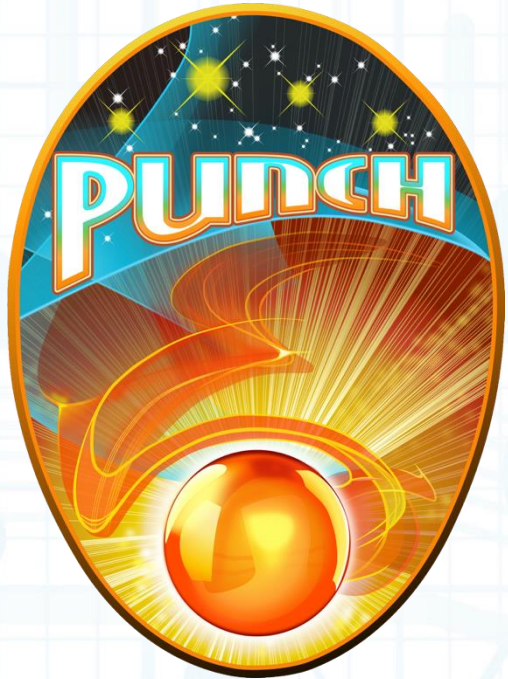




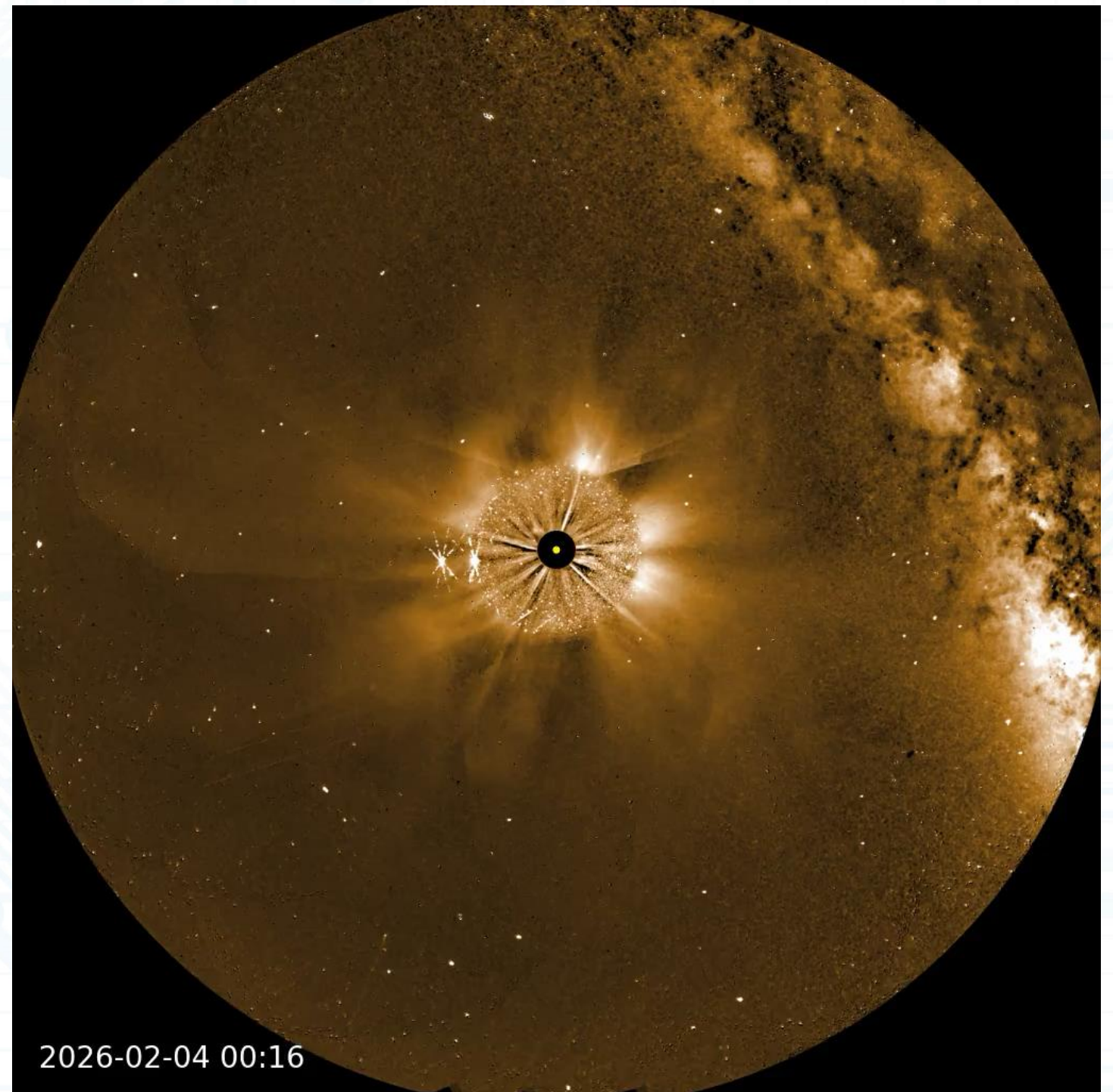
# Data System Reusability for the PUNCH Mission

