ (Need instruments for development)

Summary

- OCS handles data acquisition, live monitoring, and commanding for all telescope subsystems
    - It accommodates all of these different instruments and is designed for ease of use by non-experts
- SO ACU Agent: an example of an OCS Agent
    - Agent is functional and performs well with ACU in Emulator Mode and with real platforms
- End-to-end software testing allows us to test sequencer and scheduler → need simulators of instruments (including ACU) for this testing and development

Extra Slides

What is Crossbar?

- `Crossbar.io` is an open-source implementation of WAMP, which provides both RPC and PubSub
- `Crossbar` supports several programming languages, including Python, JavaScript, and C++
- It also supports two options for asynchronous networking (`asyncio` and `twisted`)

Data format on disk

- We use an adapted version of the `spt3g_software` framework, which we call `so3g`
- Files consist of Frames, which are processed through Pipelines

Docker

- Docker provides isolated environments that can be reproduced and shared
- We use Docker to package together Agents, libraries, and configuration files
- Agents can then be shared between computers and institutions through Docker images
- Some other software that we use (such as the ACU simulator) are also packaged with Docker

Existing DAQ, Monitoring, and Control Software

- Past CMB experiments use their own software systems or adapt the software systems of other similar experiments.
  - Example: The General Control Protocol (GCP) was originally developed for the Sunyaev-Zeldovich Array, and has been adapted by SPT, PolarBear, BICEP2, and the Keck Array.
- These software systems are typically insufficient for the scale of SO

Observatory Control System (OCS) Overview

- OCS is a distributed control system, which is designed to bring together the control, data acquisition, and monitoring for the variety of instruments needed to make an observation with an observatory.
- It is designed with the intention that users can easily learn to use OCS, and non-OCS experts can add or modify components to suit their needs.

An OCS Example: Antenna Control Unit (ACU)

Instrument overview

The SO telescope platforms are fabricated by Vertex Antennentechnik ("Vertex"). To facilitate telescope control and data streaming, Vertex also provides an Antenna Control Unit (ACU). This is an industrial PC with specialized device cards, and can be used locally (with button pushes) or remotely (through a connected computer).

Software interface development was completed with the ACU in its built-in "Emulator Mode", which allows the instrument to calculate what the encoder and housekeeping values should ideally be given motion parameters.

ACU Data Acquisition with OCS

The ACU sends telescope data over two streams:

Fast (200 Hz) Data over UDP:

- Timestamp
- Positions (Azimuth, Elevation, 3rd Axis)
- Motor currents (2x Azimuth, 1x Elevation, 2x 3rd Axis)

Slow (5 Hz) Status Data via HTTP:

- Timestamp
- Positions, velocities
- Housekeeping
- Limits, faults, errors, failures

ACU Data Acquisition with OCS

The ACU Agent has two Processes to manage data streams:

- `monitor`: connects to the Status stream, collects data from the stream, organizes it into different categories, and publishes each category to an OCS feed
- `broadcast`: connects to the UDP stream, collects data from the stream, and publishes each category to an OCS feed

