



SKA use case: Aladin data vis in Jupyter

WP4 Tech Forum 14-04-2021

SQUARE KILOMETRE ARRAY

Exploring the Universe with the world's largest radio telescope

James Collinson
Operations Data Scientist

Overview

- Recap of previous work
- Objective: Demonstrate **visualisation** of IVOA-compatible **SDC*1 data in solution notebook**
 - Hierarchical Progressive Surveys (HiPS) previously generated from synthetic SKA (SDC1) data
 - Extend for SRC use case by enabling in JupyterHub SAP
- Though not fully possible in notebook, **can easily provide web visualisation of data for anyone to use**

*SDC = Science Data Challenge

SKAO - An Observatory



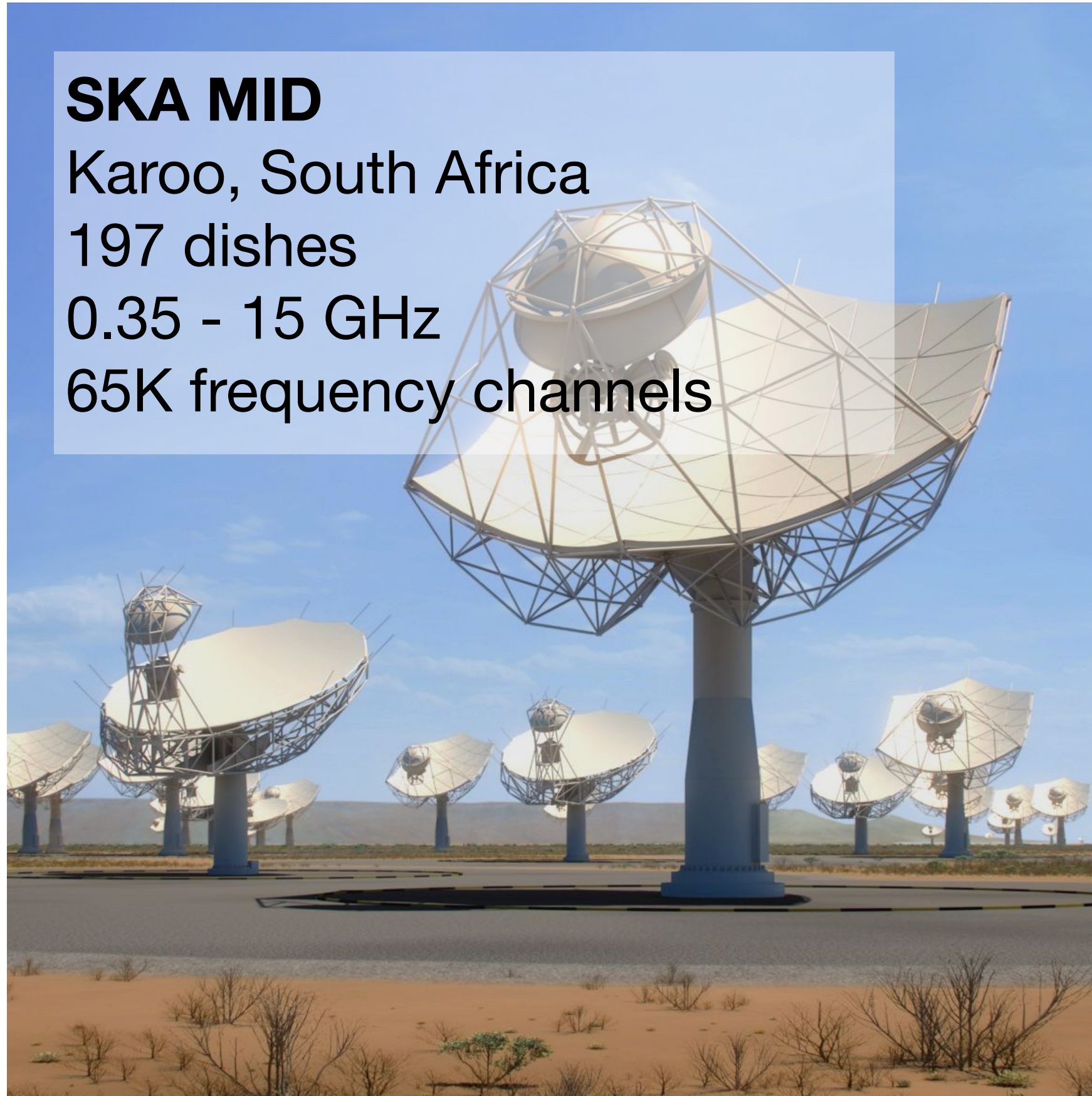
SKA MID

Karoo, South Africa

197 dishes

0.35 - 15 GHz

65K frequency channels



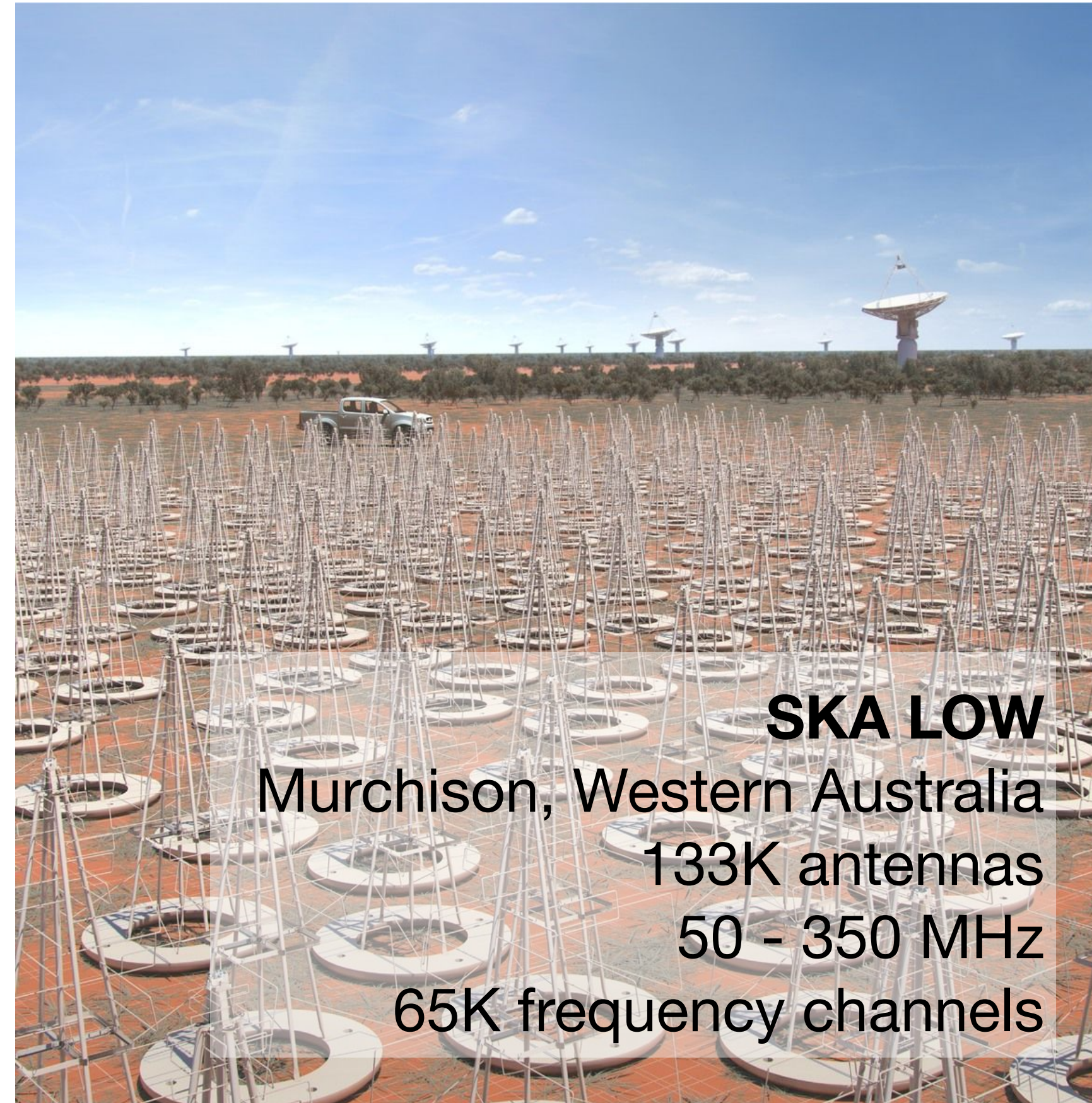
SKA LOW

Murchison, Western Australia

133K antennas

50 - 350 MHz

65K frequency channels

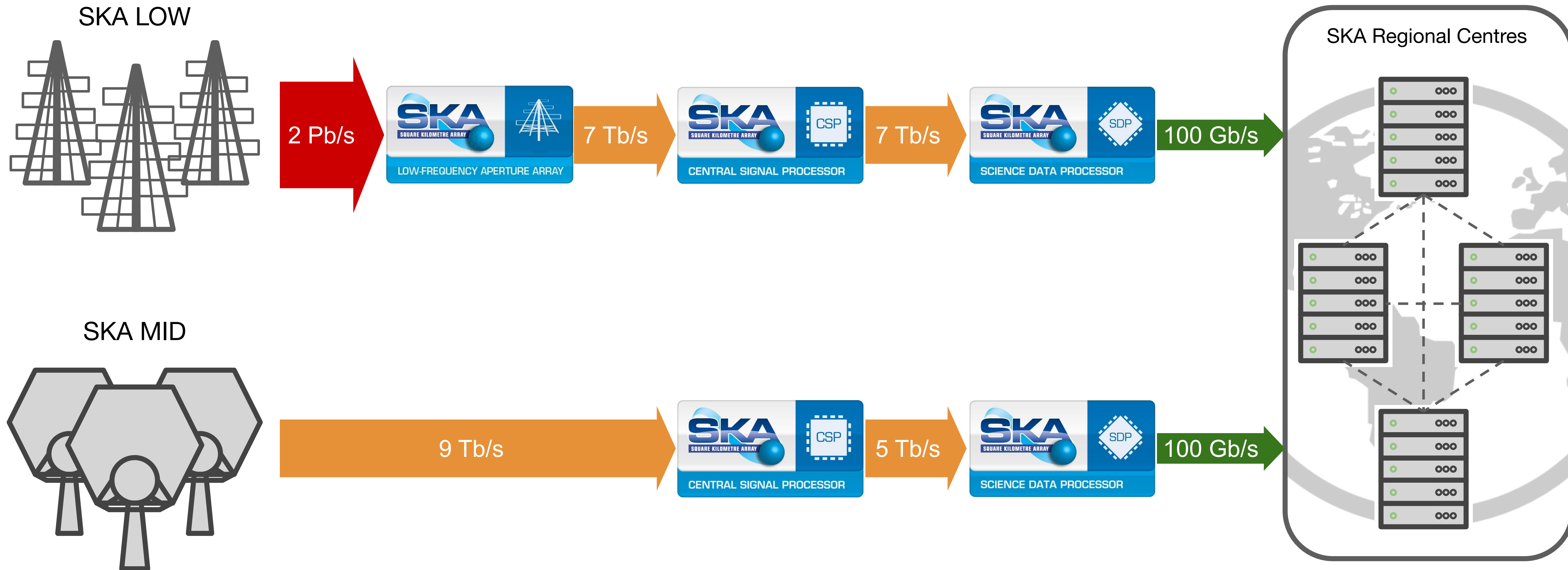


Test systems **already taking data**. Main science programmes from **~2028/9**. **50 year** lifetime.



Exploring the Universe with the world's largest radio telescope

SKA Observatory Data Flow



* Data rates approximate



SKA Regional Centres



ARCHIVE

Archival of the observatory data products. Once scientific results are published, outputs of analyses are made available.



DATA DISCOVERY

Once SDP has pushed the data to the regional centres, how will users find/peruse their data?
How will data from published results be easily found?



