

# Rubin Observatory

## Algorithms and Pipelines, The Stack, and Tasks

Yusra AlSayyad Oct 14 2020



## What? 150min of Notebooks/Talks today and tomorrow

- **Today PDT**

- 10:00 Intro to Pipelines; What is a Task? (Yusra ALSayyad)
- 10:25 How to write and run a Pipeline and PipelineTask and (Nate Lust)
- 11:10 Developing the Science Pipelines software (Tim Jenness)

- **Tomorrow PDT**

- 8:00 Live Demo. Interacting with Data Products with the Gen3 Butler and debugging a Pipeline (Jim Bosch)

**We know this is not sufficient to get up to speed.**



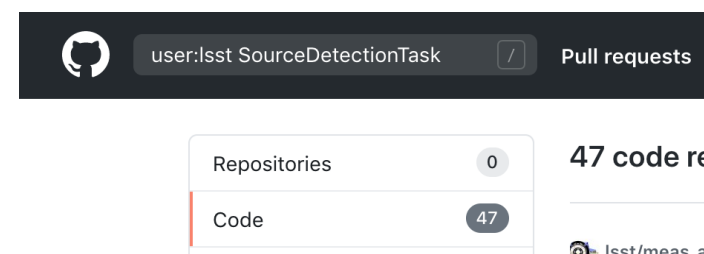
No time for Pair Coding or hands-on projects this week 😞

- If you are currently on the Algorithms and Pipelines team. **Come join us for Pair coding Wednesdays 11am-1pm PDT**
- Learning will continue throughout the year via **code reviews**.
- Get in touch with your team lead on what you need to learn



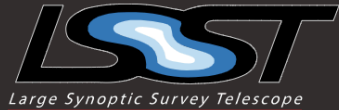
## The basics

- Bootcamp from last year:
  - <https://community.lsst.org/t/dm-boot-camp-2019/3887>
- What you need to start developing:
  - Shared stack: <https://developer.lsst.io/services/software.html>
  - eups <https://developer.lsst.io/stack/eups-tutorial.html>
- Where to ask questions:
  - [How LSST Communicates -- Jonathan Sick](#)
- What are science pipelines plans for astrometry/PSF-estimation etc?
  - Algorithms workshop videos/slides present understanding of state of the art and plans as of March 2020: [lsst.law](https://lsst.law)
- How do I do X in the codebase? (assuming you've already consulted [pipelines.lsst.io](https://pipelines.lsst.io))
  - Search Slack, Github, Ask
- What is the procedure for... ?
  - [developer.lsst.io](https://developer.lsst.io)
- How do I play nice with others?



[developer.lsst.io/work/flow.html](https://developer.lsst.io/work/flow.html)

Rubin  
Observatory



## LSST DM Developer Guide

Edition: Current

Change edition

### TEAM

Onboarding Checklist

Team Culture and Conduct Standards

Empowerment of DM team members

Data Release Production

### COMMUNICATIONS

Configuring your GitHub username in your Slack profile

[Docs](#) » DM Development Workflow with Git, GitHub, JIRA and Jenkins

[Edit on GitHub](#)

# DM Development Workflow with Git, GitHub, JIRA and Jenkins

This page describes our procedures for collaborating on LSST DM software and documentation with [Git](#), [GitHub](#) and [JIRA](#):

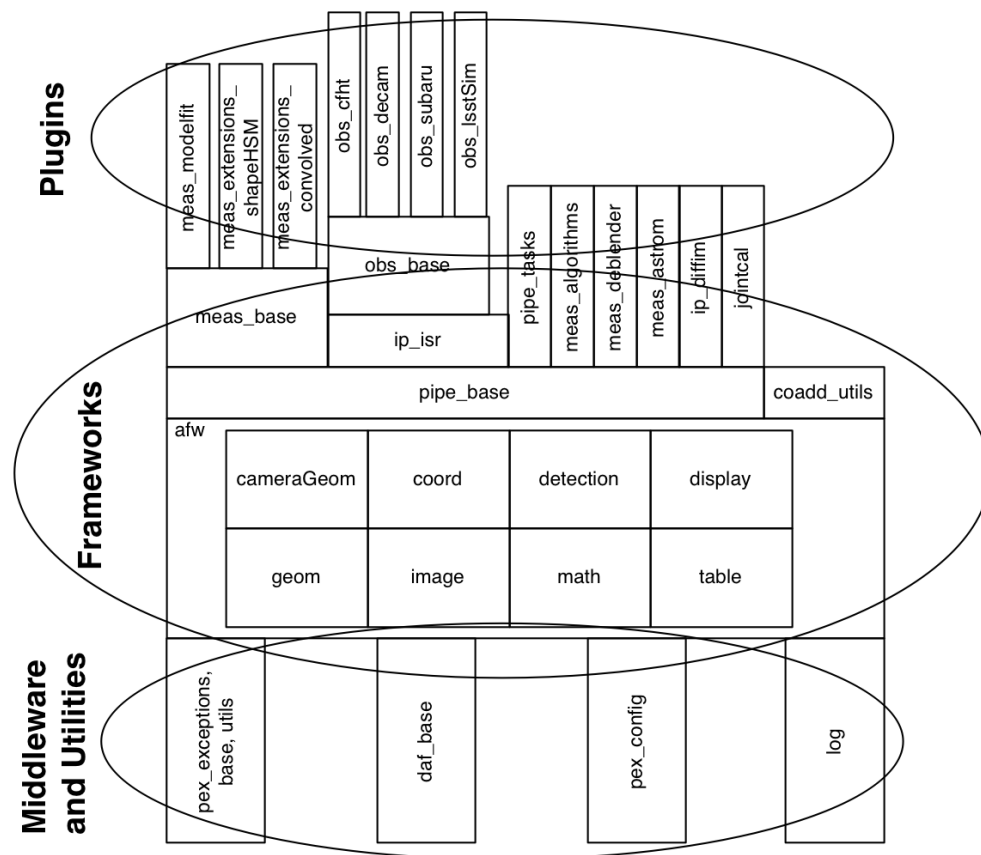
1. [Configuring Git for DM development.](#)
2. [Using JIRA for agile development.](#)
3. [DM GitHub organizations.](#)
4. [Policies for naming and using Git branches.](#)
5. [Preparing code for review.](#)
6. [Reviewing and merging code.](#)

