STOA – Script Tracking for Observational Astronomy

Peter Hague - University of Cambridge

UNIVERSITY OF CAMBRIDGE

Background

- **BaSC** and other projects required batch operations on ALMA archive

- **STOA** emerged from my efforts to streamline and automate these operations

- Now a standalone web application: https://github.com/petehague/stoa

Objectives

- Want to perform actions in batches
- Each instance of an action may need to run differently
- Must track provenance of result – ensure we know precisely which version of each script led to a specific output
- Must be able to share data, processes and outputs
- Final results should be easily accessible to others

Example task

- Want to run SExtractor on every observation that matches a certain criteria

- Ideally, archive should contain clean image, primary beam correction, and the corrected image

- One or more may not be present. May also not be immediately clear from metadata which is which, so algorithm has to guess

- Sometimes guesses wrong…
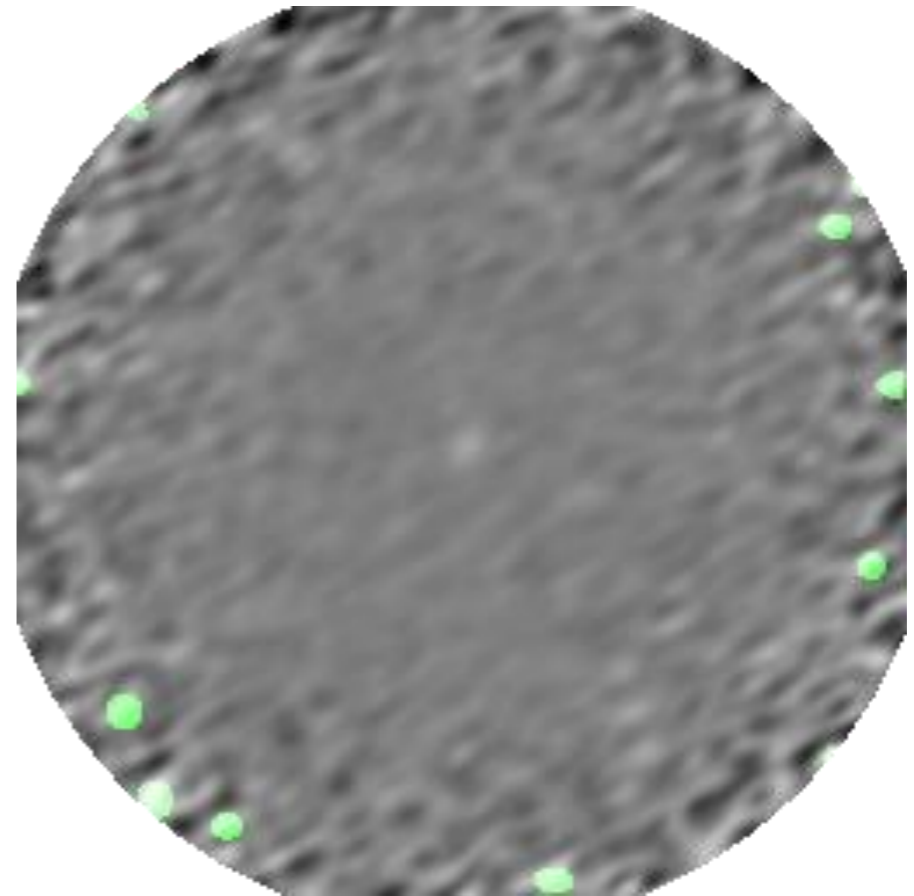
Workflows
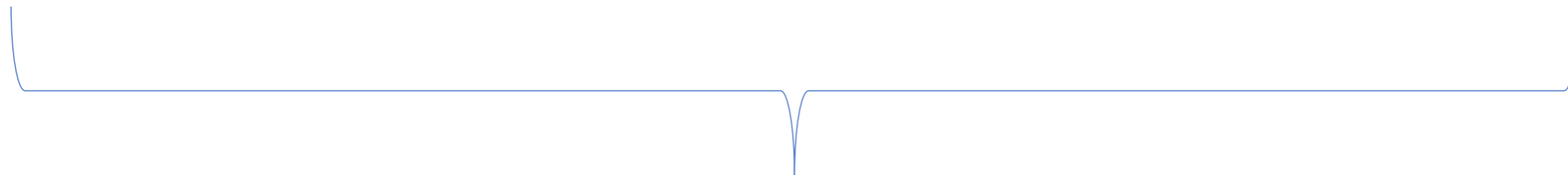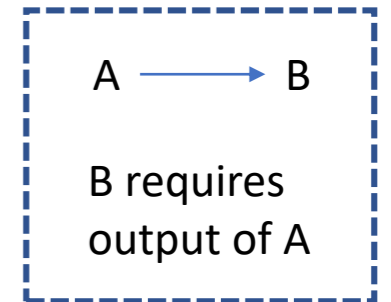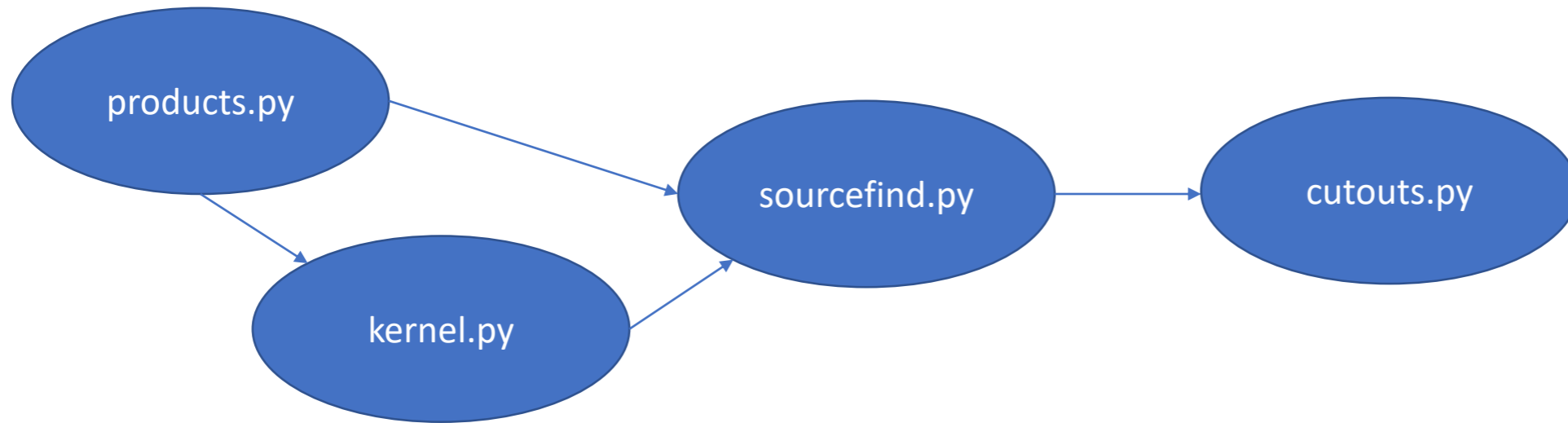
Worktables

