

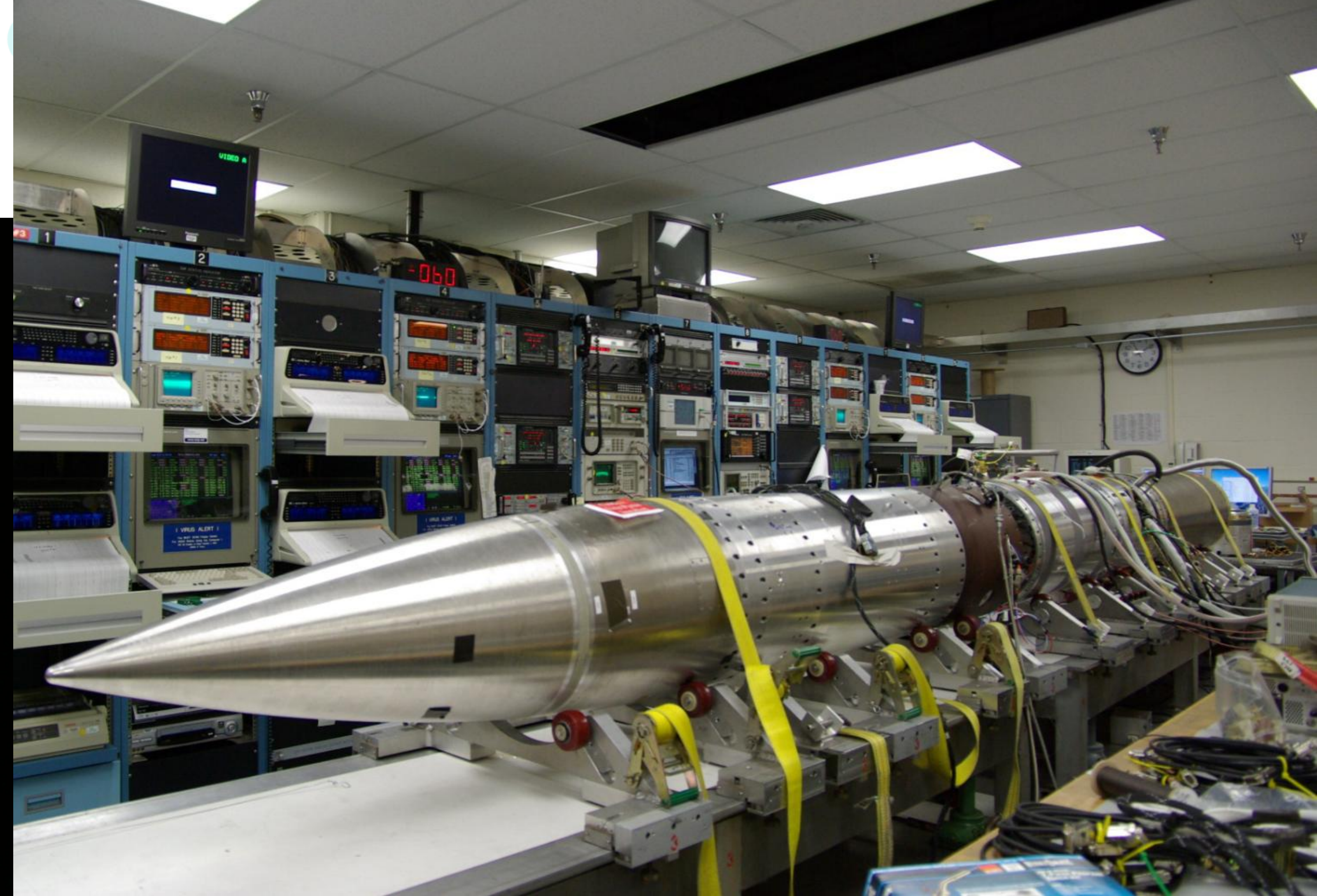


LSST Science Pipelines

Ian Sullivan | ASTR 597 | January 17, 2023



Who am I?



Ian Sullivan | ASTR 597 | January 17, 2023



Who am I?



Murchison Widefield Array (MWA)



Who am I?




Photo credit: Emily Acosta

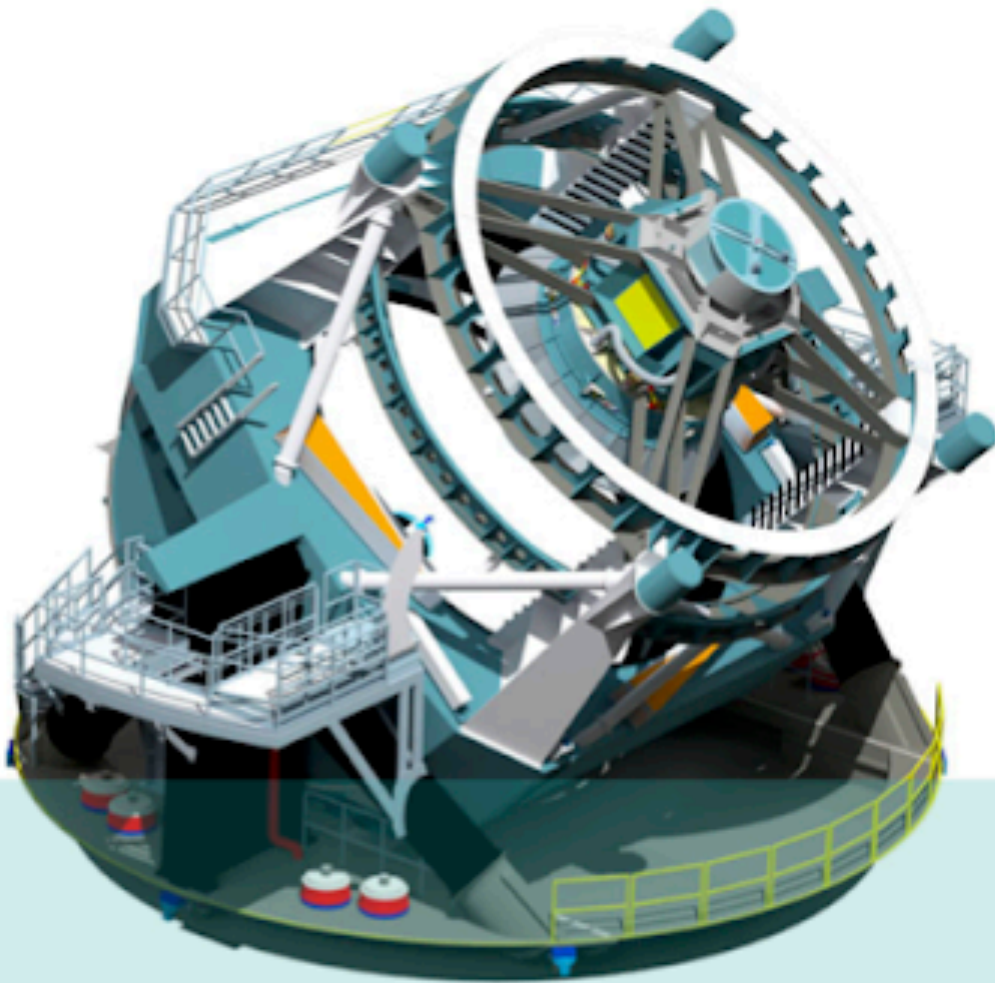


LSST Science Pipelines overview



Raw Data: 20TB/night

 Sequential 30s images covering the entire visible sky every few days



Prompt Data Products

- Alerts: up to 10 million per night
- Raw & Processed Visit Images, Difference Images, Templates
- Transient and variable sources from Difference Image Analysis
- Solar System Objects: ~ 6 million

Data Release Data Products

- Final 10yr Data Release:
- Images: 5.5 million x 3.2 Gpixels
 - Catalog: 15PB, 37 billion objects



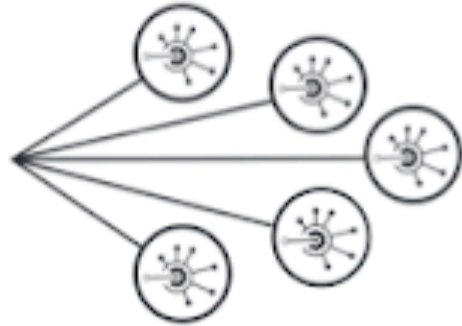
via nightly alert streams



via Prompt Products DB



via Data Releases



Community Brokers

Rubin Data Access Centres (DACs)

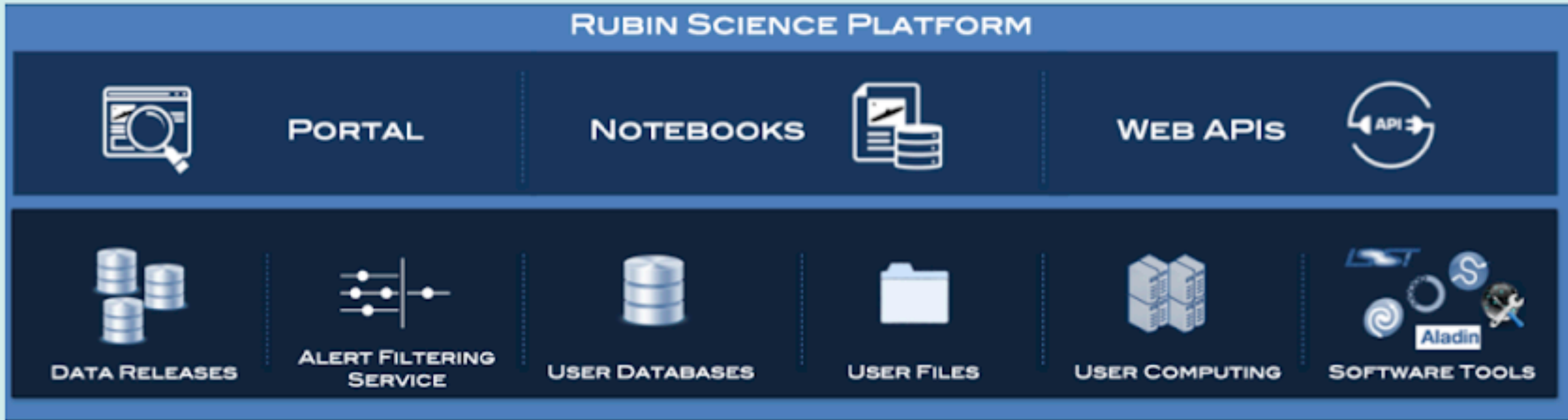
- USA (USDF)
- Chile (CLDF)
- France (FRDF)
- United Kingdom (UKDF)

Independent Data Access Centers (IDACs)

Access to proprietary data and the Science Platform require Rubin data rights

Rubin Science Platform

Provides access to LSST Data Products and services for all science users and project staff.



Credit: Leanne Guy

Credit: Leanne Guy



Rubin Data Product Categories



Prompt Data Products

Real Time Difference Image Analysis (DIA)

- Stream of ~10 million time-domain events per night (Alerts), transmitted to event distribution networks within 60s of camera readout.
- Images, Object and Source catalogs derived from DIA, and an orbit catalog for ~6 million Solar System bodies within 24h.
- Enables discovery and rapid follow-up of time domain events.



Data Release Data Products

Reduced single-epoch & deep co-added images, catalogs, reprocessed DIA products

- Catalogs of ~37 billion objects (20 billion galaxies, 17 billion stars), ~7 trillion sources and ~30 trillion forced source measurements.
- 11 Data Releases, produced ~annually over 10 years of operation.
- Accessible via the Rubin Science Platform (RSP) & Rubin Data Access Centers (DACs).



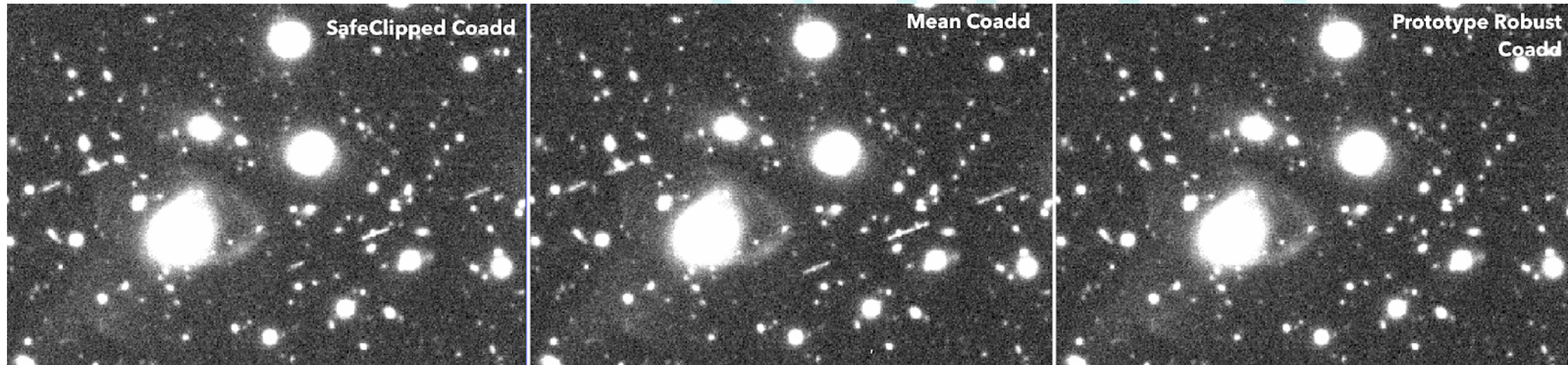
User Generated Data Products

User-produced derived, added-value data products

- Deep KBO/NEO, variable star classifications, shear maps, etc ...
- Enabled by services & computing resources at Rubin DACs and via the Rubin Science Platform (RSP).
- 10% of computing resources at the US Data Facility (USDF) will be allocated for User Generated data product storage & processing.

