

CWL-izing your pipeline

Matt Wyczalkowski
m.wyczalkowski@wustl.edu



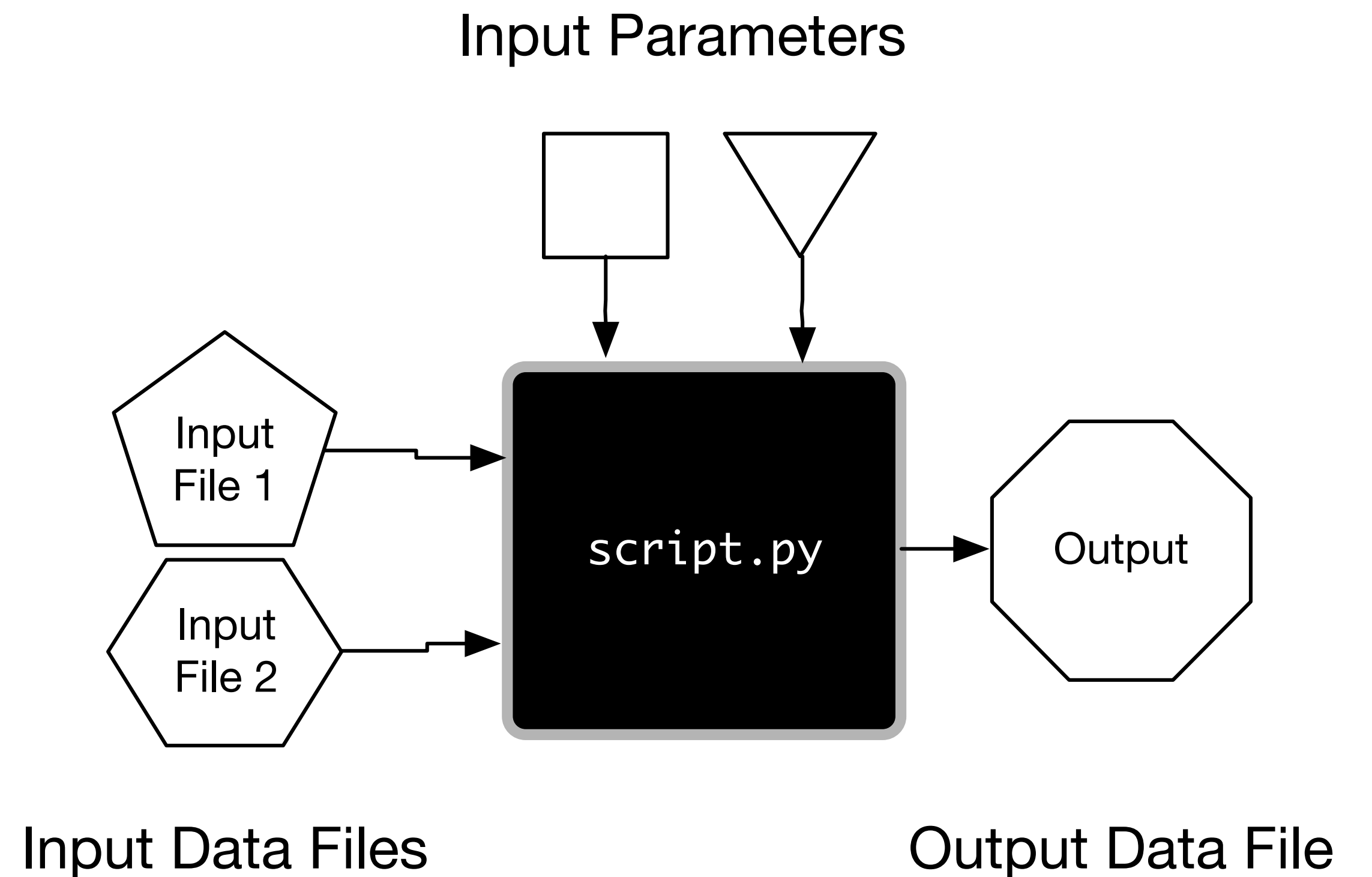
Why CWL

- Allow your tool or pipeline to be run easily on many systems
 - Laptop, Denali, MGI, CGC, Google Cloud
- Share, version workflows
 - Reproducibility