- Each row is an execution of the workflow

- Writable inputs, read-only outputs

- Software tracks status (e.g. if currently shown outputs correspond to inputs or if new outputs are pending)

- Worktables connected in relational style; automatically triggering recomputation

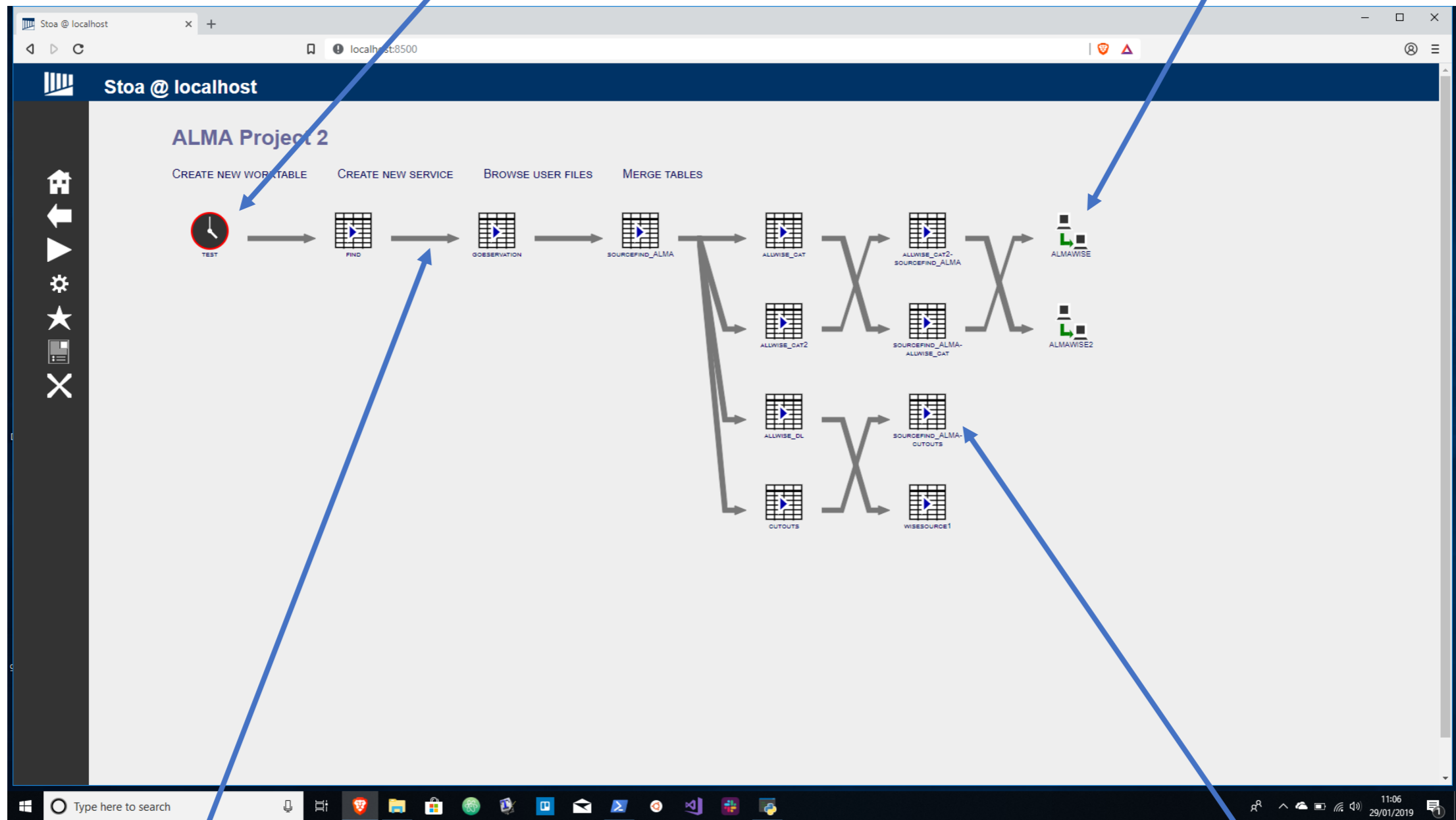| *	Inputs	Outputs

* Hidden fields – tracking processing status etc.