Credit: Leanne Guy

LSST Science Pipeline Data Products



You will get data products fast and slow

Annual Data Release Products (DRP)

11 Data releases in 10 years.

Final catalog: 15PB

Final pixels: 100PB

Prompt Data Products via nightly alert streams (Alert Production = AP)

~10 million alerts per night

issued within 60 s of shutter close



LSST Science Pipeline Data Products

You will get data products fast and slow

Annual Data Release Products (DRP)

11 Data releases in 10 years.

Final catalog: 15PB

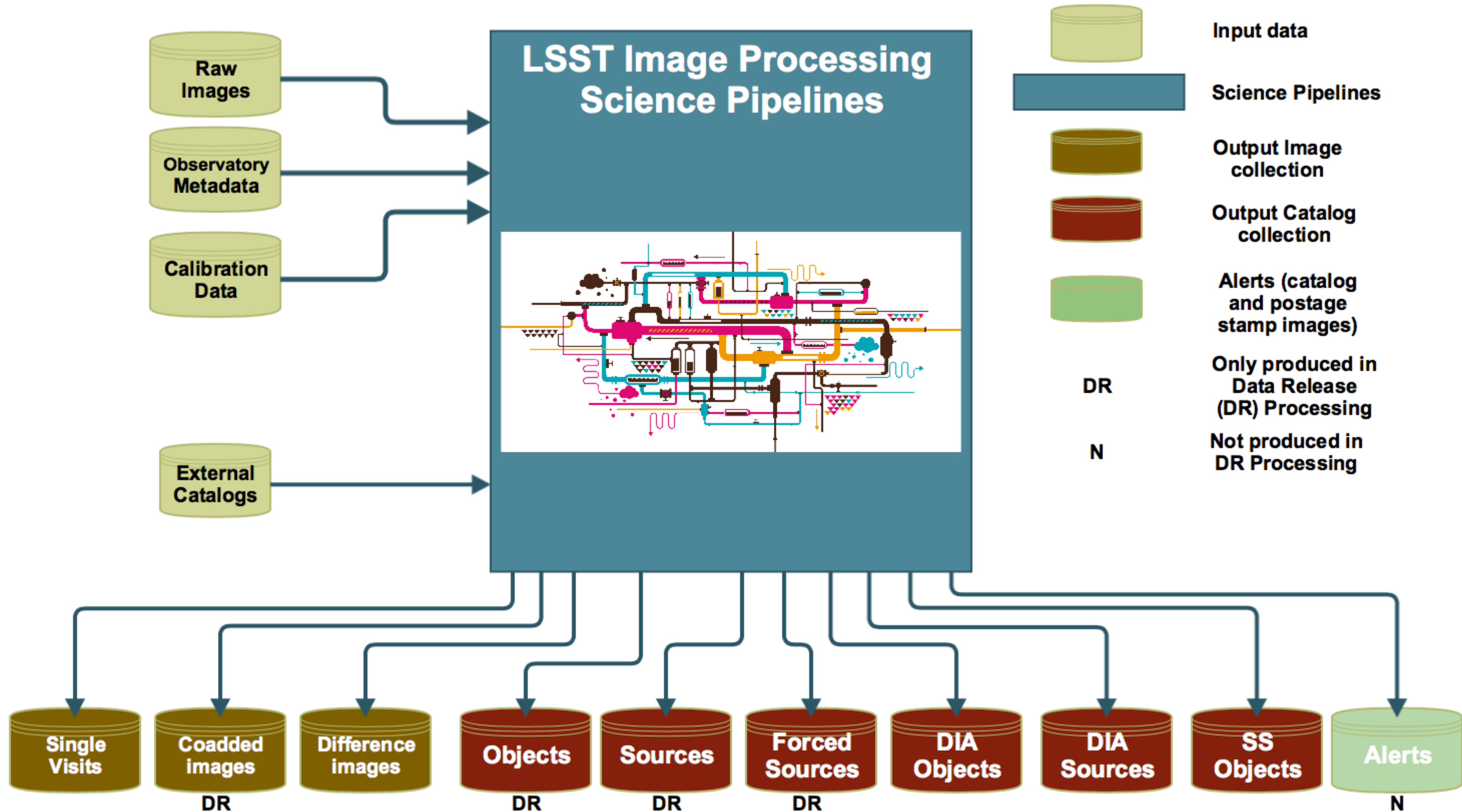
Final pixels: 100PB

Prompt Data Products via nightly alert streams (Alert Production = AP)

~10 million alerts per night

issued within 60 s of shutter close



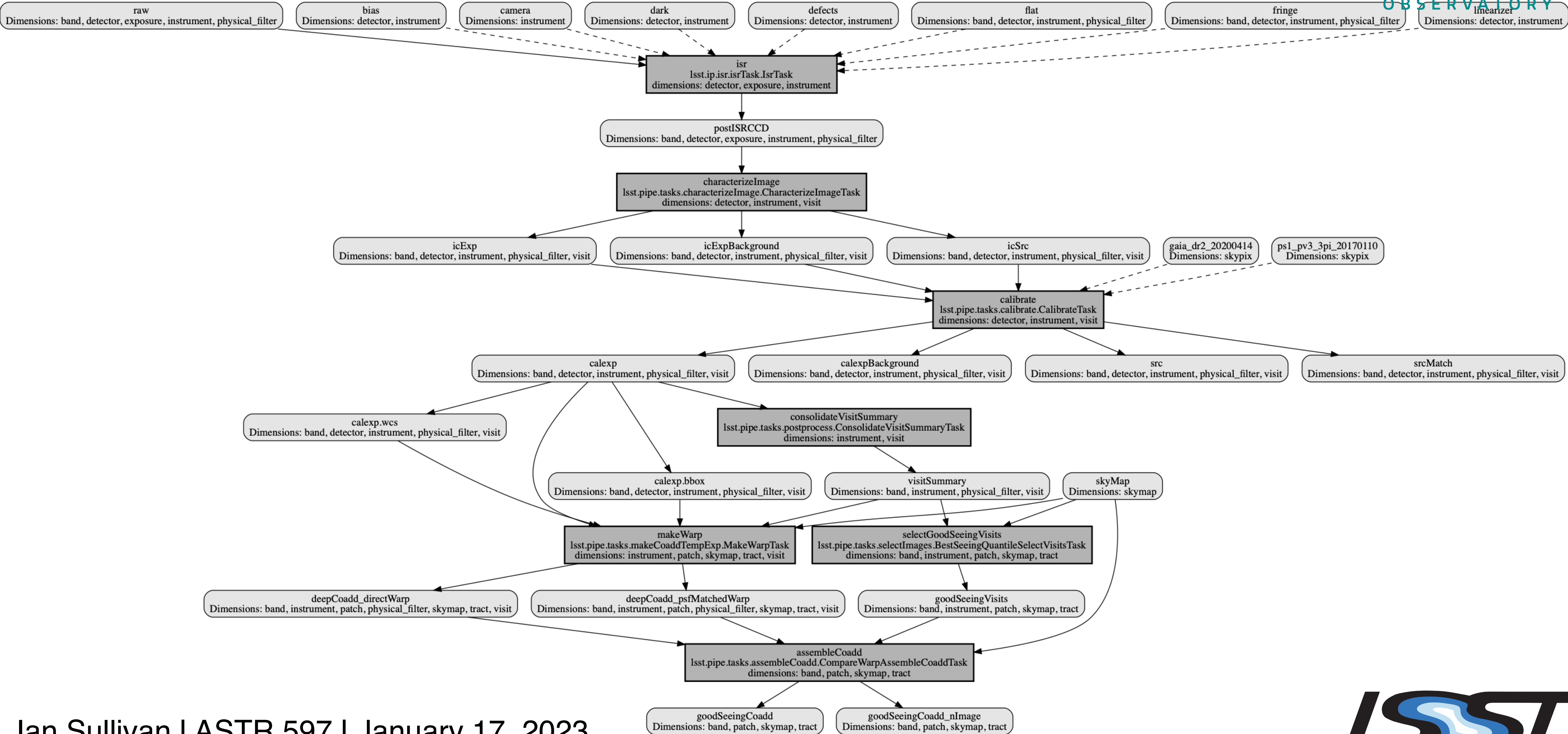


LSST Science Pipelines overview



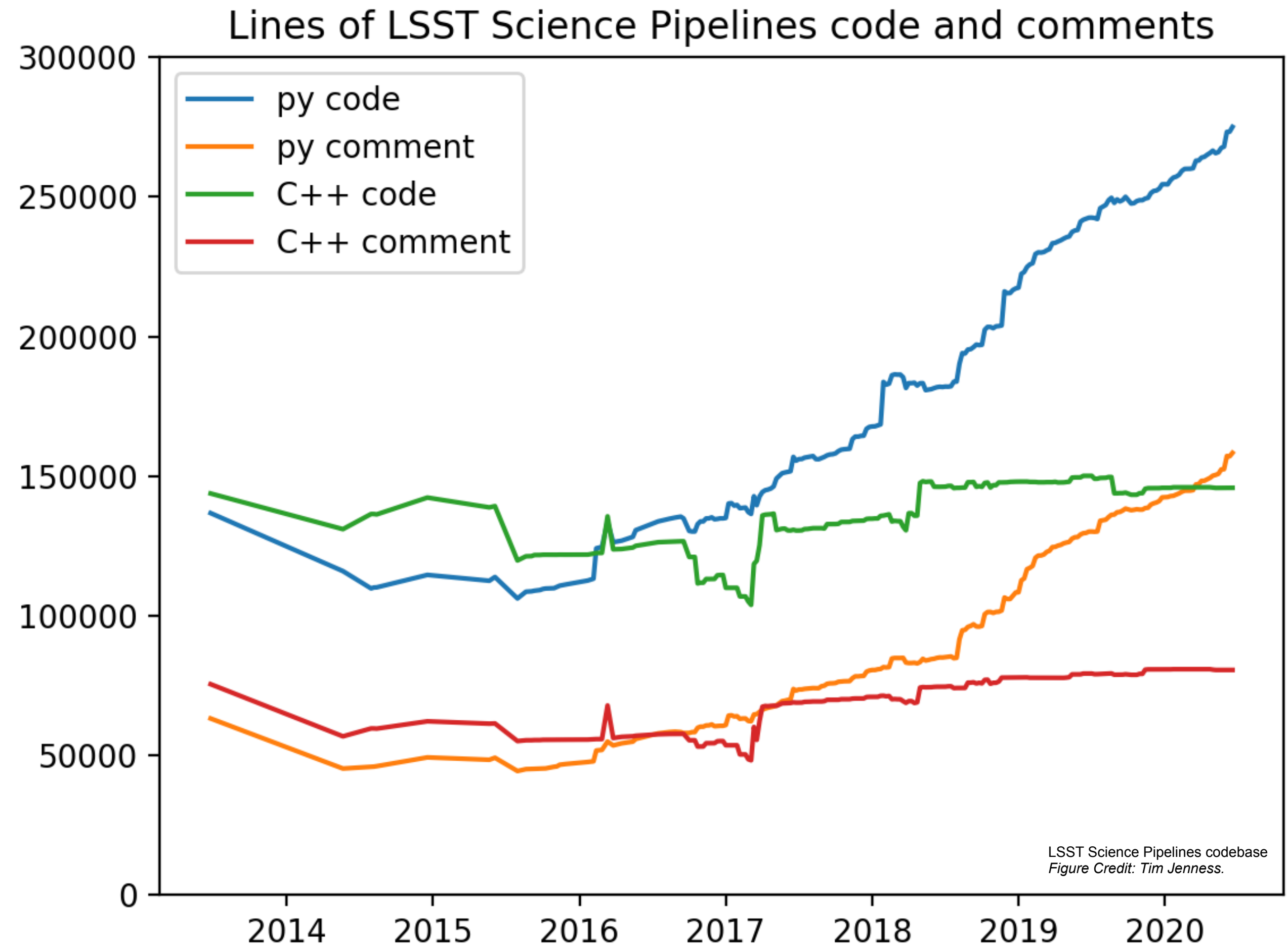
- What is a “pipeline”, anyway?
 - “a set of data processing elements connected in series, where the output of one element is the input of the next one.” (Wikipedia)
- For us, those data processing elements are the scientific algorithms, and the connections are data products.
- For a complex pipeline, we build a Directed Acyclic Graph of the connections
 - Predetermine every data product that will be produced, and the inputs and outputs of every algorithm
 - Each algorithm is executed as soon as all of the expected data products are available

Directed Acyclic Graph



How do we write code in a collaboration?