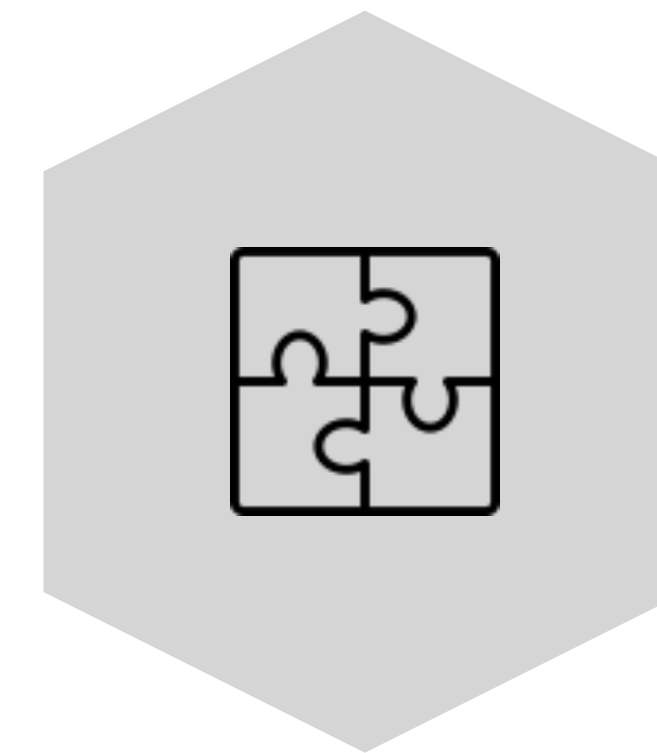
DISTRIBUTED DATA PROCESSING

Use cases are made to be reproducible. Compute comes to the data (high data volume).



USER SUPPORT

SRCs must support the key science project teams as well as general users. This will mean user ability will be varied.



INTEROPERABILITY

Multiple regional SRCs, locally resourced but interoperable. SRCs may be heterogeneous in nature but with common core functionality.