In appendices, we suggest some *best practices* for maximizing the usefulness of our Git development history:

- [Commit organization best practices.](#)
- [Commit message best practices.](#)

# The Stack: [github.com/lsst](https://github.com/lsst)

Docs: [pipelines.lsst.io](https://pipelines.lsst.io)



**Value: The whole stack is owned by the whole team**

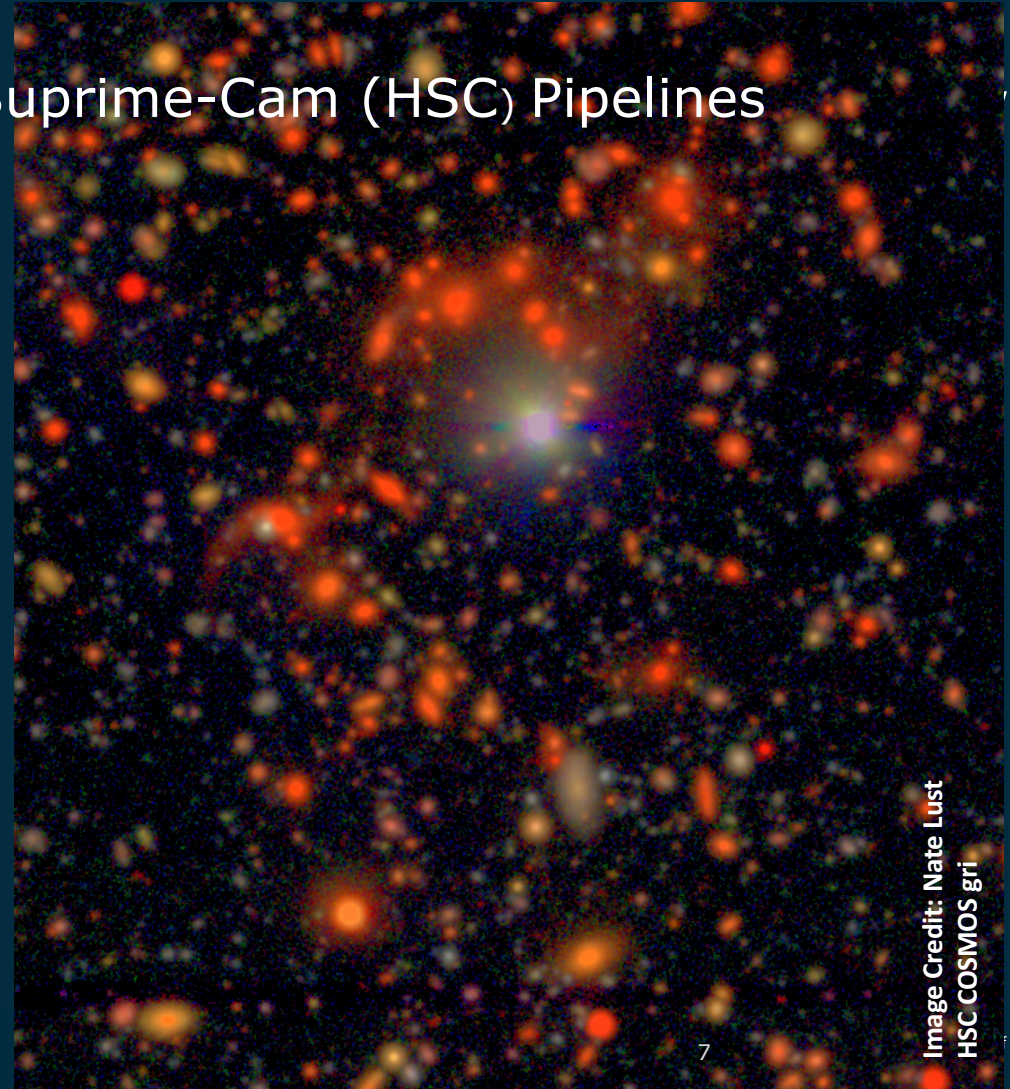
If you see something you don't like: File a ticket and fix it. If its a change of existing behavior, file an RFC and fix it.