Worktables

- **STOA** allows access to a worktable as Owner, Collaborator, and Reader.
- Owner can initiate computation, collaborator can flag rows and add comments, and reader sees worktable as a simple read only table
- Worktables can be served up as *.fits* downloads or through VO cone search protocol.
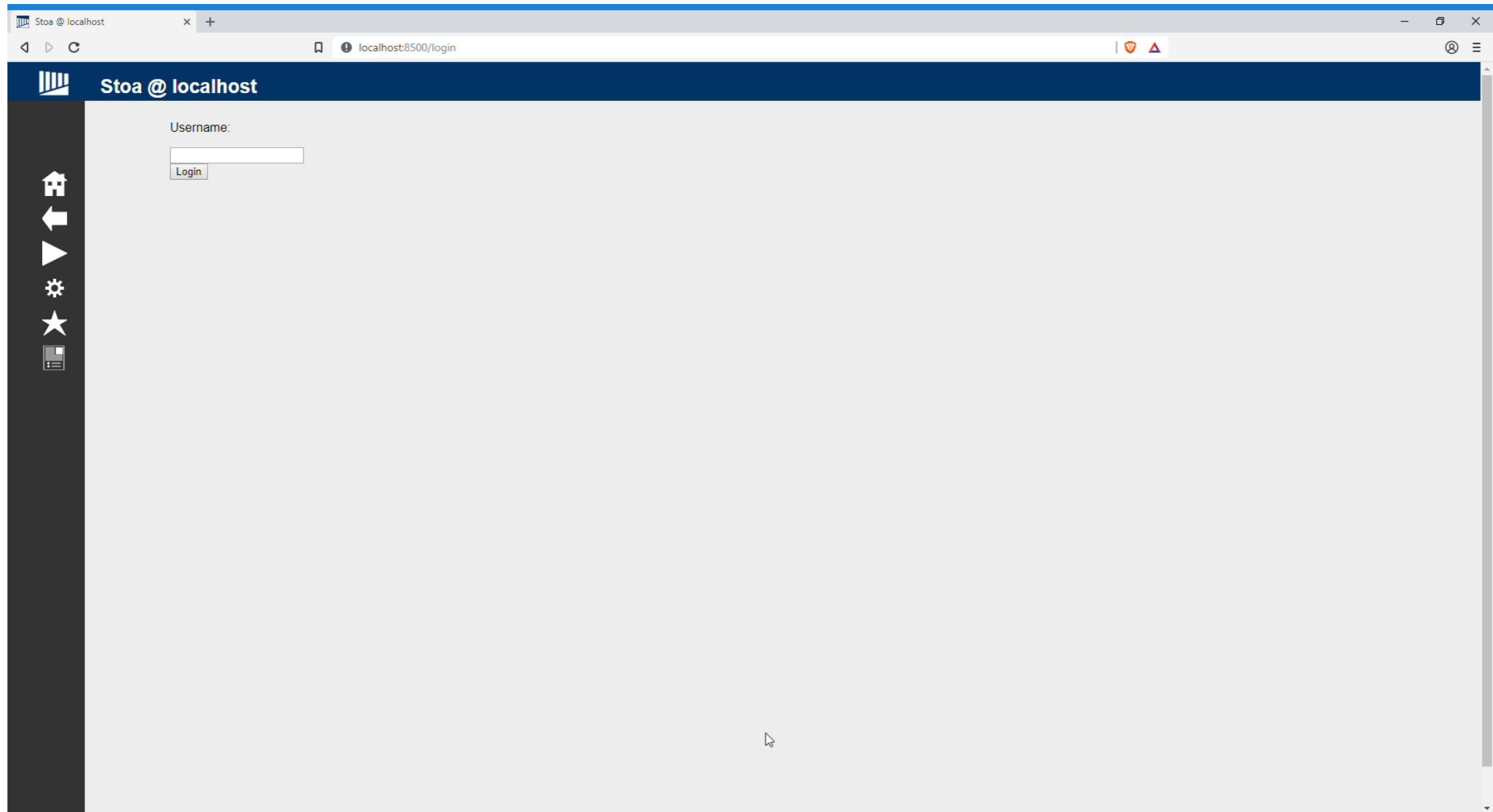
Example

Time trigger

Download/VO Service

Join Operation

Using output of a table as key for another

Example

Example

```
1   #!/usr/bin/env cwl-runner
2
3   cwlVersion: v1.0
4   class: CommandLineTool
5   baseCommand: ["python", "grabcoords.py"]
6
7
8
9   inputs:
10    pathname:
11      type: string
12      inputBinding:
13        position: 1
14
15  outputs:
16   ra:
17     type:
18       type: array
19       items: double
20    dec:
21     type:
22       type: array
23       items: double
```