- Documentation
- Coding standards
- Unit tests
- Code review
- Version control
- Continuous integration



Science Pipelines documentation



LSST Science Pipelines

LSST Science Pipelines »

Search

On this page

[The LSST Science Pipelines](#)

[Getting started](#)

[Installation](#)

[Frameworks](#)

[Python modules](#)

[Packages](#)

[Release details](#)

[Indices](#)

[More info](#)

The LSST Science Pipelines

The LSST Science Pipelines are designed to enable optical and near-infrared astronomy in the “big data” era. While they are being developed to process the data for the [Rubin Observatory Legacy Survey of Space and Time \(Rubin’s LSST\)](#), our command line and programming interfaces can be extended to address any optical or near-infrared dataset.

This documentation covers version **v24_0_0**. [Learn what’s new](#). You can also find documentation for [other versions](#).

Getting started

If you’re new to the LSST Science Pipelines, these step-by-step data processing tutorials will get you up and running.

Data processing tutorial series (these were developed using the `w_2021_33` version of the science pipelines):

- Part 1 [Data repositories](#)
- Part 2 [Single frame processing](#)
- Part 3 [Image and catalog display](#)
- Part 4 [Global calibration](#)
- Part 5 [Image coaddition](#)
- Part 6 [Source measurement](#)
- Part 7 [Multi-band catalog analysis](#).

<https://pipelines.lsst.io/>

LSST developer guide



Edition: main

Change edition

Search docs

TEAM

[Onboarding Checklist](#)

[Team Culture and Conduct Standards](#)

[Focus Friday](#)

[Meeting Free Weeks](#)

[Empowerment of DM team members](#)

[Data Release Production](#)

COMMUNICATIONS

[Slack Culture](#)

[Configuring your GitHub username in your Slack profile](#)

[Home](#) / [LSST DM Developer Guide](#)

[Edit on GitHub](#)

LSST DM Developer Guide

This is an internal guide for [LSST DM](#) staff during project construction, and for Rubin Observatory Data Production staff in operations. It's also openly available so that others can understand how we're building the LSST's data management subsystem. In some cases, other Rubin groups (for example Telescope & Site Software) have chosen to follow various sections as it applies to them.

This guide includes a mix of normative requirements and helpful, descriptive, pages. When it's particularly important that you closely follow a standard, we include an annotation box at the top of the page.

Any member of DM can contribute to this guide. It's published from the https://github.com/lsst-dm/dm_dev_guide GitHub repo. Check out the [README](#) to get started.

Jump to: [Team](#) · [Communications](#) · [Project documentation](#) · [Work management](#)

Development guides: [Overview](#) · [C++](#) · [Python](#) · [Pybind11](#) · [JavaScript](#) · [ReStructuredText](#) · [DM Stack](#) · [Git](#) · [Editors](#) · [Legal](#) · [User documentation style](#)

Services: [Overview](#) · [Jenkins](#)

<https://developer.lsst.io/>

Unit tests

- Tests of a single feature, or of required behavior
- Every line of code should be covered by a test
- Tests ensure that future code changes or new features maintain existing functionality
- Tests of python code can be run by pytest, which will provide a test coverage report and show missed statements or code branches

```
class AlardLuptonSubtractTest(lsst.utils.tests.TestCase):  
  
    def test_allowed_config_modes(self):  
        """Verify the allowable modes for convolution.  
        """  
        config = subtractImages.AlardLuptonSubtractTask.ConfigClass()  
        config.mode = 'auto'  
        config.mode = 'convolveScience'  
        config.mode = 'convolveTemplate'  
  
        with self.assertRaises(FieldValidationError):  
            config.mode = 'aotu'  
  
    def test_mismatched_template(self):  
        """Test that an error is raised if the template  
        does not fully contain the science image.  
        """  
        xSize = 200  
        ySize = 200  
        science, sources = makeTestImage(psfSize=2.4, xSize=xSize + 20, ySize=ySize + 20)  
        template, _ = makeTestImage(psfSize=2.4, xSize=xSize, ySize=ySize, doApplyCalibration=True)  
        config = subtractImages.AlardLuptonSubtractTask.ConfigClass()  
        task = subtractImages.AlardLuptonSubtractTask(config=config)  
        with self.assertRaises(AssertionError):  
            task.run(template, science, sources)  
  
    def test_equal_images(self):
```

Code review



DM-36265: Fix subtractImages failures when compatibility mode is off #236

Merged isullivan merged 1 commit into main from tickets/DM-36265 on Nov 9, 2022

Conversation 1 Commits 1 Checks 1 Files changed 1

isullivan commented on Nov 1, 2022

Use only the smaller of the template and science bbox for calculating the matching kernel.

yalsayyad reviewed on Nov 8, 2022

```
python/lsst/ip/diffim/makeKernel.py Outdated
... @@ -330,7 +332,7 @@ def makeKernelBasisList(self, targetFwhmPix=None,
330 332
331 333         return basisList
332 334
333 - def _buildCellSet(self, templateMaskedImage, scienceMaskedImage, candidateList):
335 + def _buildCellSet(self, templateMaskedImage, scienceMaskedImage, candidateList,
        imageBBox):
```

yalsayyad on Nov 8, 2022 · edited

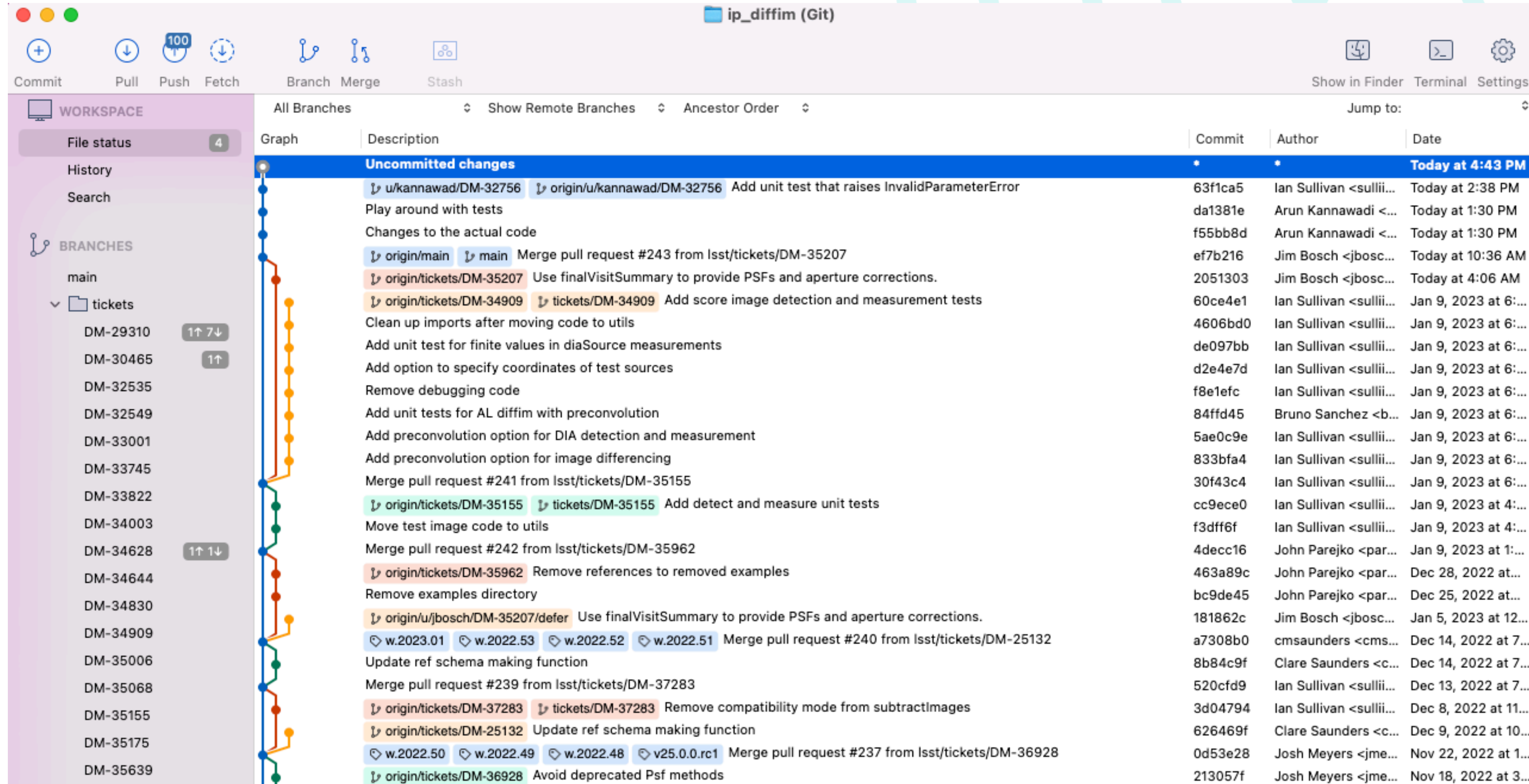
This seems like an unnecessary interface change to me:

1. You don't use your `bbox` variable again in code that calls it after line 123.
2. `imageBBox` can be exactly determined from the other two inputs already there: `templateMaskedImage` and `scienceMaskedImage`.

Recommendation would be, right before constructing the `SpatialCellSet` in `kernelCellSet = lsst.afw.math.SpatialCellSet(imageBBox, sizeCellX, sizeCellY)` below you could:

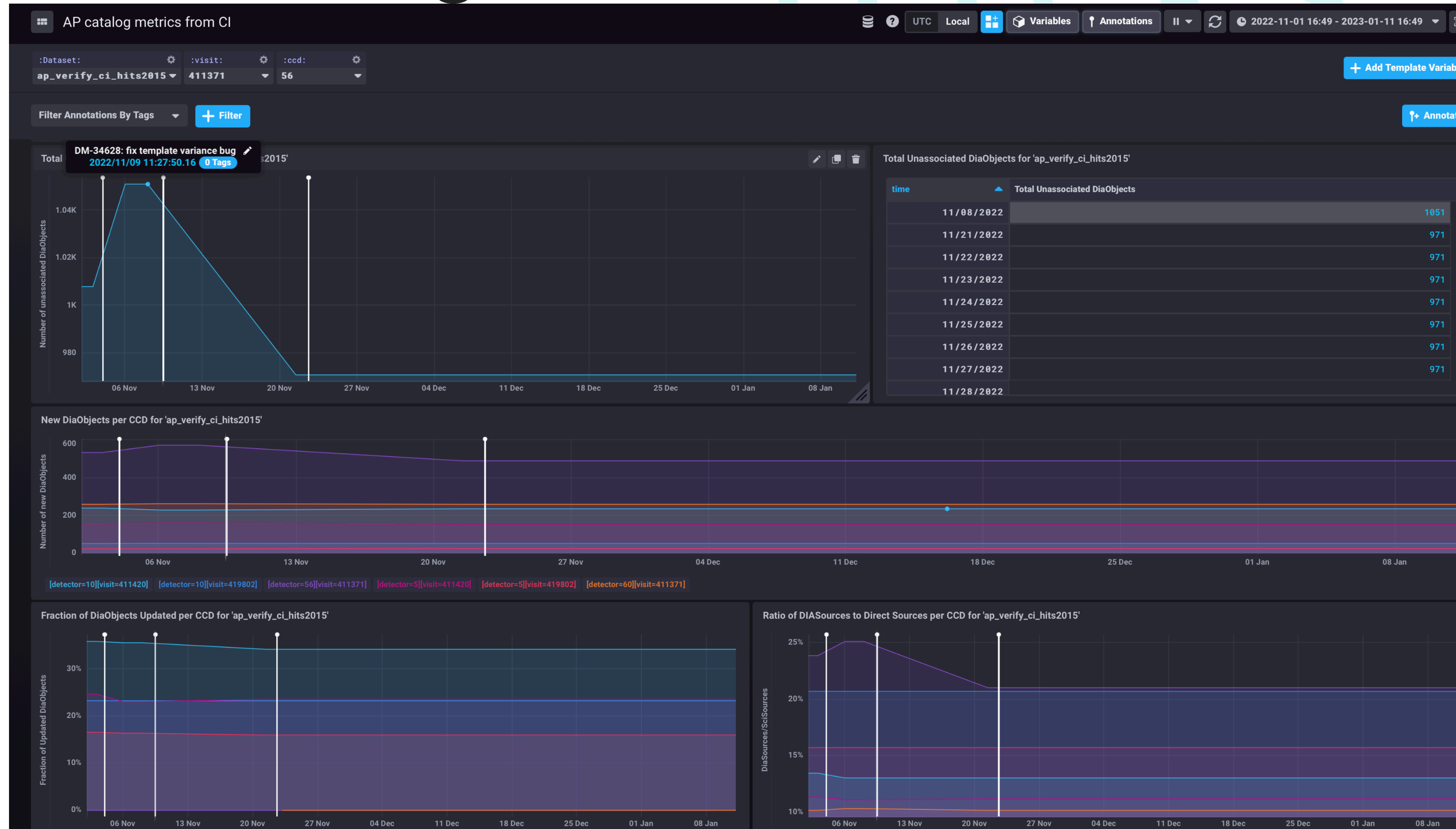
```
imageBBox = templateMaskedImage.getBBox()
imageBBox = imageBBox.clip(scienceMaskedImage.getBBox())
```