This stack is your stack



We say “HSC” frequently because the LSST Pipelines **are** the Hyper Suprime-Cam (HSC) Pipelines

Survey Comparison	LSST	HSC (Subaru Strategic Program)
Effective Aperture	6.5m	8.2m
Filters	ugrizy	grizy + narrow
Exposure time per visit	~30s	~240s
Field of View	10 deg <sup>2</sup> 3.5 deg diam	1.8 deg <sup>2</sup> 1.5 deg diam
Num CCDs	189 (4k x 4k)	103 (4k x 2k)



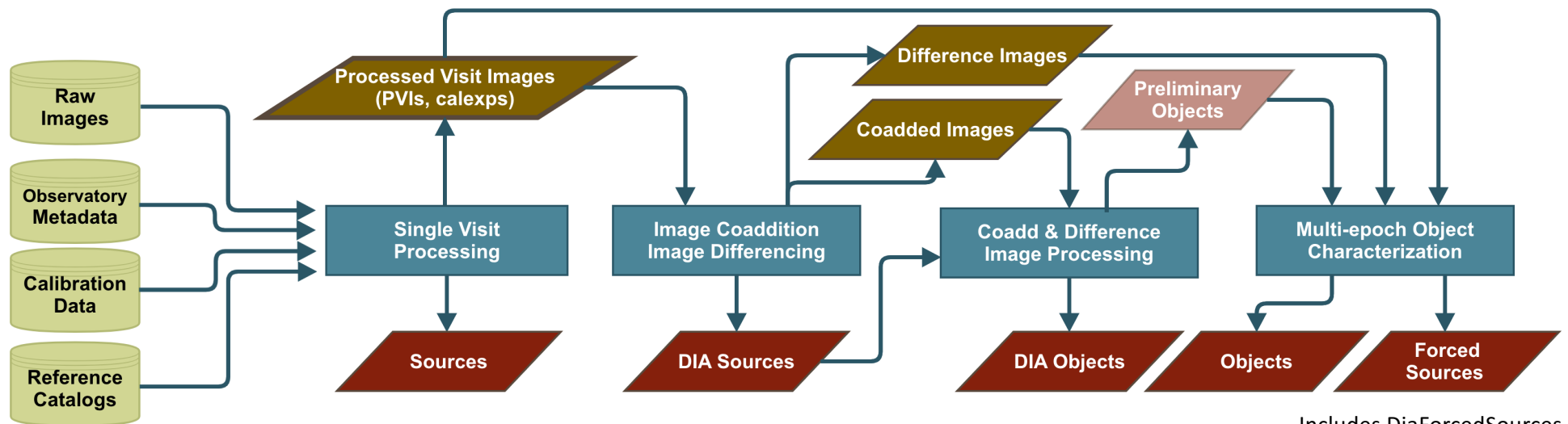
The DRP and AP pipelines are constructed from the same algorithmic components