Example

CWL allows much more complex workflows, invoking multiple steps
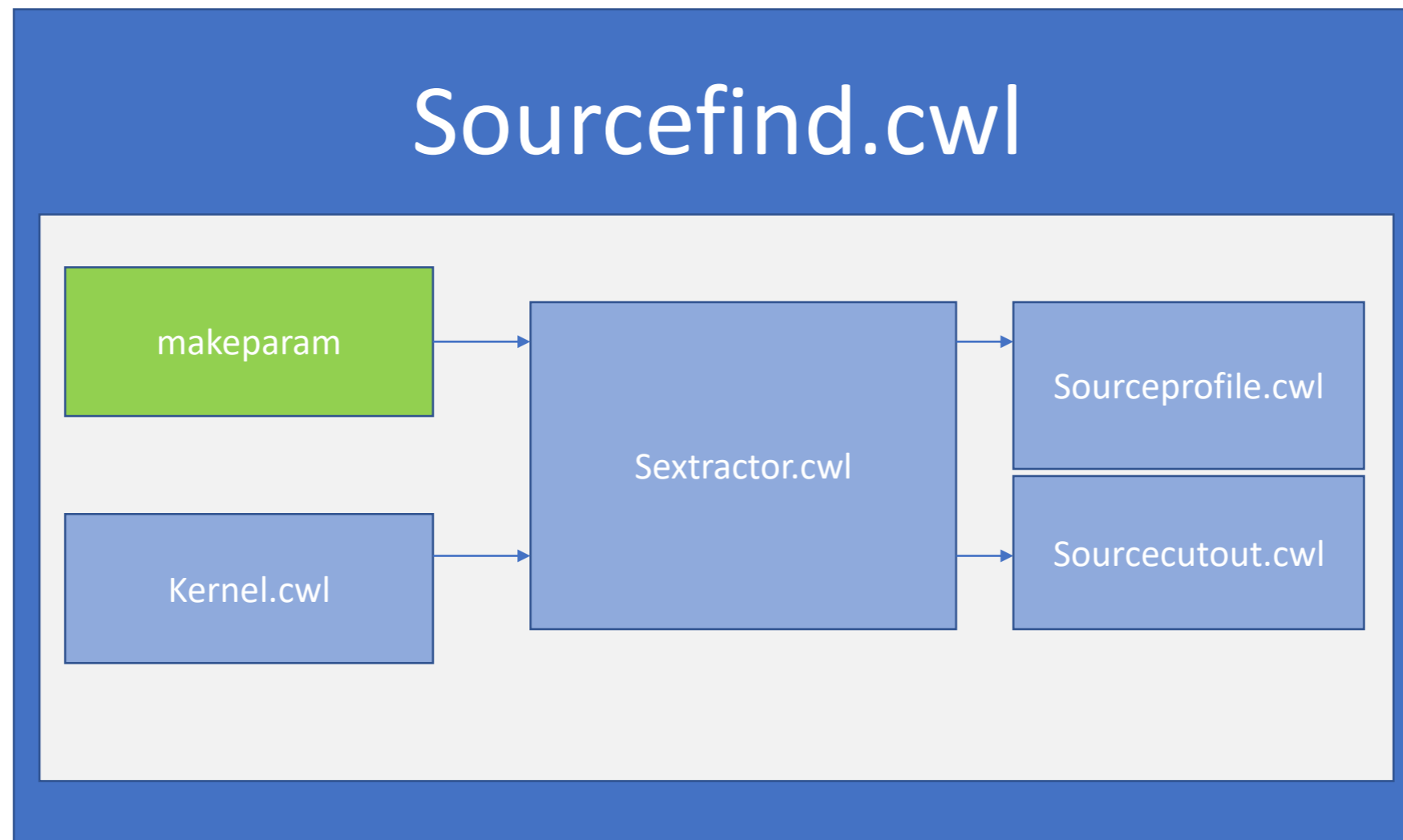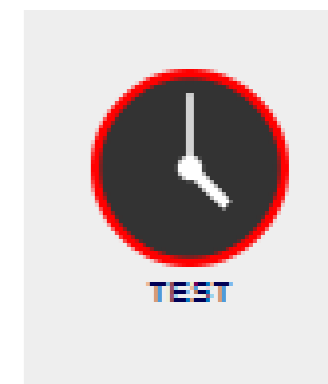
Templates/Built in actions