SOUTHWEST RESEARCH INSTITUTE®

Marcus Hughes, on behalf of the PUNCH Science Operations Center  
DASH Conference, October 6, 2026

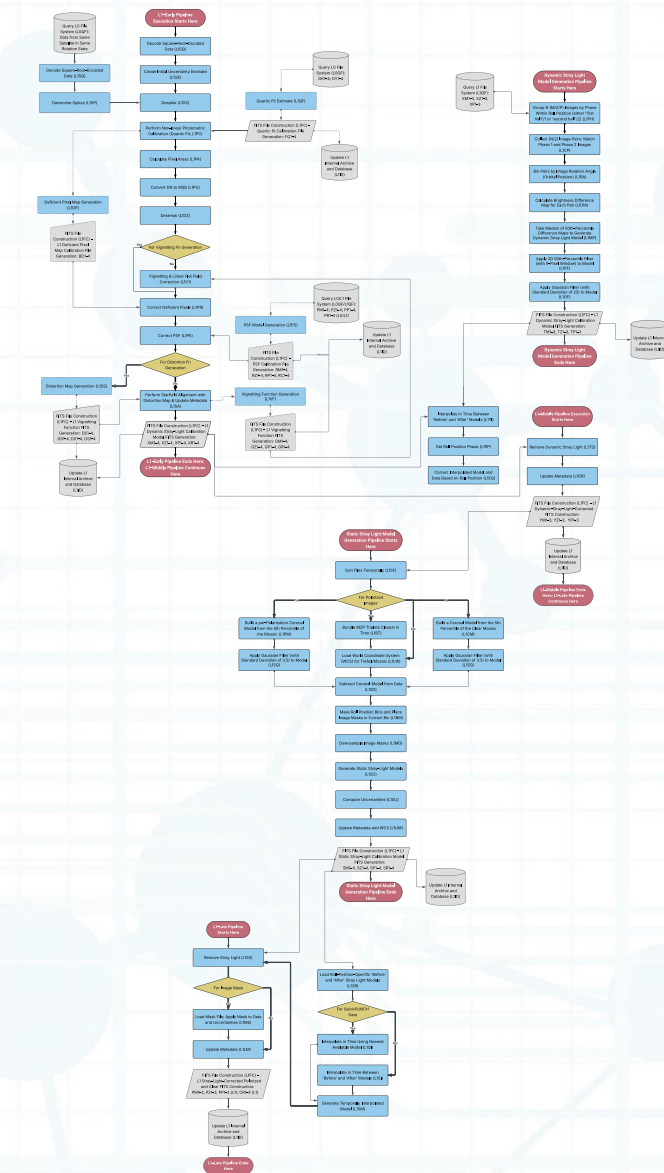
# What is PUNCH?

- **Polarimeter to UNify the Corona and Heliopshere**
- PUNCH is a NASA Small Explorers Mission, launched March 11, 2025.
- A constellation of 4 spacecraft that create a single virtual instrument.
  - 3 Wide Field Imagers (WFI)
  - 1 Narrow Field Imager (NFI)
- Goal: Measure how the corona transitions to become the solar wind throughout the heliosphere.



# Overview

- Both MOC and SOC are at SwRI Boulder.
- Computing is done for both using on premise machines.
- Science code is developed primarily internally in the SOC with communication with scientists.
- Well over 100 data products.
- Very complex processing required.



High level steps of one level of processing!