**Rubin**  
Observatory

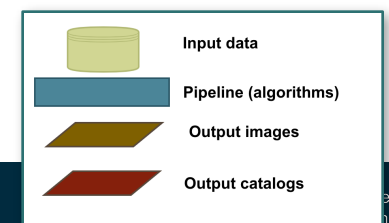




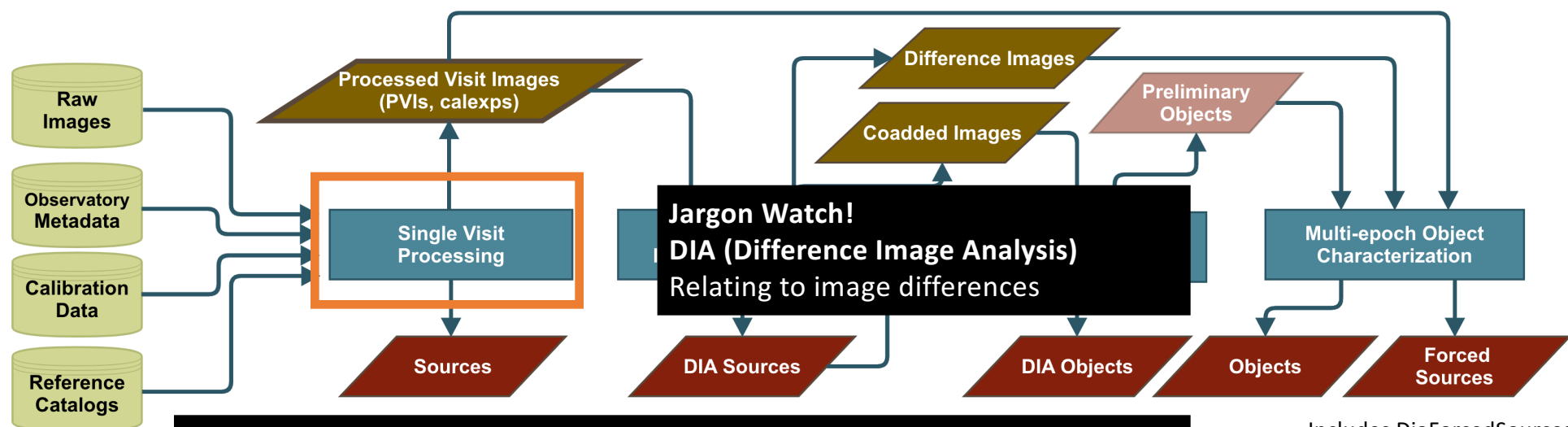
# High level overview of the a Data Release Production



Includes DiaForcedSources



# High level overview of the a Data Release Production

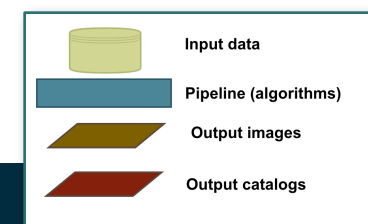


## Jargon Watch!

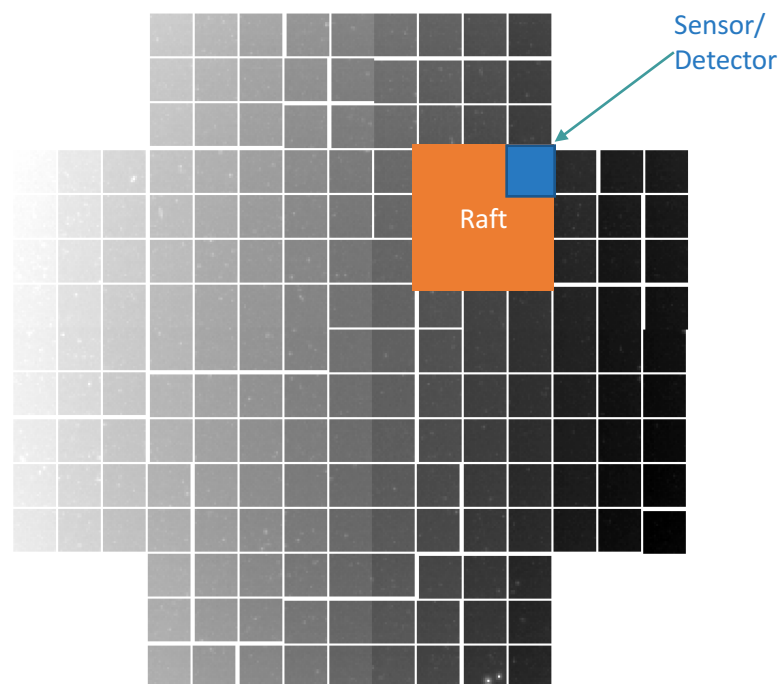
**Sources:** Measurements from a single observation

**Objects:** Measurements utilizing many (typically all) observations. The logical aggregate (incl. motion, color, variability, deep). **Not matched Sources!**