Credit: Rohini Joshi



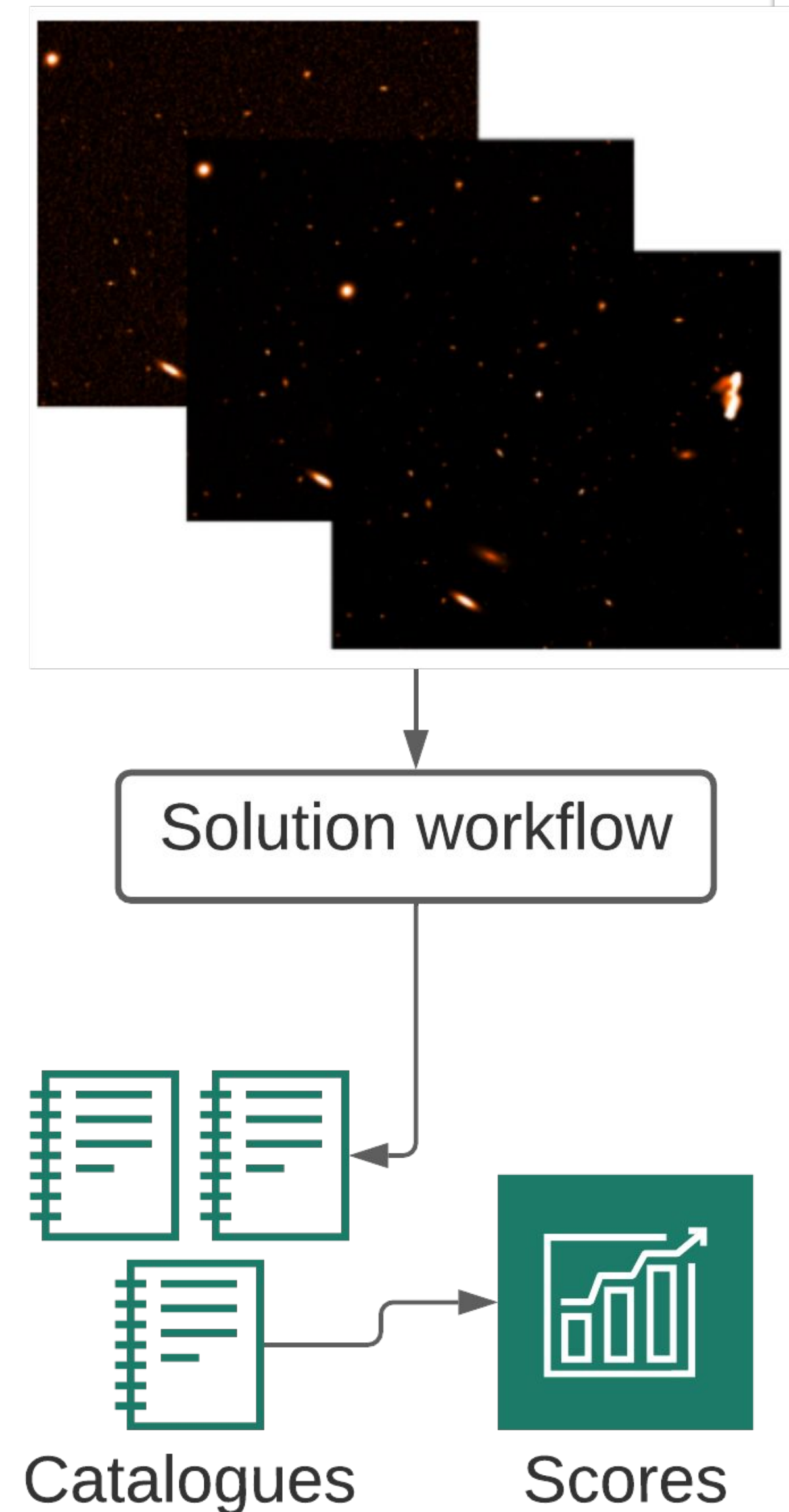
Previous Work - Recap

- SDC1 solution workflow
 - Notebook-based analysis - images slow to visualise
- Hierarchical Progressive Surveys (HiPS)
 - [IVOA-recommended](#) schema for enabling [progressive views](#) of surveys
 - Hierarchical tiling of pre-generated images in structured archive
 - Tools provided for generation from e.g. FITS