Version control



Commit	Author	Date
•	•	Today at 4:43 PM
63f1ca5	Ian Sullivan <sullii...>	Today at 2:38 PM
da1381e	Arun Kannawadi <...>	Today at 1:30 PM
f55bb8d	Arun Kannawadi <...>	Today at 1:30 PM
ef7b216	Jim Bosch <jbosc...>	Today at 10:36 AM
2051303	Jim Bosch <jbosc...>	Today at 4:06 AM
60ce4e1	Ian Sullivan <sullii...>	Jan 9, 2023 at 6:...
4606bd0	Ian Sullivan <sullii...>	Jan 9, 2023 at 6:...
de097bb	Ian Sullivan <sullii...>	Jan 9, 2023 at 6:...
d2e4e7d	Ian Sullivan <sullii...>	Jan 9, 2023 at 6:...
f8e1efc	Ian Sullivan <sullii...>	Jan 9, 2023 at 6:...
84ffd45	Bruno Sanchez <b...>	Jan 9, 2023 at 6:...
5ae0c9e	Ian Sullivan <sullii...>	Jan 9, 2023 at 6:...
833bfa4	Ian Sullivan <sullii...>	Jan 9, 2023 at 6:...
30f43c4	Ian Sullivan <sullii...>	Jan 9, 2023 at 6:...
cc9ece0	Ian Sullivan <sullii...>	Jan 9, 2023 at 4:...
f3dff6f	Ian Sullivan <sullii...>	Jan 9, 2023 at 4:...
4decc16	John Parejko <par...>	Jan 9, 2023 at 1:...
463a89c	John Parejko <par...>	Dec 28, 2022 at...
bc9de45	John Parejko <par...>	Dec 25, 2022 at...
181862c	Jim Bosch <jbosc...>	Jan 5, 2023 at 12:...
a7308b0	cmsaunders <cms...>	Dec 14, 2022 at 7:...
8b84c9f	Clare Saunders <c...>	Dec 14, 2022 at 7:...
520cfd9	Ian Sullivan <sullii...>	Dec 13, 2022 at 7:...
3d04794	Ian Sullivan <sullii...>	Dec 8, 2022 at 11:...
626469f	Clare Saunders <c...>	Dec 9, 2022 at 10:...
0d53e28	Josh Meyers <jme...>	Nov 22, 2022 at 1:...
213057f	Josh Meyers <jme...>	Nov 18, 2022 at 3:...

Continuous integration



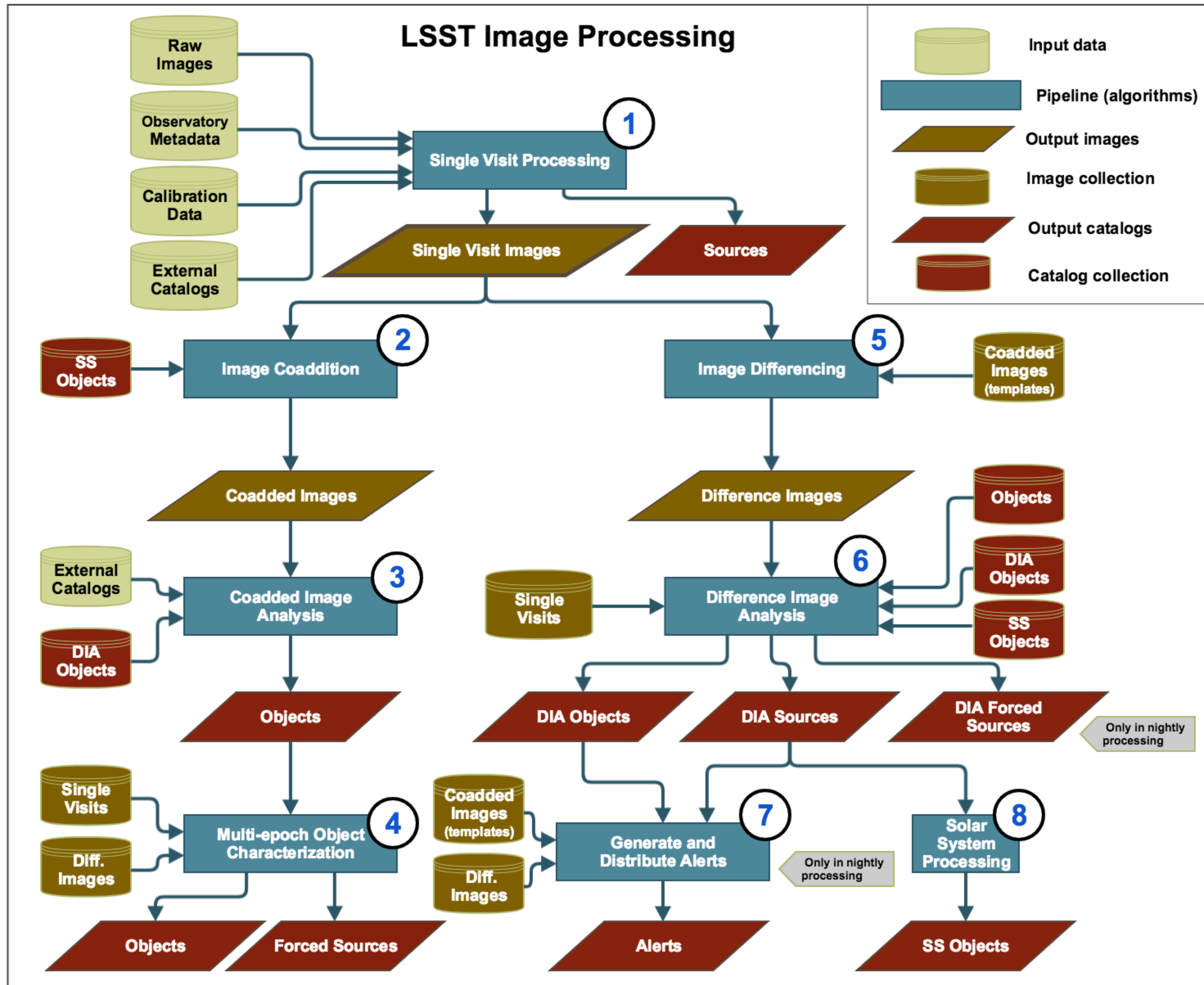
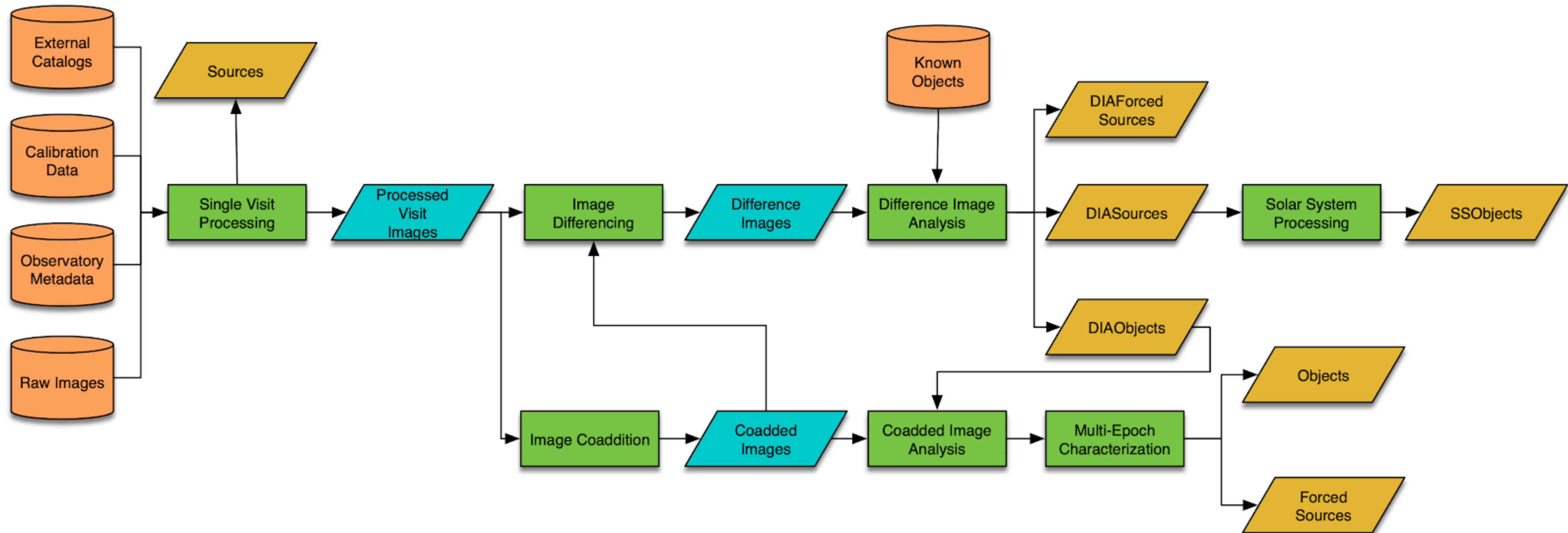
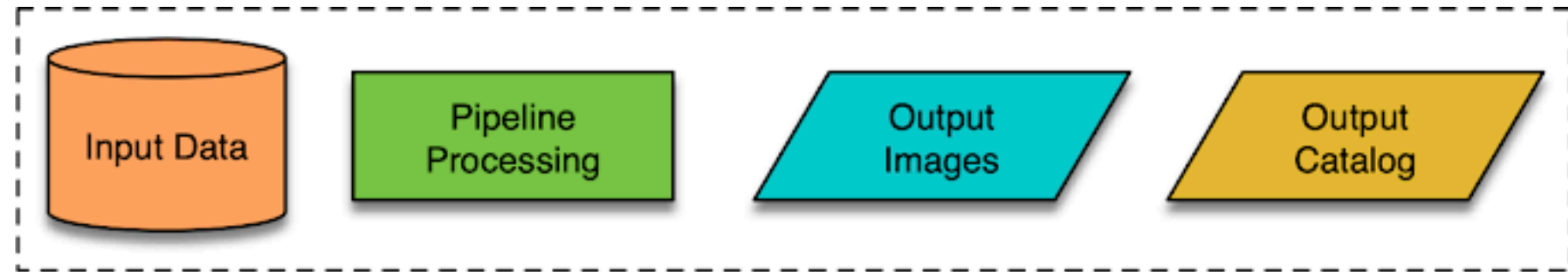


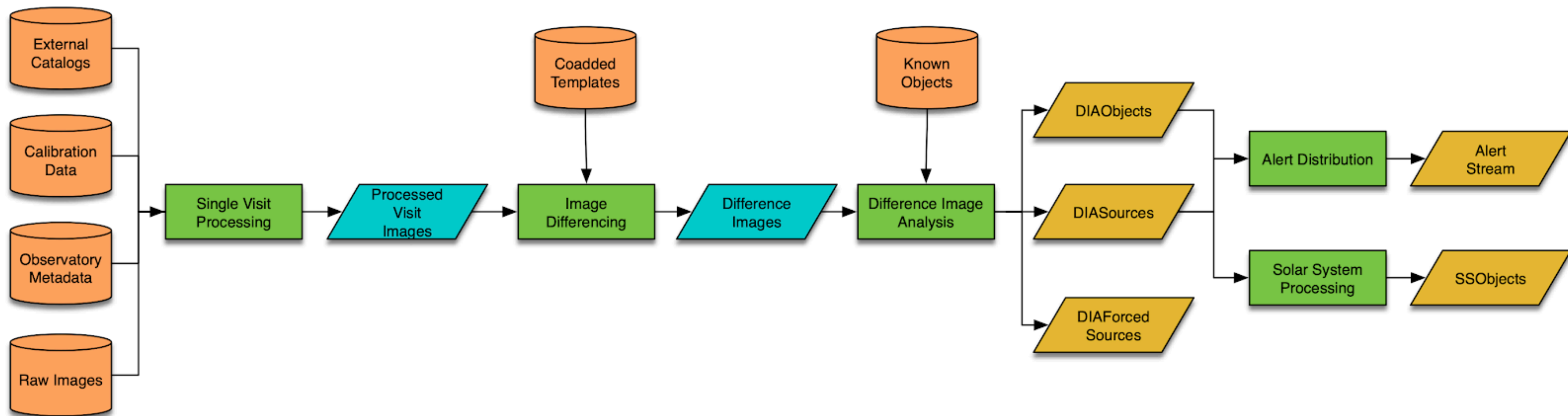
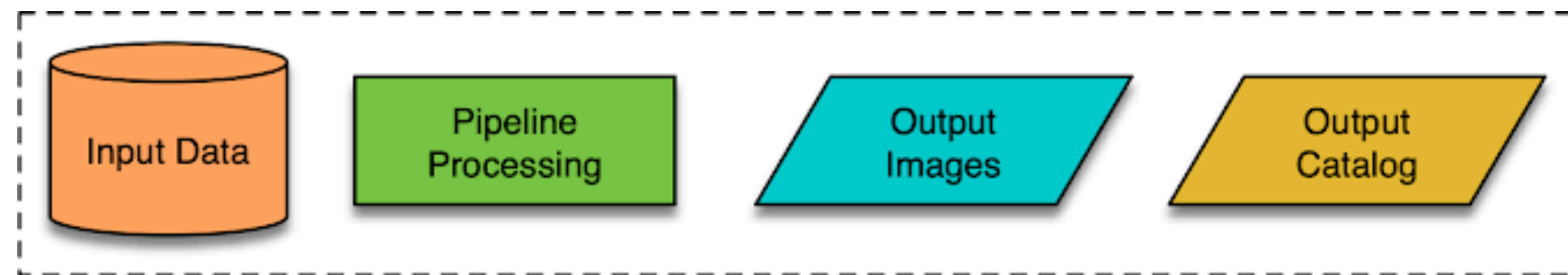
Figure from ls.st/dpdd