Includes DiaForcedSources

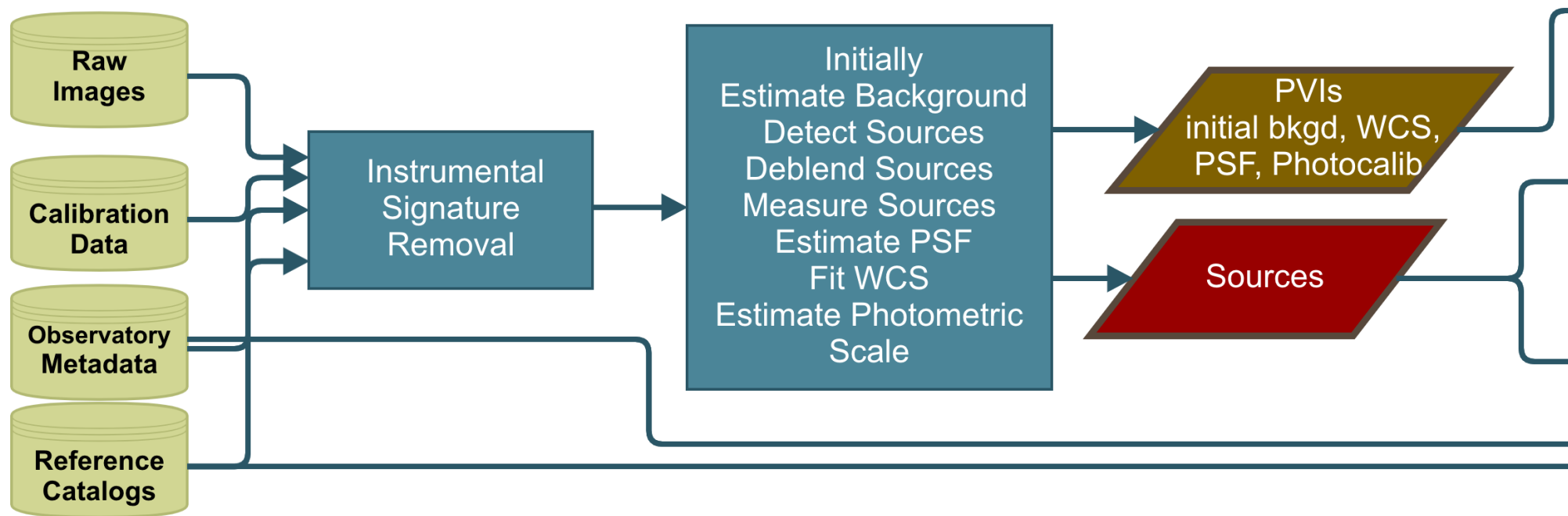


## Jargon watch: Visits, CCDs, Exposures

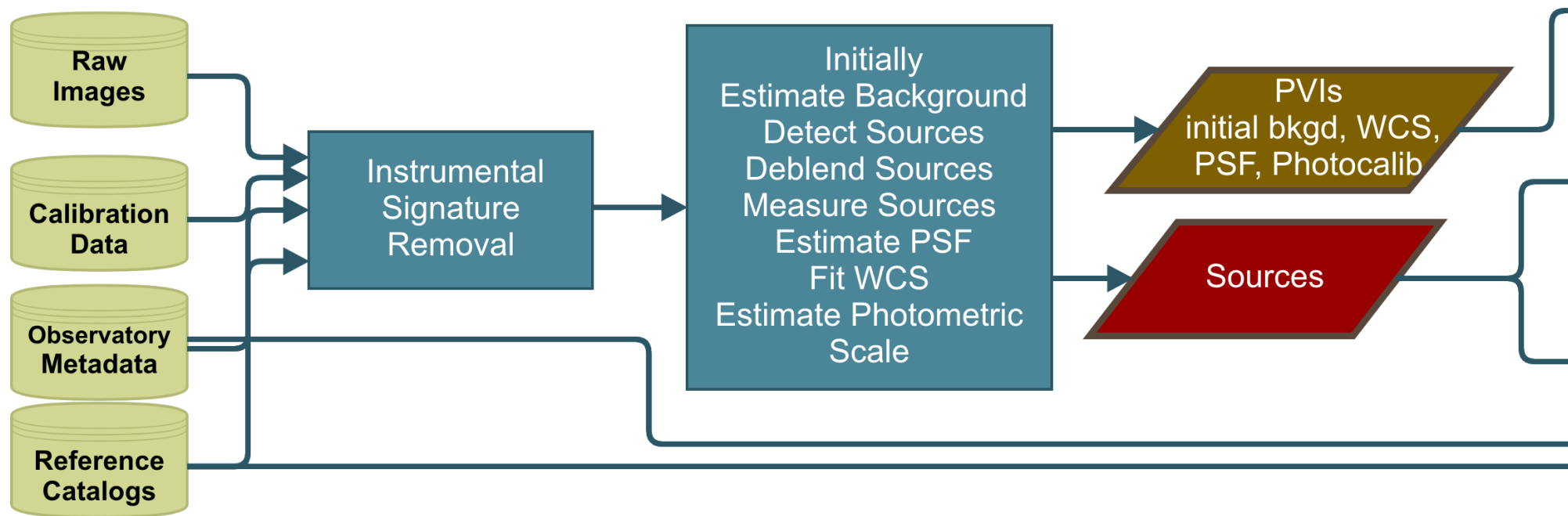


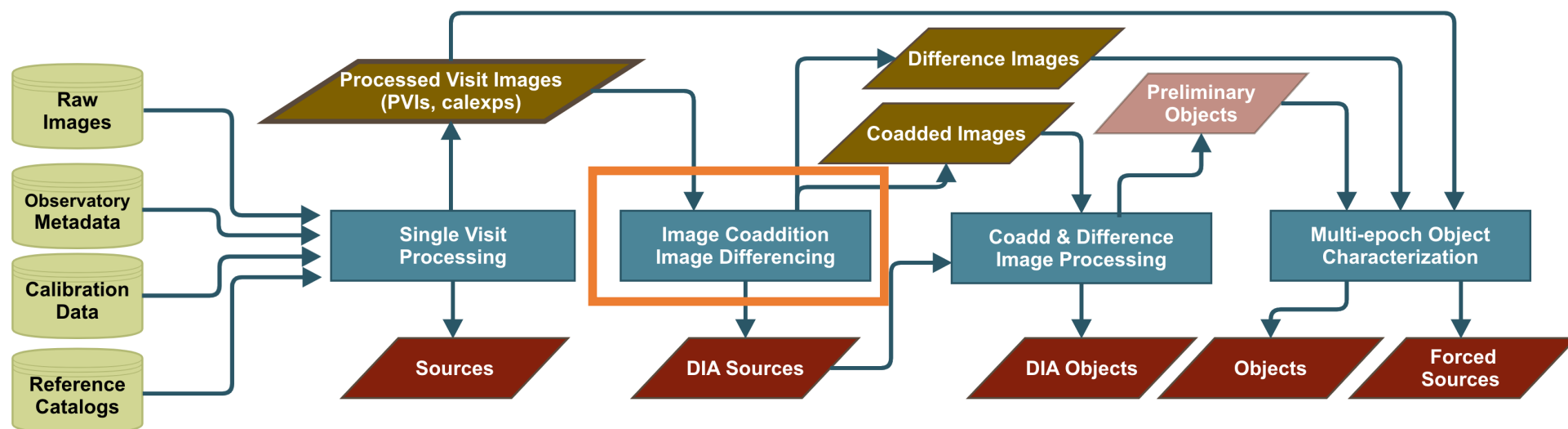
**Exposure:** A software Object that contains an image plane, mask plane, variance plane, PSF model, WCS, photoCalib and visit metadata. Stored in FITS format