# Parts of a Data System

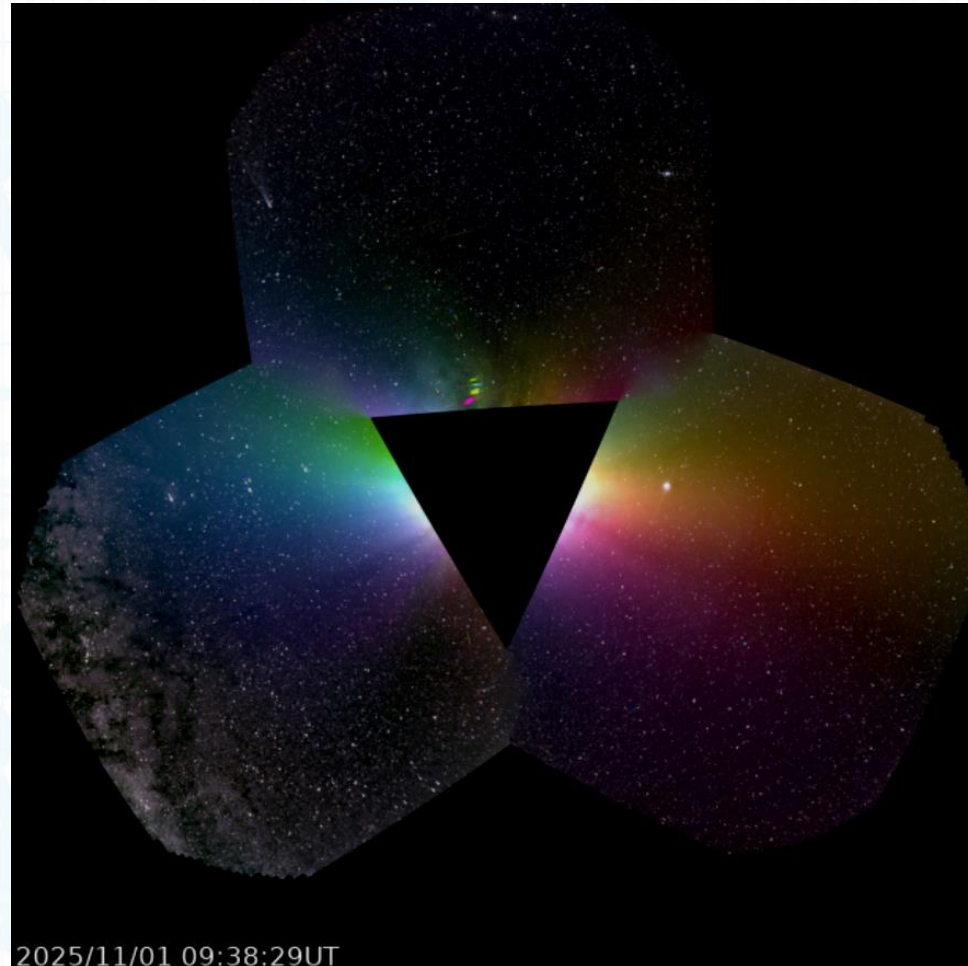
- Architecture
  - Overall design of the SDS.
- Data Products
  - Standardized formats for outputs.
- Computing Infrastructure
  - On prem vs cloud infrastructures.
- Components
  - Science processing components being open sourced.
- Data Flow
  - Telemetry, ingest, automation, versioning schemes, archiving.
- Development
  - How do you work? Team management. Testing strategies.

Borrowed from  
**Mission Science Data Systems  
(A Mission-Specific Template)**  
<https://zenodo.org/records/17316944>