Data Release Processing Overview



Produced once a year after the first Data Release

Prompt Processing Overview

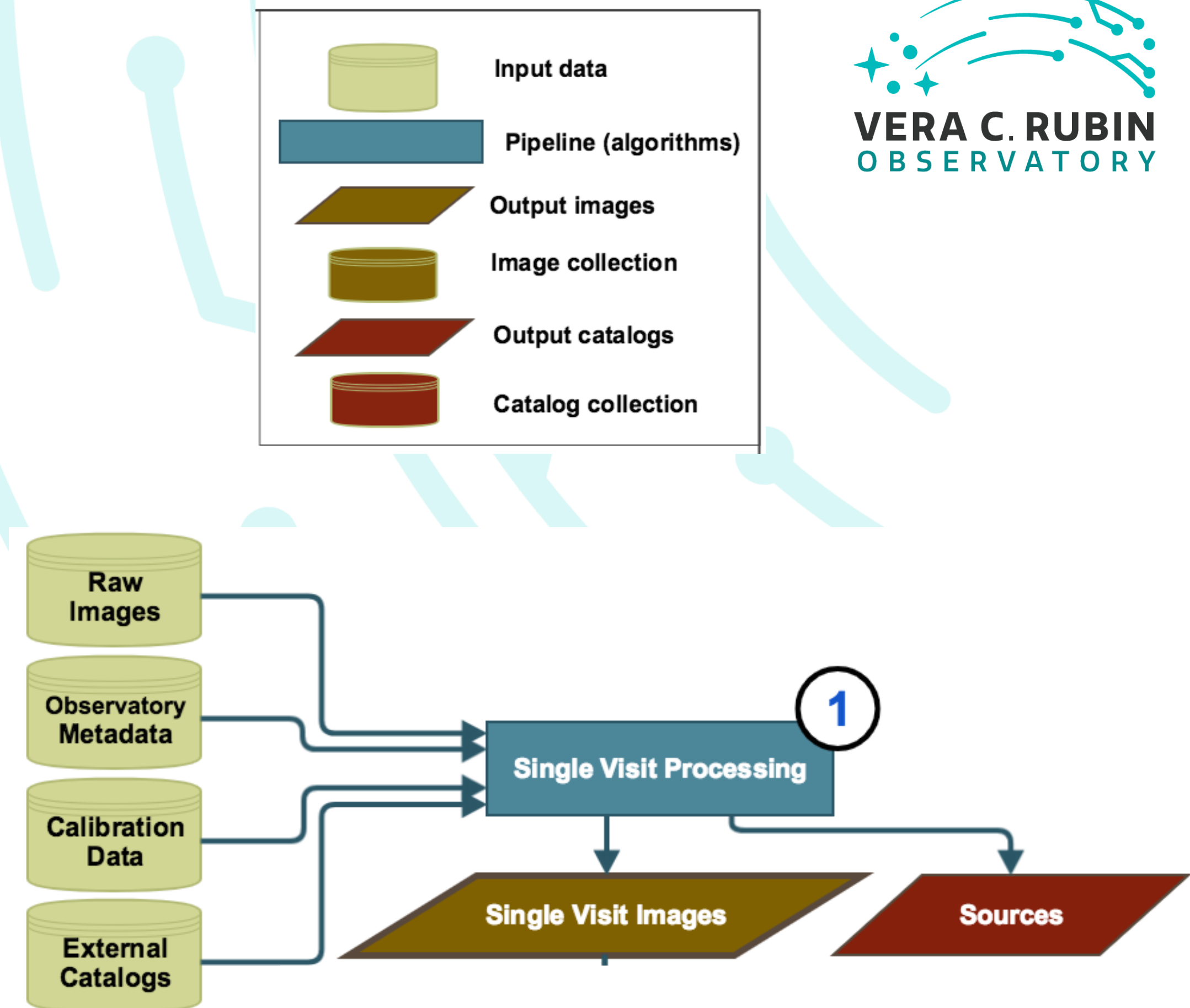


Produced nightly, in close to real time

1: Single Visit Processing

AP and DRP

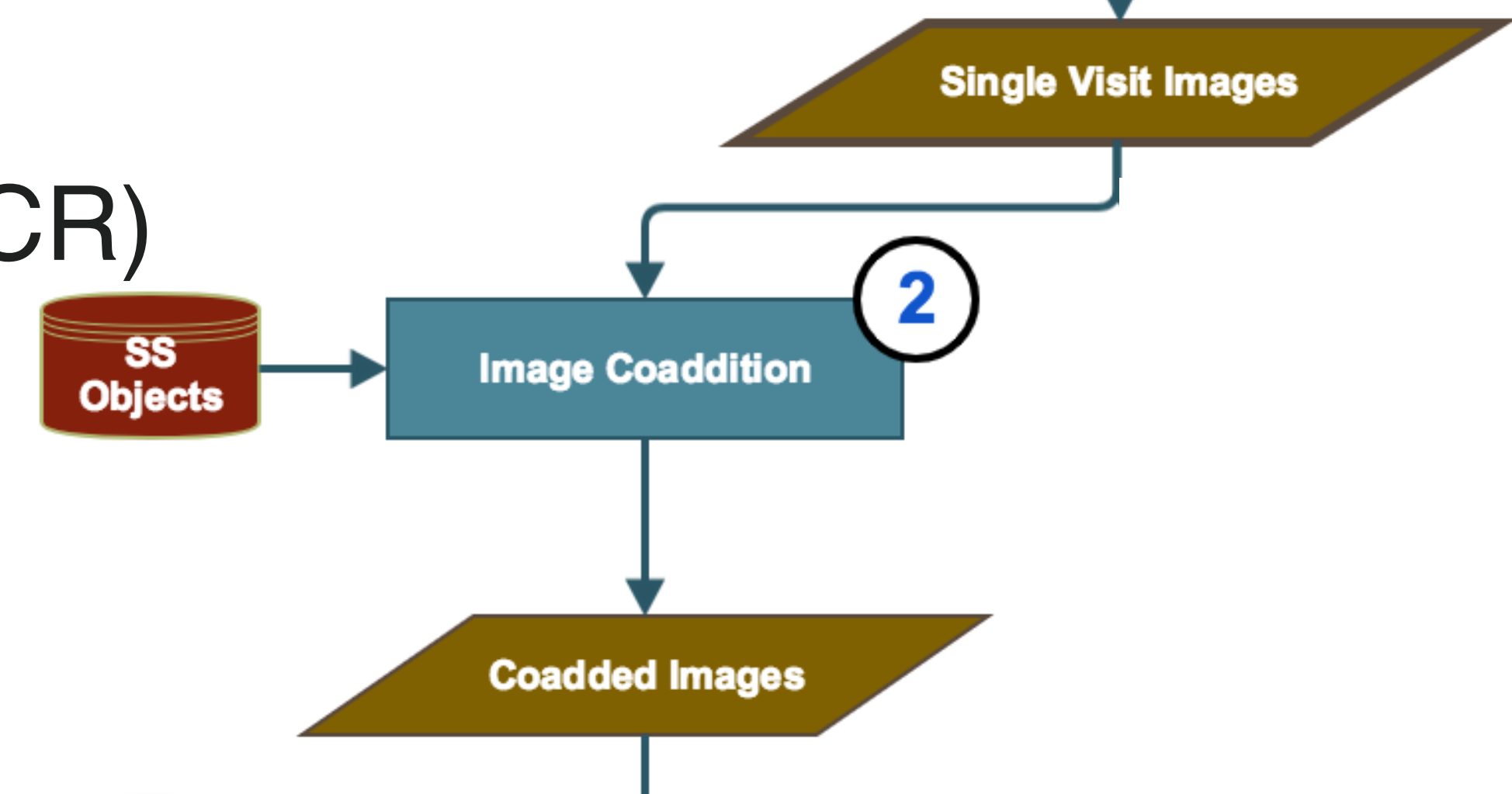
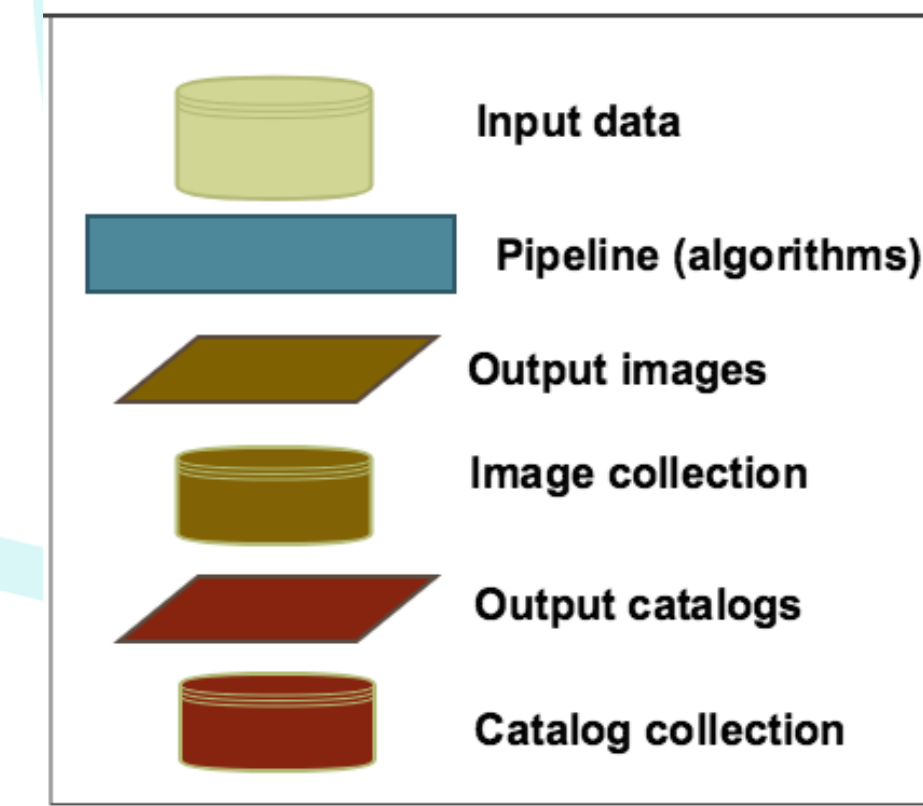
- Principal algorithmic components:
 - Image characterization
 - Background fitting
 - PSF modeling
 - Cosmic ray removal
 - Calibration
 - Astrometric
 - Photometric
- Data products:
 - Source catalogs (**src**)
 - Calibrated images (**calexp**)