## Single Visit Processing: IsrTask, CharImageTask, CalibrateTask ...



## Single Visit Processing: IsrTask, CharImageTask, CalibrateTask ...





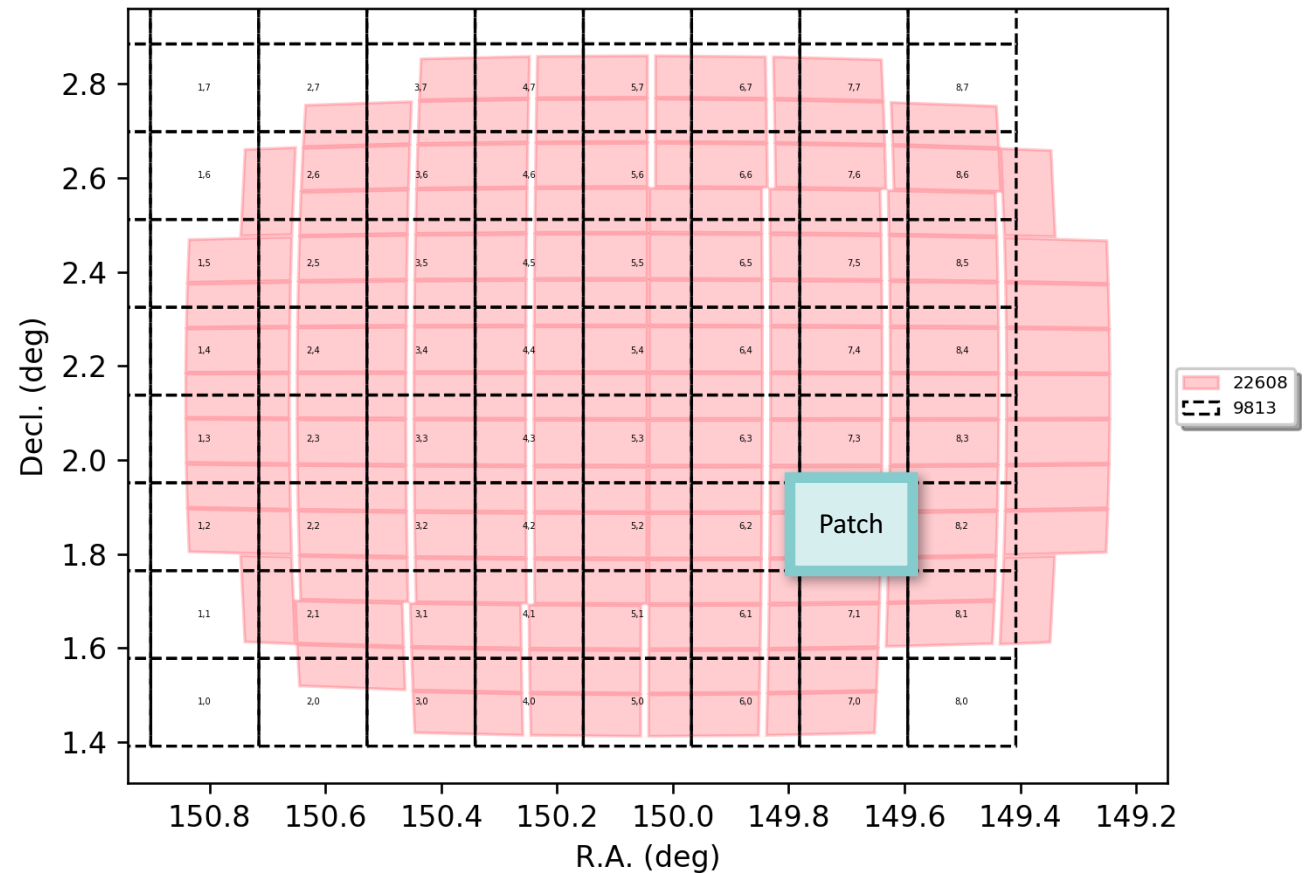


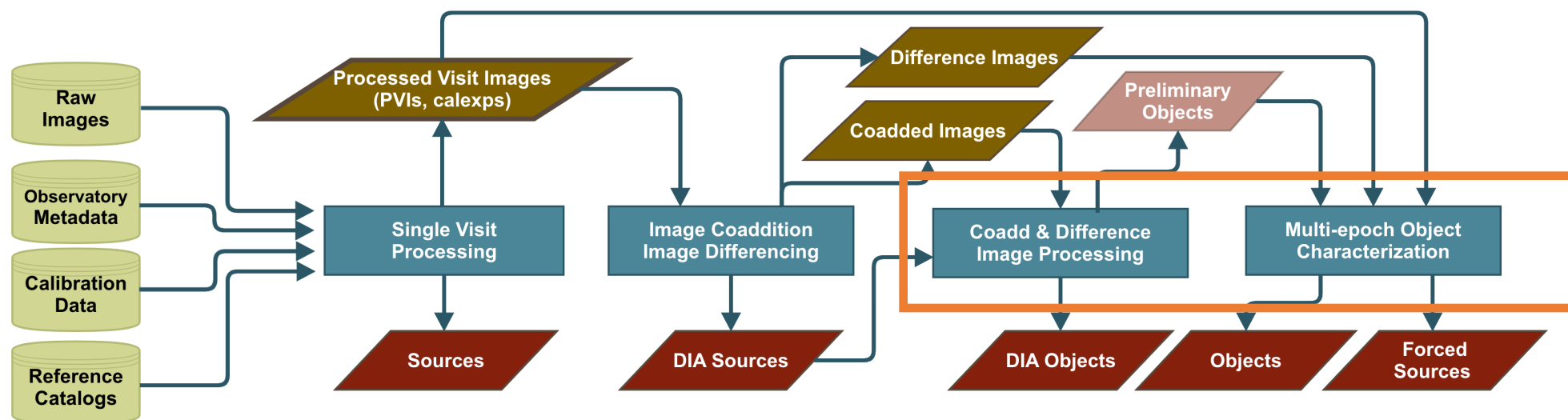