The OCS feeds are routed through crossbar

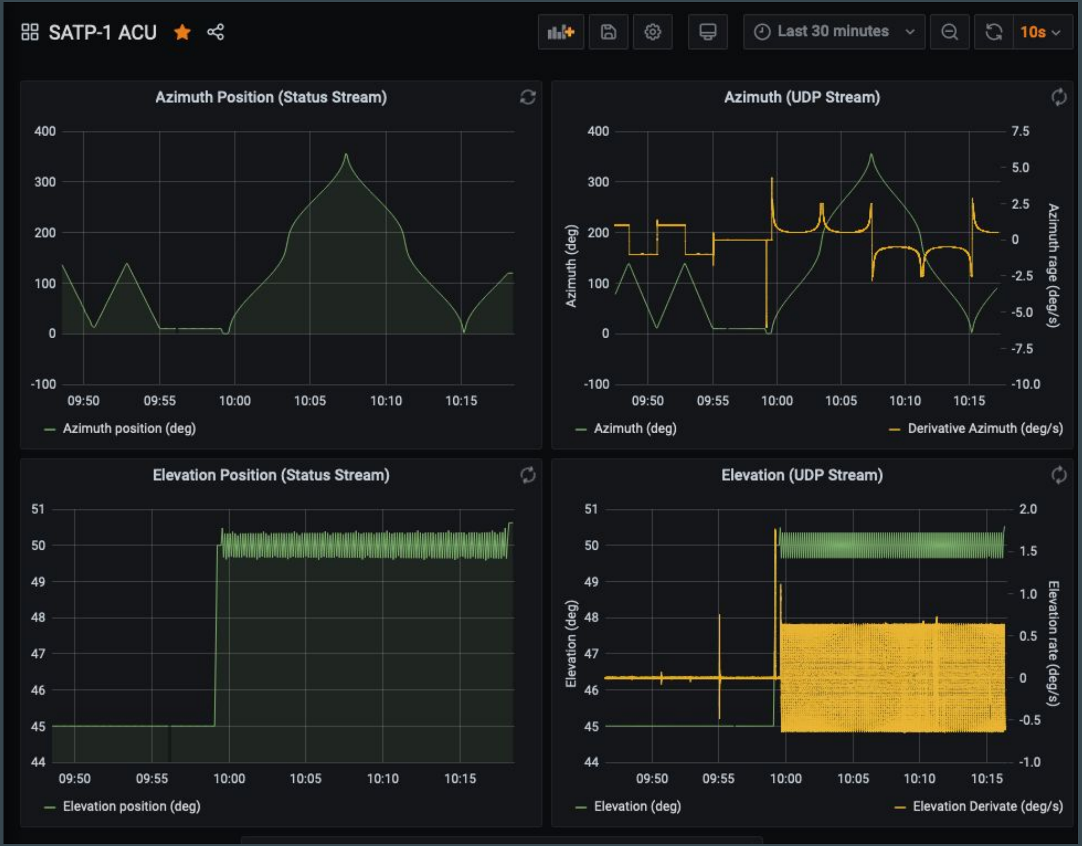
Data is written to disk by the Aggregator Agent, and interpreted by the Influx Agent for live monitoring with Grafana

ACU Commanding with OCS

The ACU Agent has 5 Tasks and Processes to manage commanding:

- `go_to`: commands the telescope to move to a specified (azimuth, elevation) position at maximum speed
- `set_boresight`: commands the telescope to move to a specified 3rd axis position at maximum speed
- `fromfile_scan`: uploads a set of points from a file and executes a scan
- `generate_scan`: calculates and uploads a set of points based on azimuth-only constant velocity scan parameters, and executes the scan. This kind of scan can be finitely long, or can run forever.
- `stop_and_clear`: puts the telescope in Stop mode and clears out any uploaded points

OCS-ACU in Action: SO SAT Acceptance Testing



Scan performed by an SO Small Aperture Telescope platform at the Vertex factory in Duisburg, Germany, in November 2020.

ACU Interactions with Other Agents

To complete a successful observation, we need the ACU functions to execute in sequence with functions from other instruments (detectors, half-wave plate, etc.). To do this, we have a Sequencer, which times the execution of Clients associated with certain Agent operations.

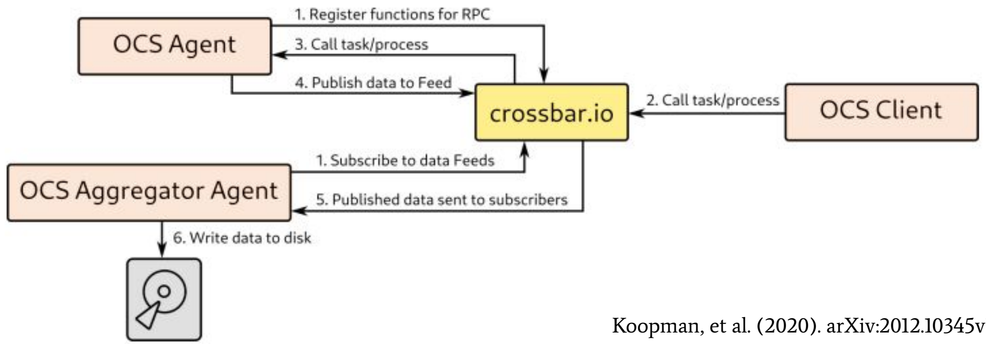
Instruments are not centralized in one location, so instead we build software-only simulators for each hardware component.

OCS Structure

Agents: Software servers that interface with hardware components or other software components. Agents contain the function ("Tasks" and "Processes") that are needed to perform an operation or data acquisition.

Clients: Scripts that choreograph the functions within one or more Agents.

Crossbar: WAMP router that facilitates RPC and PubSub communications



Koopman, et al. (2020). arXiv:2012.10345v1

ACU Simulator

The ACU simulator has 2 components:

- An HTTP server that manages all of the data, including the Status fields and the UDP fields. This server can also receive commands, and is designed to interact with the ACU Agent commands.
- A UDP server that receives the UDP fields from the HTTP server and publishes them in packets of 10 data points to a port. The data is encoded in the same way as the UDP data from the ACU.

This simulator is run in a Docker container when used for multi-simulator testing.