2: Image Coaddition

DRP

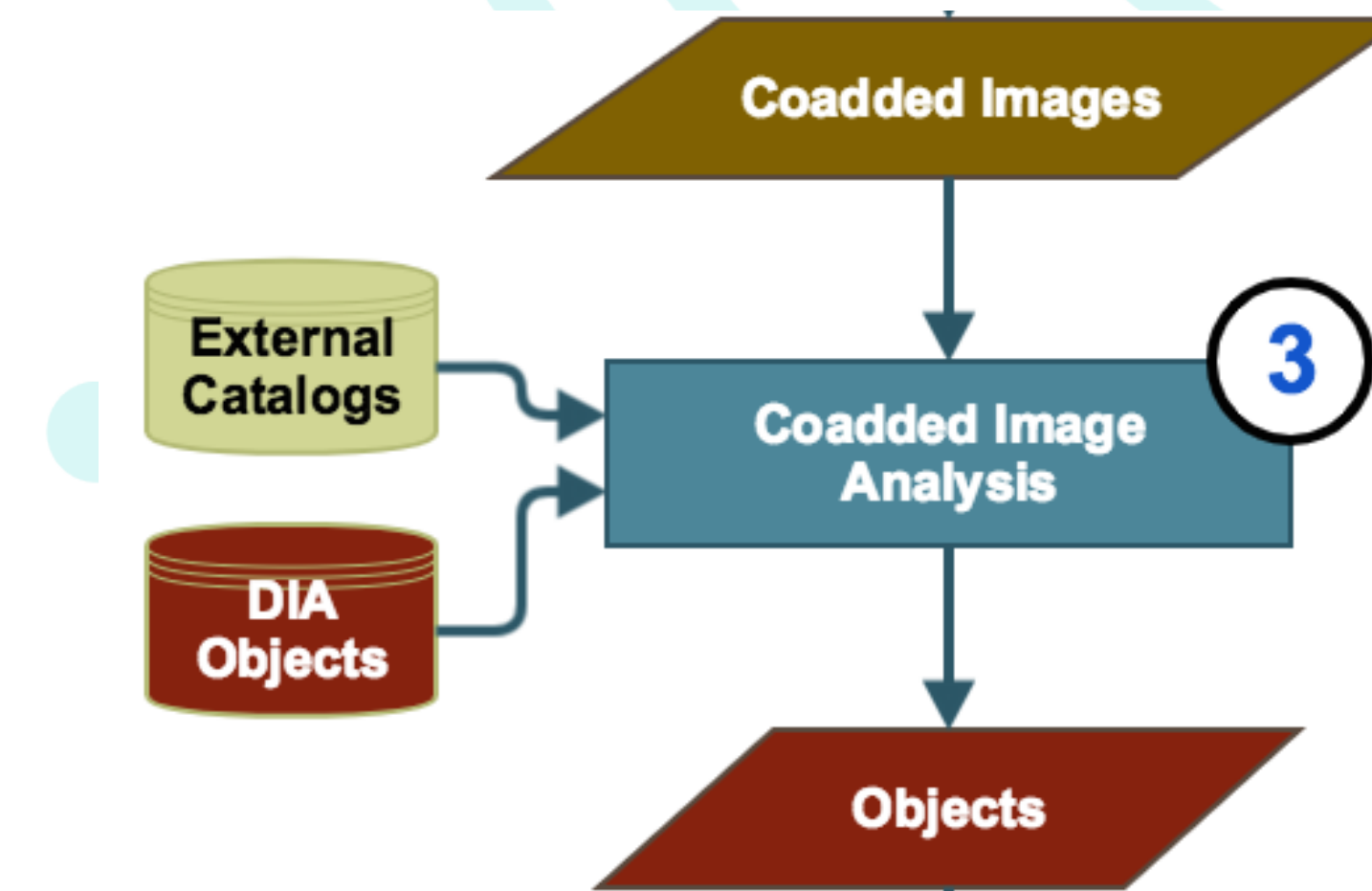
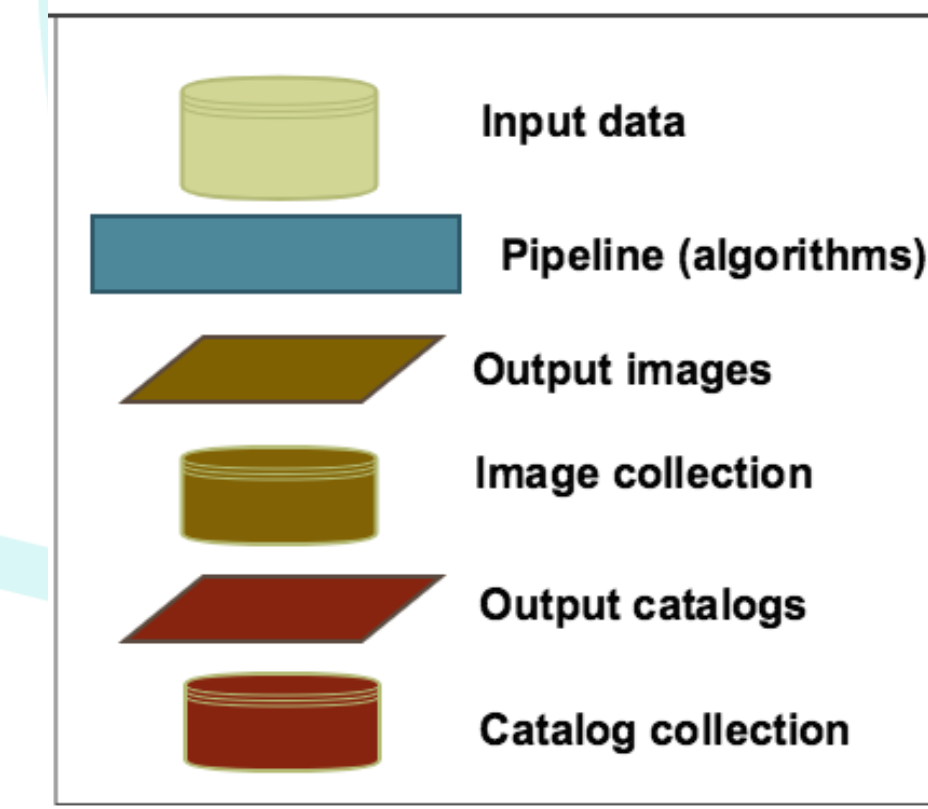
- Principal algorithmic components
 - Warping calibrated images to a skymap
 - PSF-matching
 - Identifying and masking artifacts
 - Modeling Differential Chromatic Refraction (DCR)
 - Stacking warped images
- Data products
 - Coadded images (e.g. **deepCoadd**, **goodSeeingCoadd**, **dcrCoadd**, ...)



3: Coadded Image Analysis

DRP

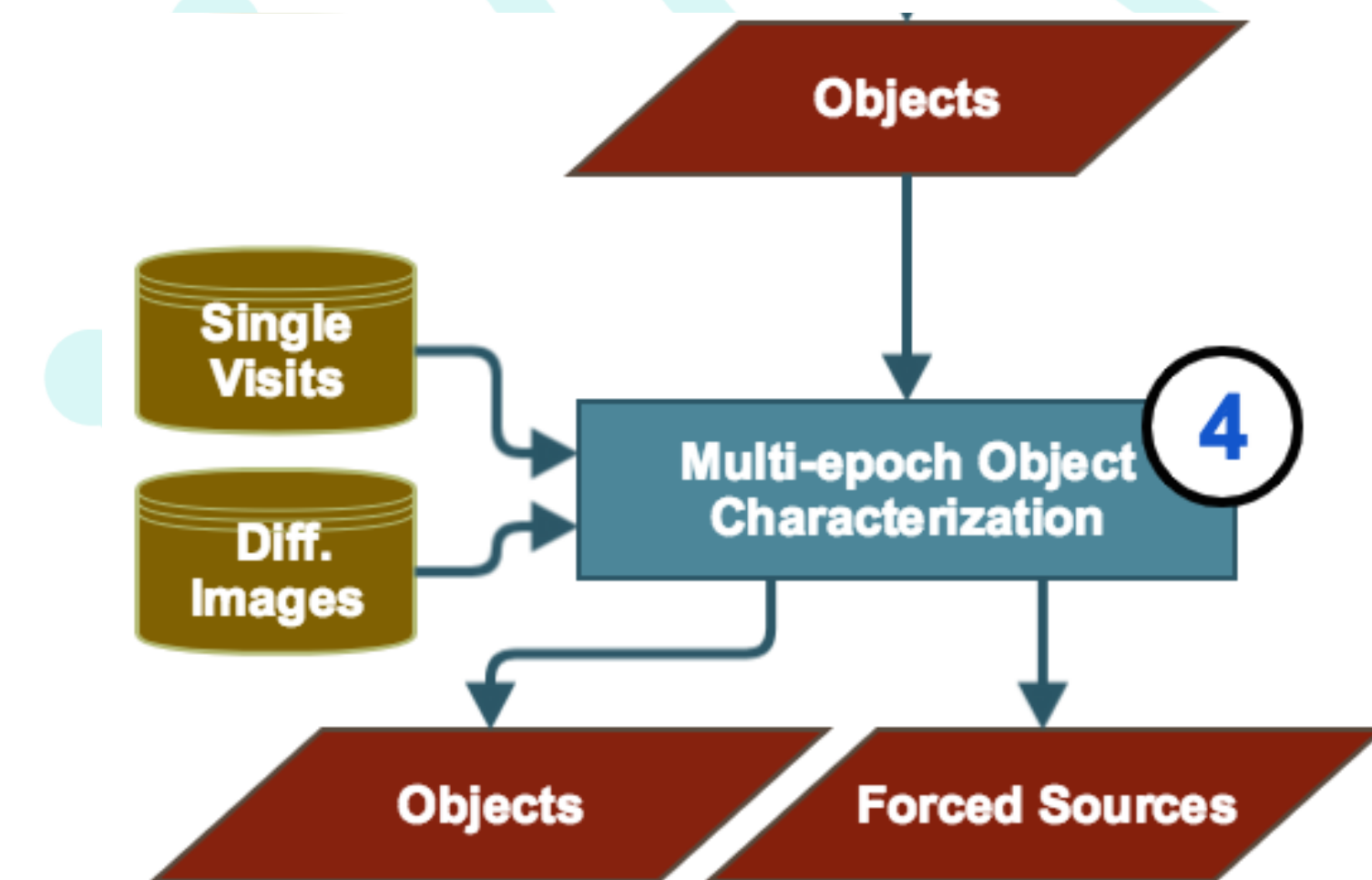
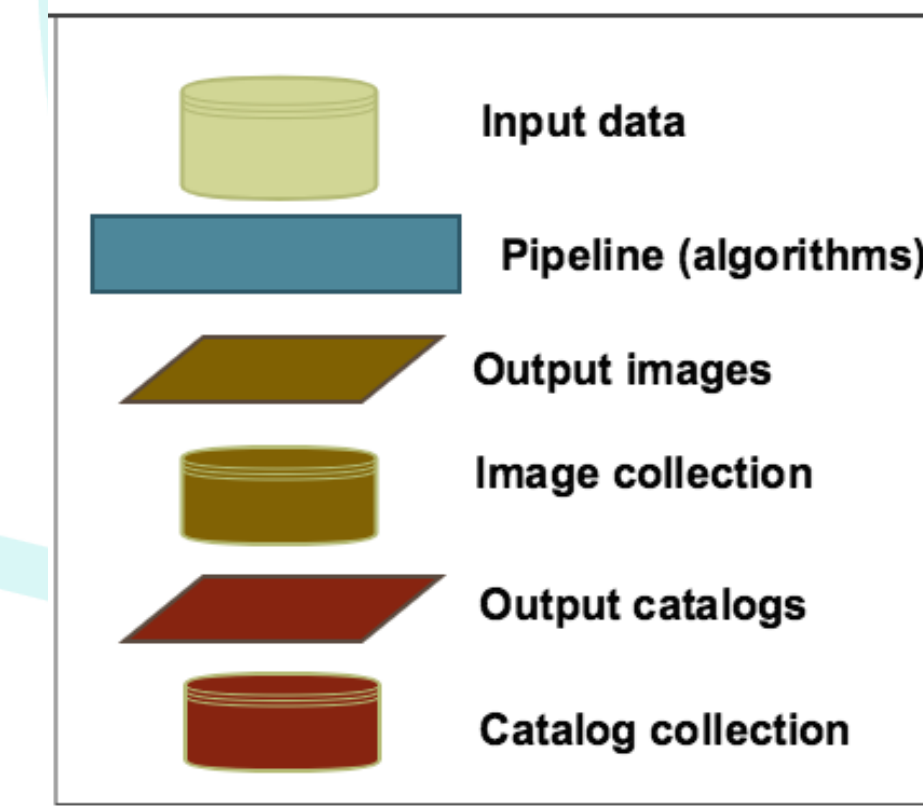
- Principal algorithmic components
 - Detection of static sources
 - Deblending sources
 - Merging sources into objects
 - Measurements on objects
- Data products
 - Object catalogs (* **Coadd**)