- Previously generated from synthetic SKA ([SDC1](#)) data
 - Visualised using Aladin Desktop



Recap: JupyterHub SKA analysis environment

- Science Data Challenge 1
 - Build expertise in community
 - Simulated radio images in 3 bands (560, 1400, 9200 MHz)
 - Source finding and classification
- Workflows lacked reproducibility



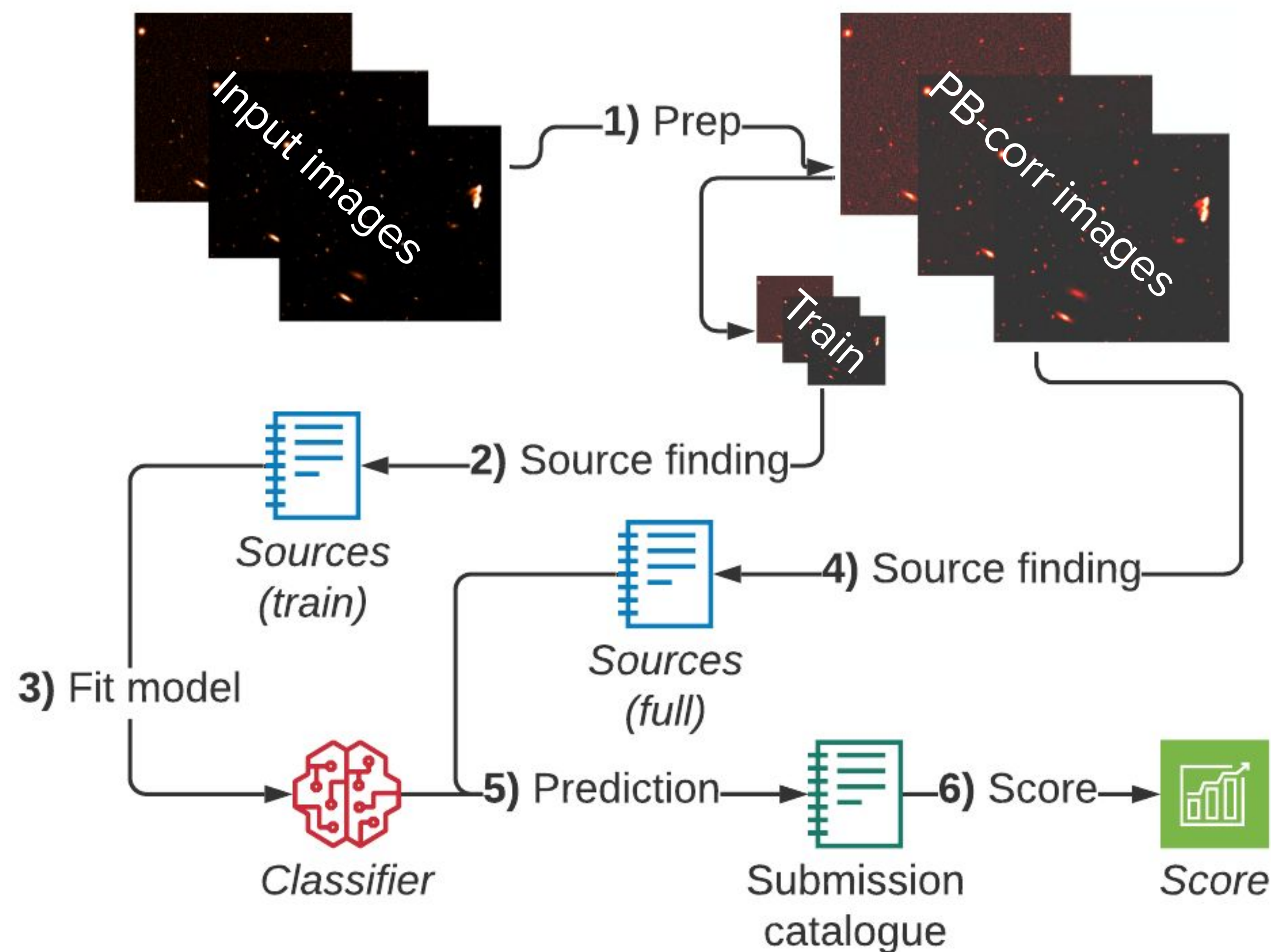
Recap: JupyterHub SKA analysis environment