## Jargon Watch! Tract, Patch HSC's skyMap:

SkyMap: a Software Object that defines a coadd's:

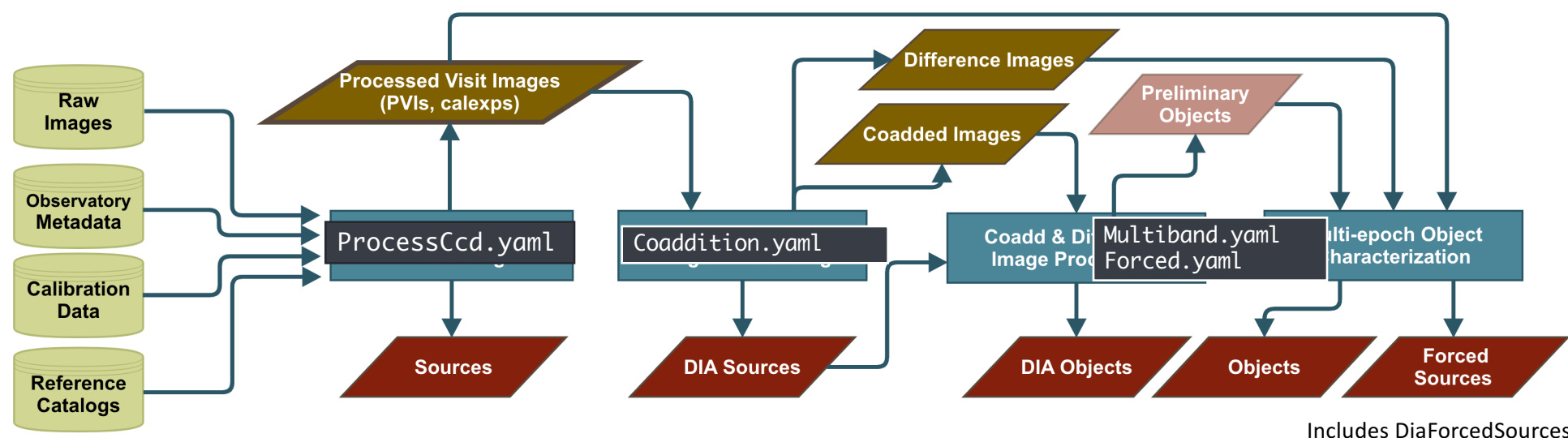
- WCS/Projection: TAN (gnomonic)
- Pixel Scale: ~Native
- Tract Size: ~FOV
- Patch Size: ~CCD