4: Object Characterization

DRP

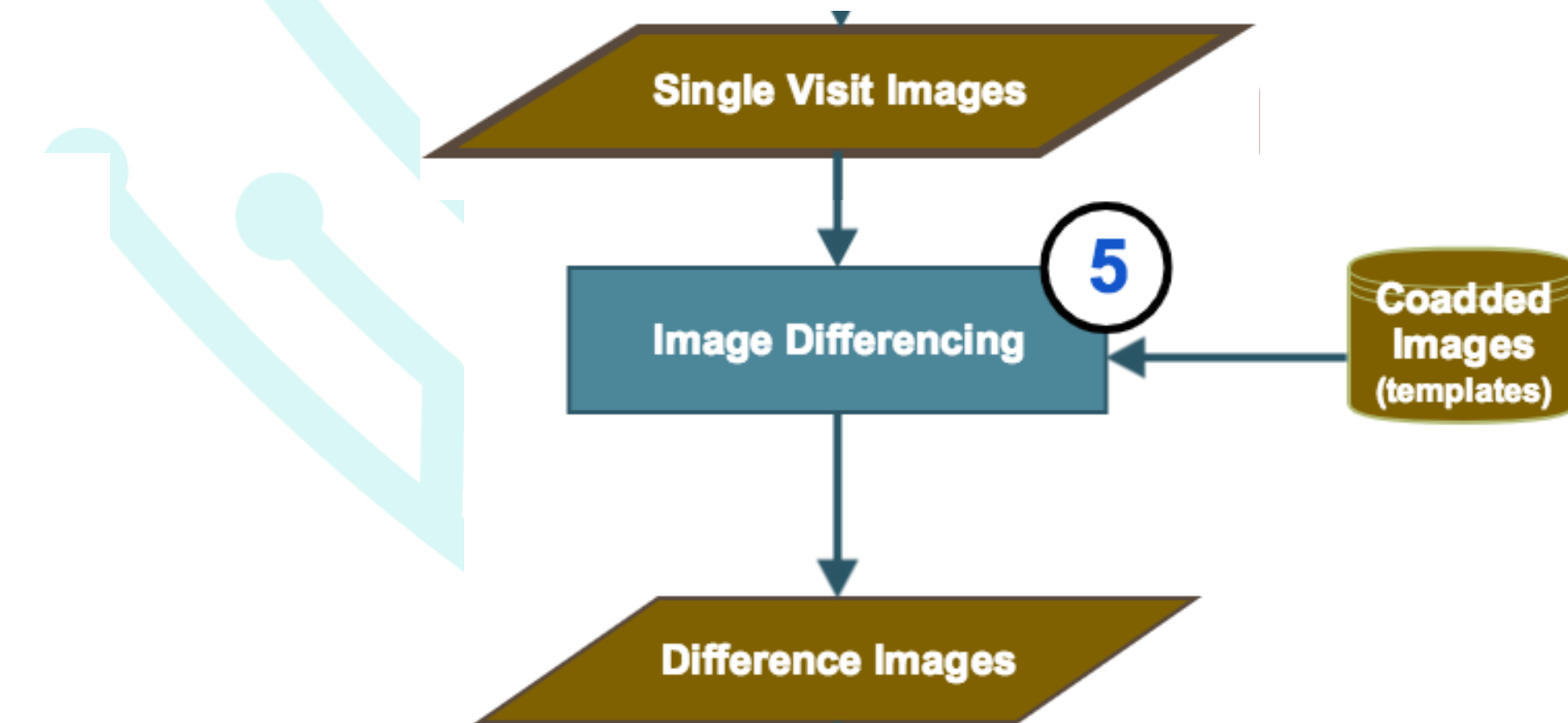
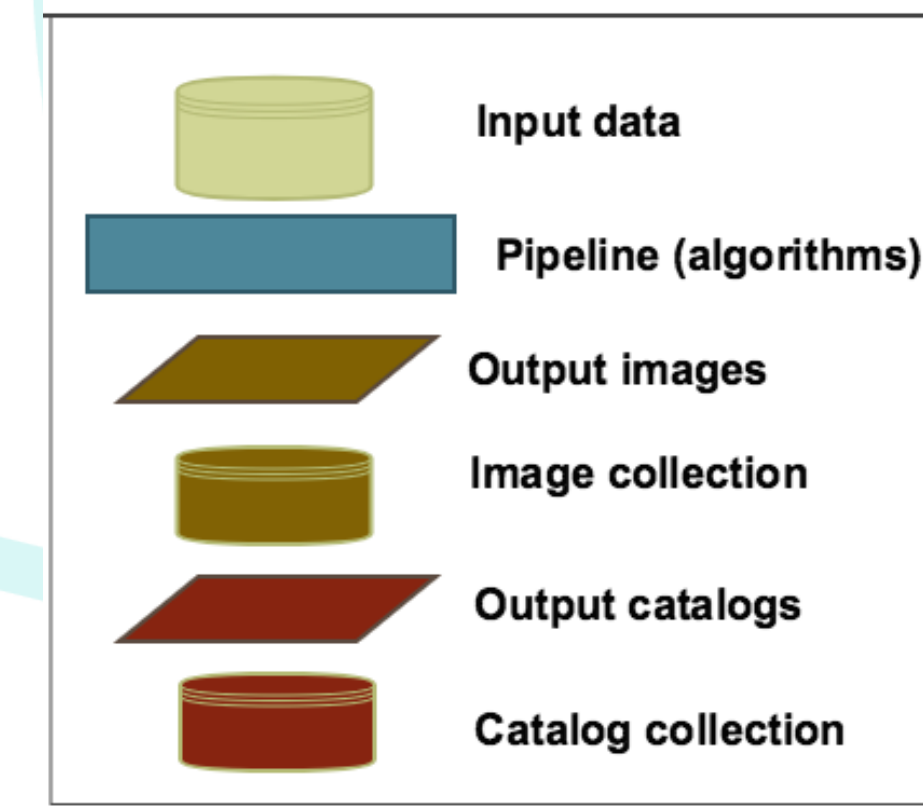
- Principal algorithmic components
 - Source measurements
 - Stellar motion fitting
 - Variability measurements
 - Photometric redshifts
- Data products
 - Object catalogs (* **Coadd_**)
 - Forced source catalogs (* **Coadd_forcedSrc**)



5: Image Differencing

AP and DRP

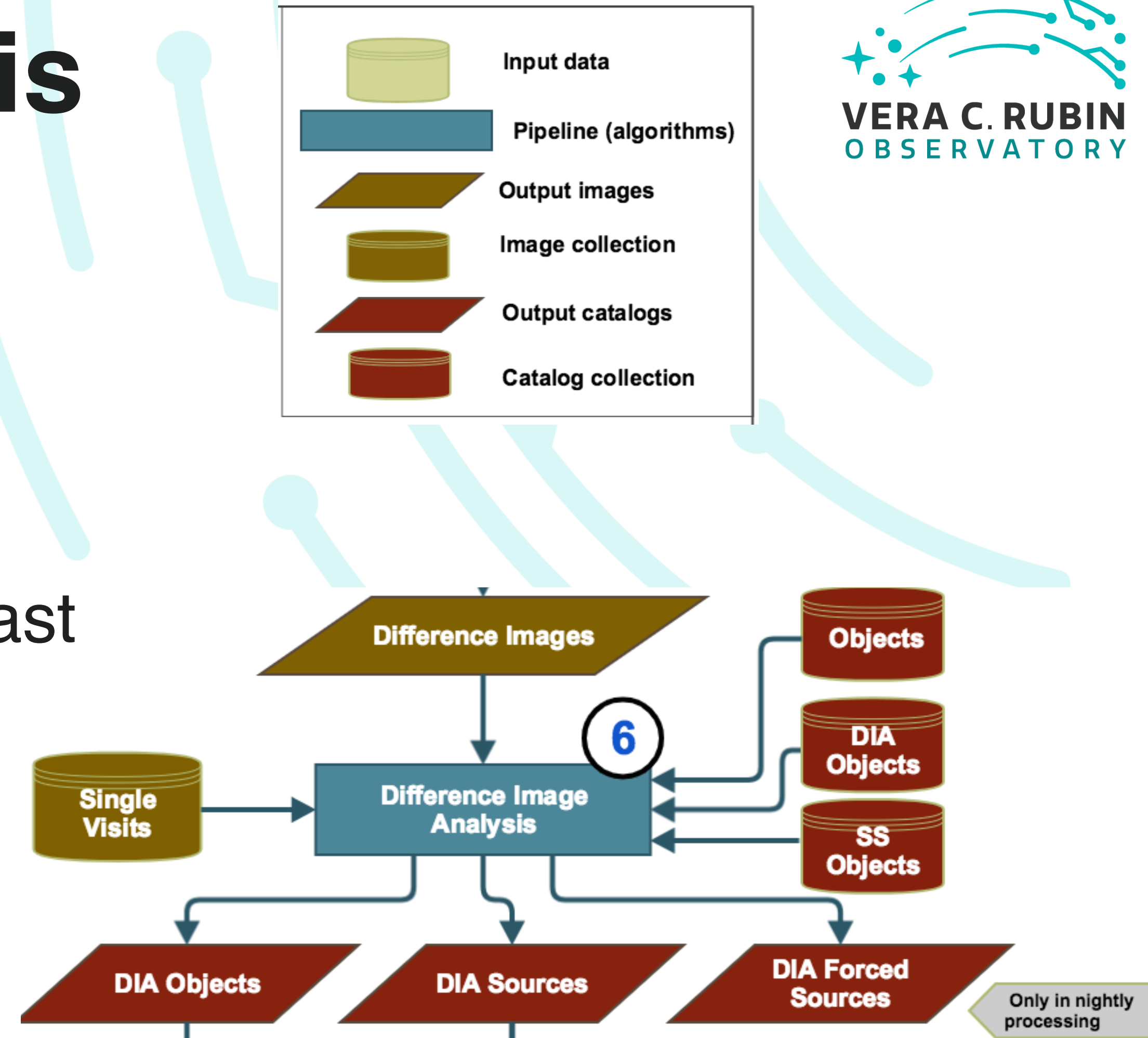
- Principal algorithmic components
 - Warping (template)
 - PSF matching
- Data products
 - Difference images (* **Diff_differenceExp**)



6: Difference Image Analysis

AP and DRP

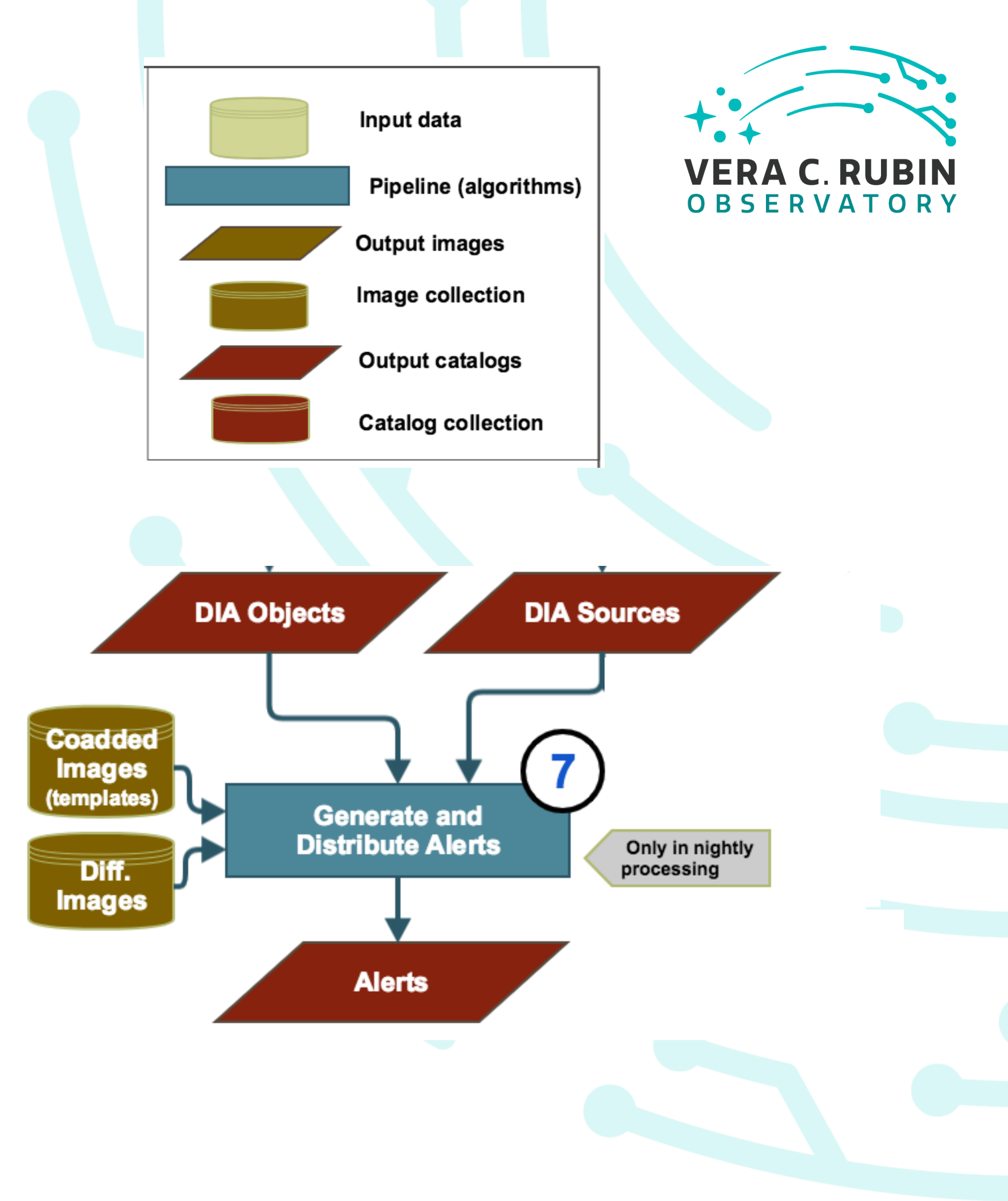
- Principal algorithmic components
 - Detection of DIA sources
 - Measurement
 - Forced measurement on objects from last 30 days
 - Object association
- Data products
 - Difference Image Analysis (DIA) sources (***Diff_diaSrc**)
 - DIA objects (**apdb** database)