- Solution:

- PyBDSF source finding
- ML classification

- Output:

- Environment with tools
- Python wrappers with API
- Solution script (adaptable)
- [GitLab](#)



Recap: JupyterHub SKA analysis environment

- SDC1 env on JupyterHub
- Built [Docker image](#) on top of jupyter/base-notebook
 - Unit tests pass
- Next steps:
 - Scale up instance provision
 - Integration testing
 - Provide example notebook

Spawner Options

<input type="radio"/>	Default The default image
<input type="radio"/>	Jupyter data science notebook data science env
<input checked="" type="radio"/>	SKA SDC1 notebook Notebook env for SDC1 analysis

Spawn

Context: Workflow from user perspective

LOGIN TO SAP



User will authenticate with external service to login to Science Analysis Platform

START SKA ANALYSIS NOTEBOOK KERNEL



Choose from a selection of notebook server environments depending on requirement

SELECT SKA DATA IN DATA LAKE



In running kernel, user can download data from data lake prototype

VISUALISE IN NOTEBOOK



Interactive visualisation of chosen data using Aladin viewer in Jupyter notebook



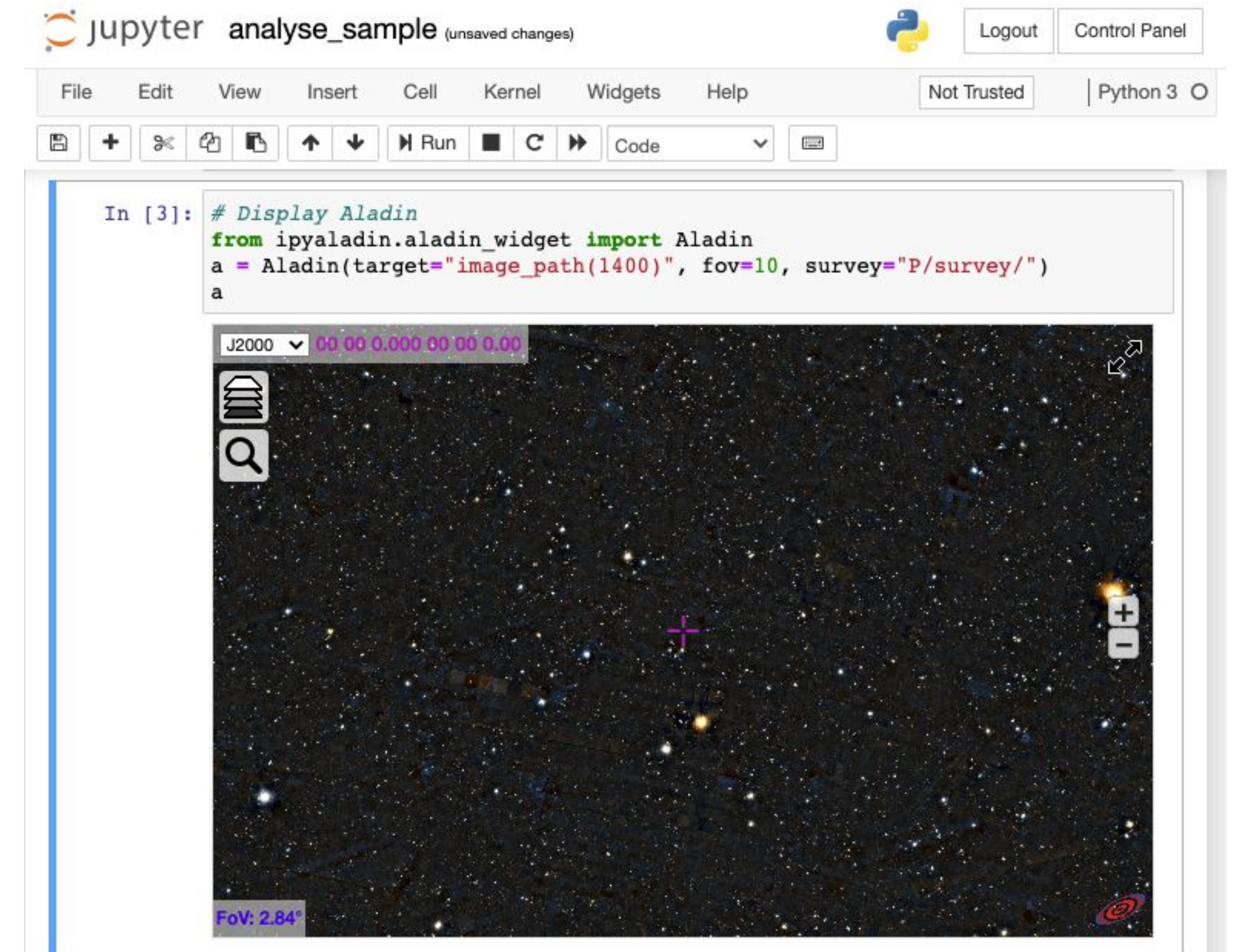
Step 1: Data lake + SAP integration

- Merge existing functionality into single environment:
 - Rucio data lake client
 - SDC1 analysis
- Build Rucio client into SDC1 image - particular about environment requirements



Step 2: ipyaladin

- Python package for Aladin vis in notebooks
- Make modifications to **enable custom image display** - not possible
 - Functionality yet to be included
 - Latest code not available



The screenshot shows a Jupyter Notebook interface with the following code in a cell:

```
In [3]: # Display Aladin
from ipyaladin.aladin_widget import Aladin
a = Aladin(target="image_path(1400)", fov=10, survey="P/survey/")
a
```

The output of the code is a star field image displayed in a widget. The widget includes a toolbar with icons for zooming and a status bar at the top showing "J2000" and coordinates "00 00 0.000 00 00 0.00". A purple crosshair is visible on the image, and a "FoV: 2.84°" label is at the bottom left.

Outcomes



- Data lake downloads enabled
- ipyaladin has some limitations currently
 - [Unable to point to custom surveys](#)
 - Source code on Github out of date (as of 16-04-21)
 - Cannot build in current form (e.g. js paths moved from ipyaladin/static to js/src/)
- HiPS served from web server
 - <https://aladin.skatelescope.org/test3> Navigate to (0, -30)



Next steps

- Upload full SDC data cubes to AWS web service
 - Publicly accessible
 - In future, could embed Aladin Lite into SDC# website
- Explore alternatives to Aladin for JupyterHub use case
 - CARTA: <https://cartavis.org/>