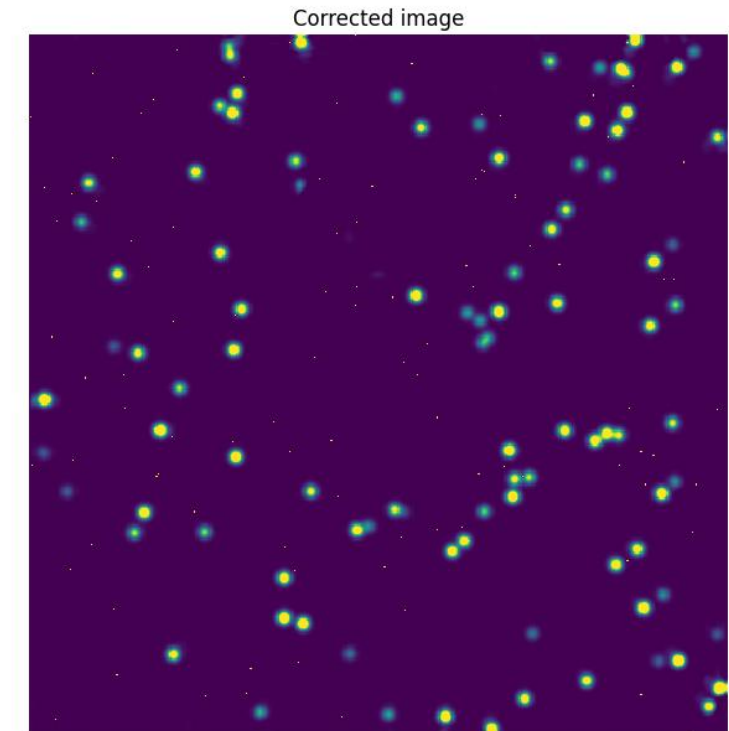
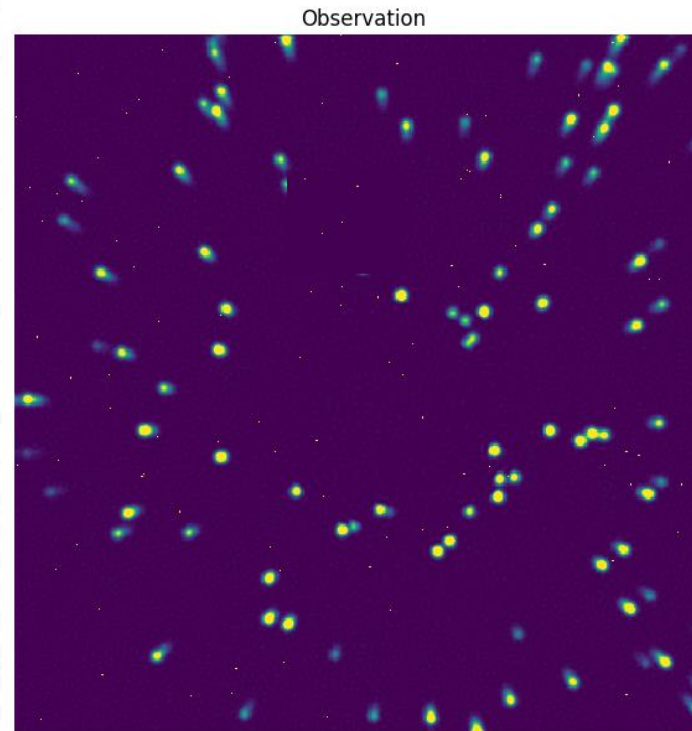
# What Makes a Part Reusable?

- Instrument agnostic design
  - Implementations without hard-coded parameters or features that are hyper-specific to a certain mission.
- Clear documentation
  - Without documentation, no one else can utilize.
- Ease of deployment
  - If it takes longer to use for your purpose than do it from scratch, it won't be used.
- Standard formats
  - Inventing your own format creates barriers.
- Maintenance
  - No one wants to use a tool that will deprecate in a year.