7: Alert Distribution

AP

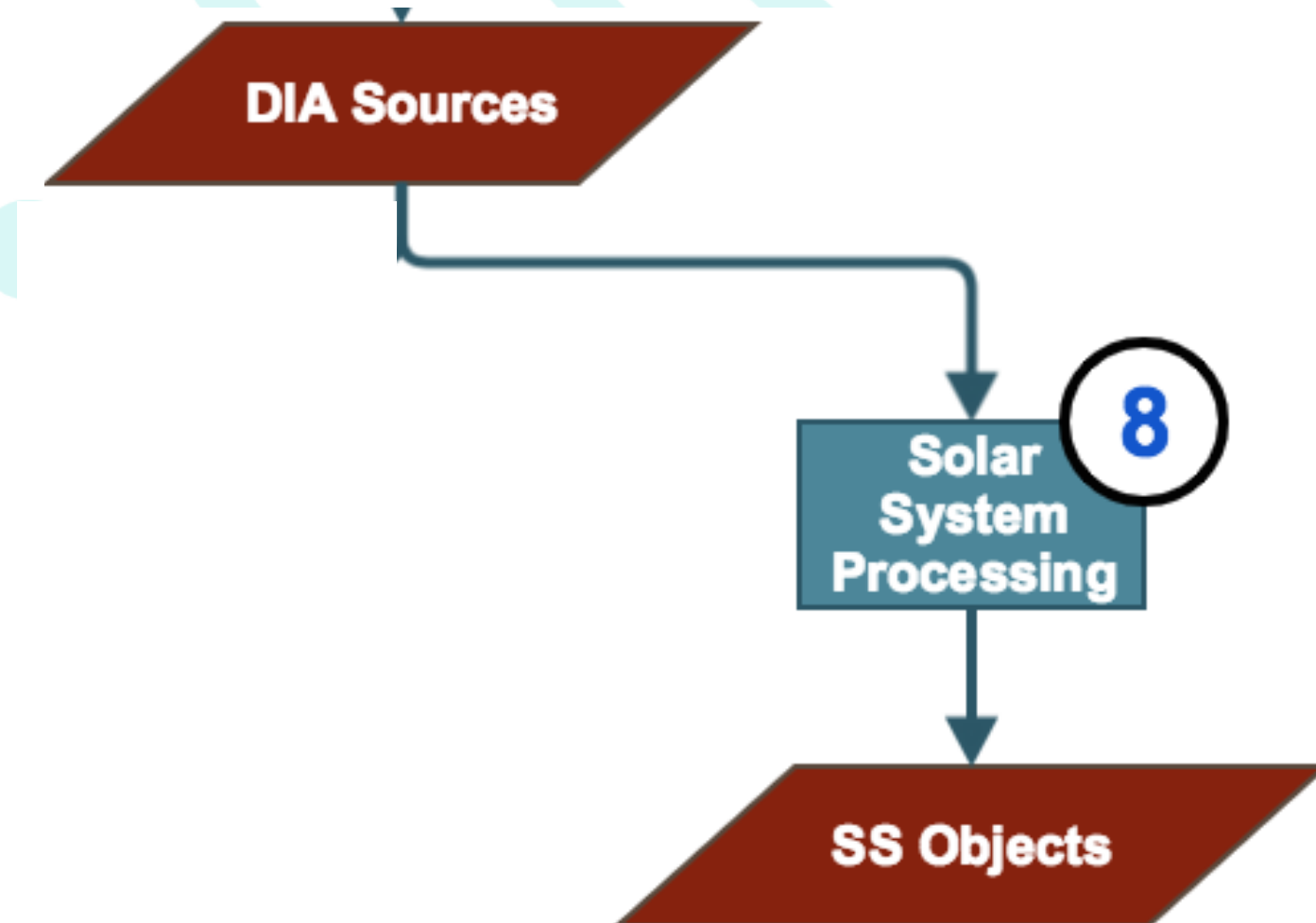
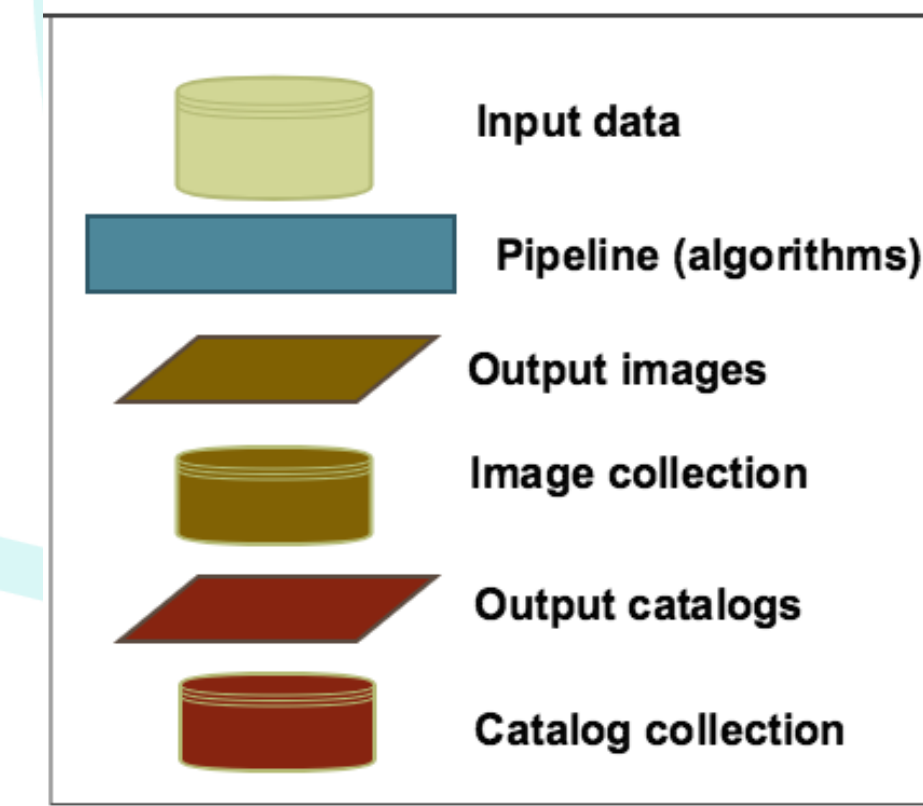
- Principal algorithmic components
 - Alert packaging
 - Cutout extraction
 - Alert filtering
- Data products
 - Packaged alerts



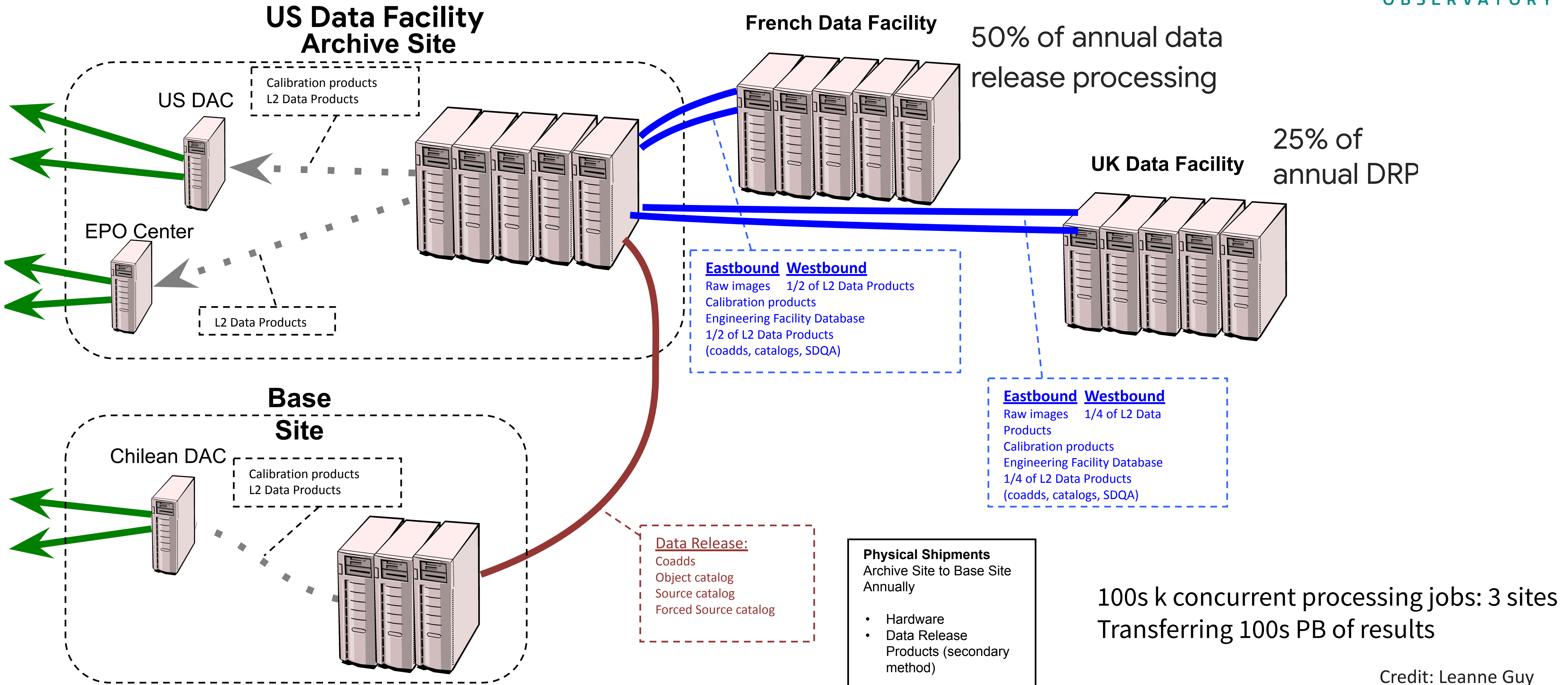
8: Solar System Processing

AP and DRP

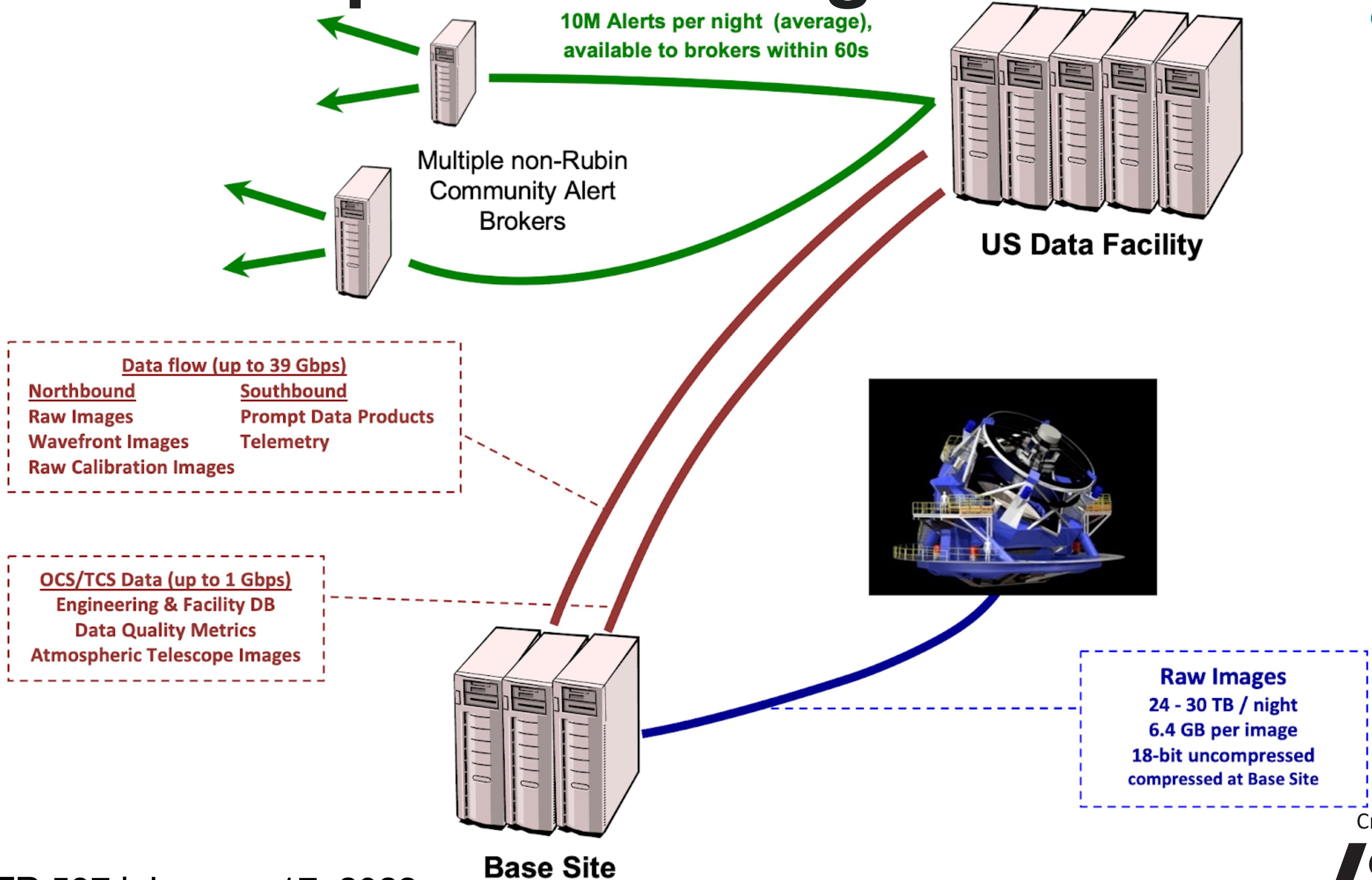
- Principal algorithmic components
 - Known SSO association
 - Moving object association (linking)
 - Orbit fitting
- Data products
 - Solar System object catalog
 - Updated **MPCORB** database



Data flow: Data Release Production



Data flow: Prompt Processing



How to get involved



- This class is a great start!
- Many people in the department working on LSST:
 - Science pipelines developers
 - Ari Heinze
 - Brianna Smart
 - Erin Howard
 - Ian Sullivan
 - John Parejko
 - Krzysztof Findeisen
 - Meredith Rawls
 - Nima Sedaghat
 - Commissioning Scientists
 - Bryce Kalmbach
 - Chris Suberlak
 - Neven Caplar
 - Data Management Scientists
 - Colin Slater
 - Melissa Graham
 - Peter Yoachim
 - Professors
 - Andrew Connolly
 - Eric Bellm
 - Mario Juric
 - Zeljko Ivezic