Modular implementation makes it easy to swap projections and tessellations at runtime





# These abstract processing steps map to Pipelines



## Tasks:

[github.com/lsst-dm/Nov19\\_bootcamp/blob/master/notebooks/HowToWriteATask.ipynb](https://github.com/lsst-dm/Nov19_bootcamp/blob/master/notebooks/HowToWriteATask.ipynb)

TL;DR If you take one thing away from this talk, it is go to <http://pipelines.lsst.io>, then click on [lsst.pipe.base](#).

### What is a Task?

Tasks implement astronomical data processing functionality. They are:

- **Configurable:** Modify a task's behavior by changing its configuration. Automatically apply camera-specific modifications
- **Hierarchical:** Tasks can call other tasks as subtasks
- **Extensible:** Replace ("retarget") any subtask with a variant. Write your own subclass of a task.

1. <https://pipelines.lsst.io/modules/lsst.pipe.base/task-framework-overview.html>
2. <https://pipelines.lsst.io/modules/lsst.pipe.base/creating-a-task.html>



## Tasks:

[github.com/lsst-dm/Nov19\\_bootcamp/blob/master/notebooks/HowToWriteATask.ipynb](https://github.com/lsst-dm/Nov19_bootcamp/blob/master/notebooks/HowToWriteATask.ipynb)

**Rubin**  
Observatory

The screenshot shows a web browser window with the URL `pipelines.lsst.io/#python-modules`. The page is titled "LSST Science Pipelines" and has a "Forum" link in the top right. The main content area is for the `lsst.pipe.base` module. On the left, there is a sidebar with a search bar and a list of links under the heading "On this page". The main content area has the heading `lsst.pipe.base` and a paragraph describing the module. Below this, there is a section titled "Using lsst.pipe.base" with two sub-sections: "Overview" and "Using command-line tasks".

**LSST Science Pipelines** Forum

Search

**On this page**

- [lsst.pipe.base](#)
- [Using lsst.pipe.base](#)
- [Overview](#)
- [Using command-line tasks](#)
- [Developing tasks and command-line tasks](#)
- [Contributing](#)
- [Python API reference](#)
- [lsst.pipe.base Package](#)
- [Functions](#)
- [Classes](#)
- [Class Inheritance Diagram](#)
- [lsst.pipe.base.testUtils Module](#)
- [Functions](#)

**lsst.pipe.base**

The `lsst.pipe.base` module provides base classes for the task framework. Tasks package the algorithmic units of the LSST Science Pipelines. You can create, configure, and run tasks with their Python APIs. Some tasks, called command-line tasks, are also packaged into data processing pipelines that you can run from the command line.

**Using lsst.pipe.base**

**Overview**

- [Overview of the task framework](#)

**Using command-line tasks**

- [Using Butler data repositories and reruns with command-line tasks](#)
- [Specifying data IDs with command-line tasks](#)

## Tasks:

[github.com/lsst-dm/Nov19\\_bootcamp/blob/master/notebooks/HowToWriteATask.ipynb](https://github.com/lsst-dm/Nov19_bootcamp/blob/master/notebooks/HowToWriteATask.ipynb)

---

**TL;DR** If you take one thing away from this talk, it is go to <http://pipelines.lsst.io>, then click on [lsst.pipe.base](#).

---

On the landing page for `lsst.pipe.base` documentation <https://pipelines.lsst.io/modules/lsst.pipe.base/index.html>, you'll see a number of tutorials on how to use Tasks and how to create one.

`CmdlineTask` extends `Task` with commandline driver utils for use with Gen2 Butlers, and will be deprecated soon. However, not all the links under "CommandlineTask" will become obsolete. For example, [Retargeting subtasks of command-line tasks](#) will live on.

prerequisites for understanding the `PipelineTask`:

1. <https://pipelines.lsst.io/modules/lsst.pipe.base/task-framework-overview.html>
2. <https://pipelines.lsst.io/modules/lsst.pipe.base/creating-a-task.html>