# Opportunities for Reuse for PUNCH: Standalone Packages



***solpolpy***: convert between  
polarization systems



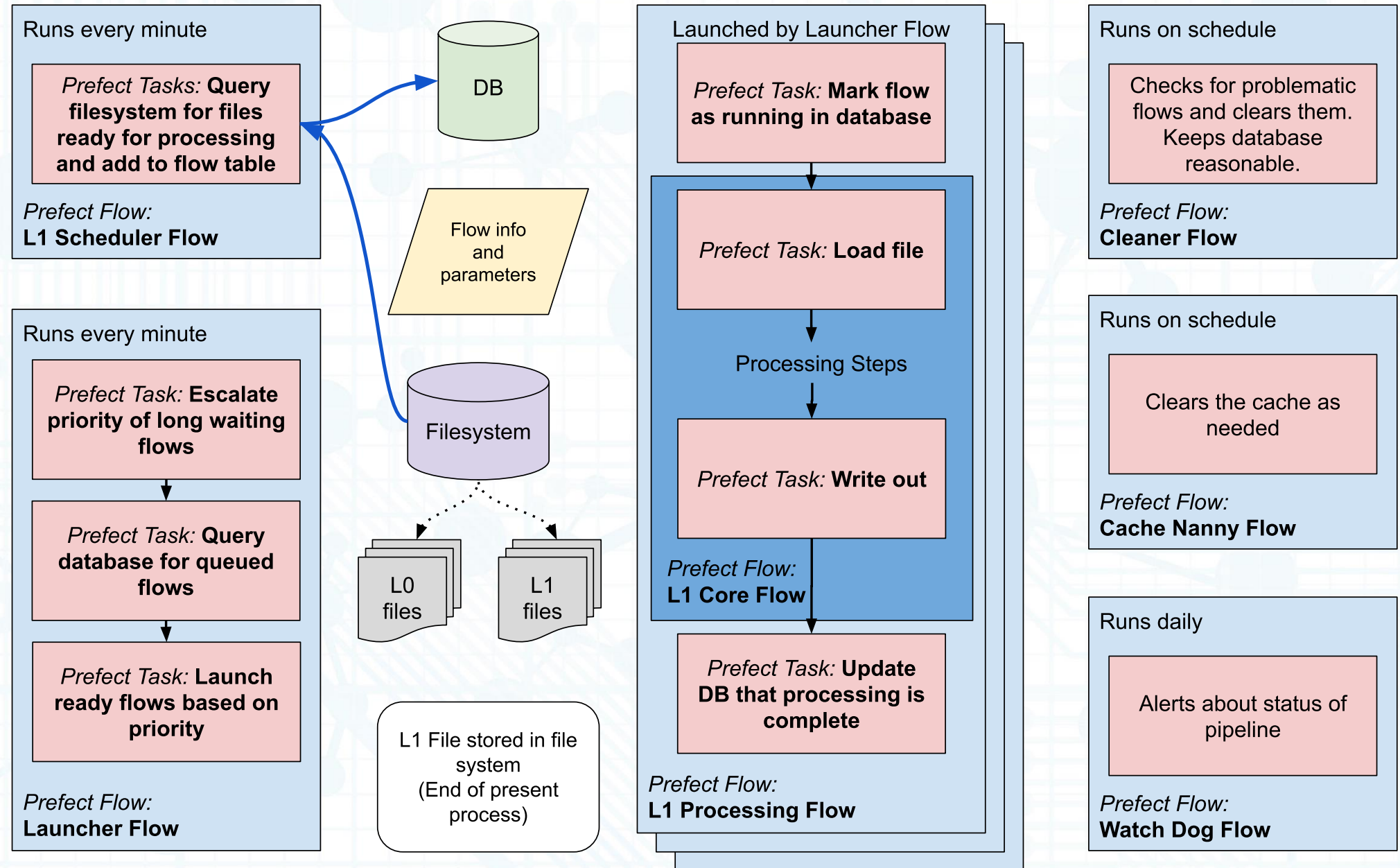
***regularizepsf***: make your point  
spread functions uniform

# Opportunities for Reuse for PUNCH: Algorithms

- PUNCH code is all open sourced.
  - Reusability lies on a spectrum.
- 
- Some solutions are PUNCH-specific:
    - Image despiking
    - Some of our stray light modeling
  - Some solutions are generic:
    - Square root decoding
    - Efficient and parallelized NaNPercentile method for image cubes
    - NaN-aware Gaussian Filter
    - Shared Memory NDArrays

# Opportunities for Reuse for PUNCH: Automation

- Incorporates Prefect.
- Due to a complex file dependency, we developed a flexible “scheduler” and “launcher” framework.
- We have two data streams, science and QuickPUNCH, that need different prioritizations.



# Opportunities for Reuse for PUNCH: Development Strategy

- Simulation driven development
  - Make simulated data prior to launch
  - Test on imperfect simulations
- Document processes
  - Record how to do standard procedures
  - Have a clear onboarding and offboarding protocol
- Team dynamics
  - Encourage openness and teamwork with a positive culture
  - Retrospectives with portions for management-included and team-only

# Friction Points for Reuse

- Mission funding is provided for implementing your mission's SDS, not for a generic system.
- Developing standalone packages makes it harder to test code changes directly in the pipeline.
- Documentation *cannot* be deferred. Be sure to do it as you go.
- Stopping to contribute back to community packages you don't control can slow integration and development.

# Summary

- Striving towards reuse isn't easy but helps the entire community.
- There are degrees of reusability.
  - A direct import versus a design pattern to follow.
- PUNCH has directly reusable packages: solpolpy and regularizepsf.
- punchbowl has many reusable components: square root decoding, utility functions for Numba operations.
- PUNCH leverages community packages: Astropy, Reproject, SunPy, NDCube, Prefect.
  - Some solutions have been contributed back into the community packages.
  - There's room for more feedback.