Basic operations (e.g. bash 'find')
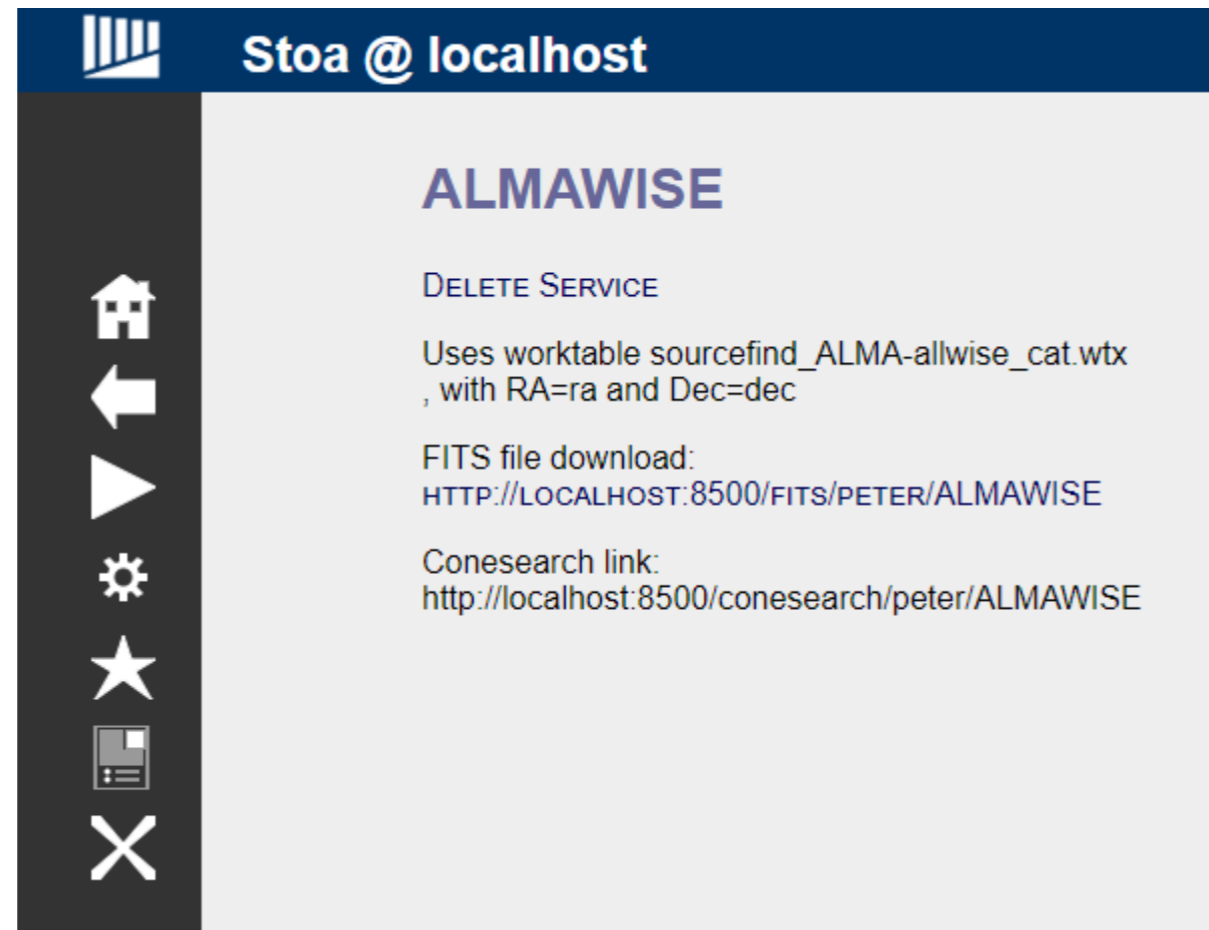

Access to services

Time trigger – initiate computation at regular intervals

Public table server

Example

Current public table interface

The Future

- Authentication – can't have public STOA services without it!

- Figure out how STOA fits in with current services/registries

- Expand the action library

- Help users to build worktables more easily

- Look at more integration with other software – specifically for provenance and minimal recomputation

Summary

https://www.github.com/petehague/STOA

Available under APACHE license

Contact: prh44@cam.ac.uk