Four stages of development

- Core tool development
- Dockerization
- CWL Tool
- CWL Workflow

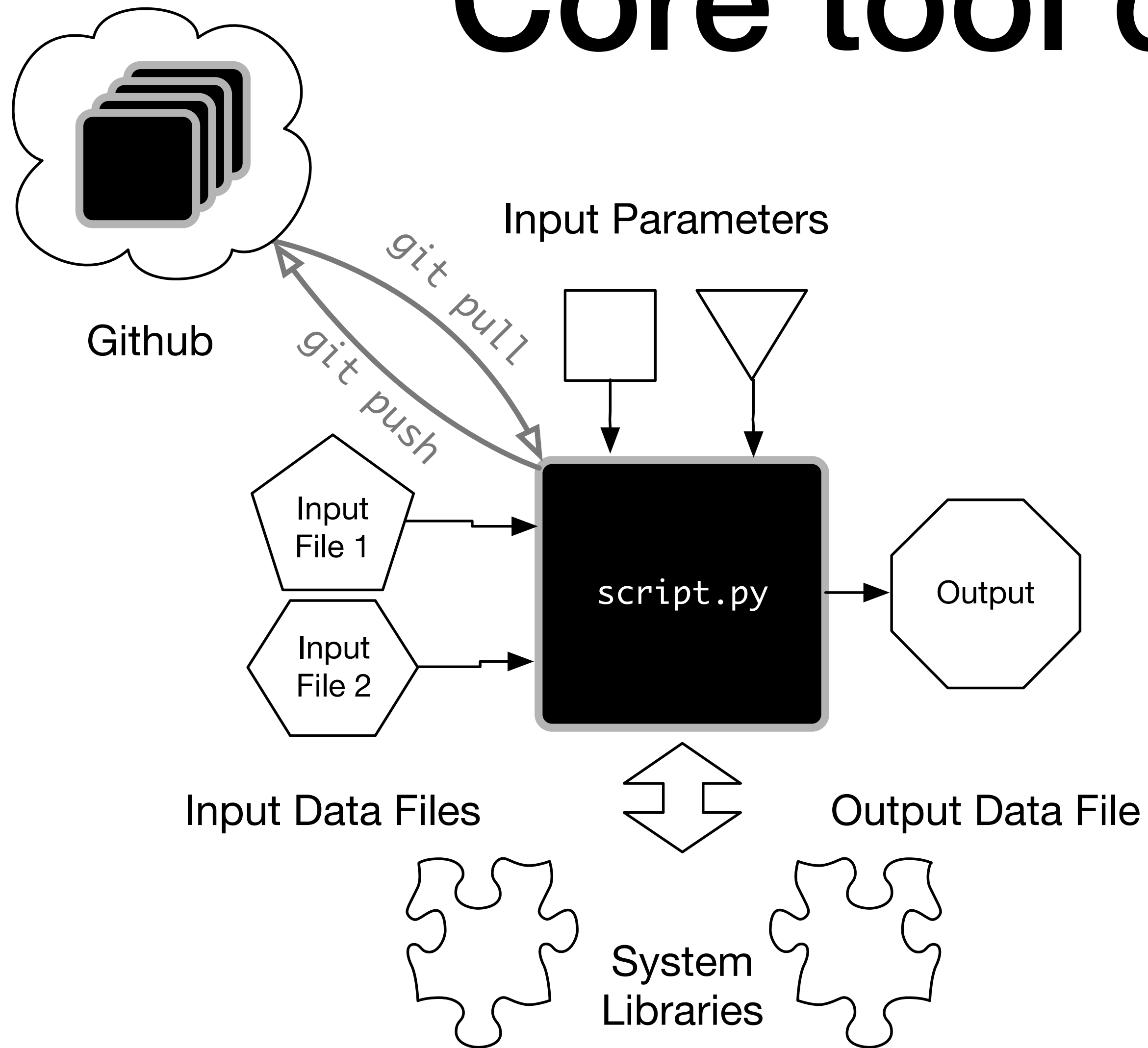
Core tool development



```
python script.py -a -x vcf -o output.dat input1.bed input2.bam
```

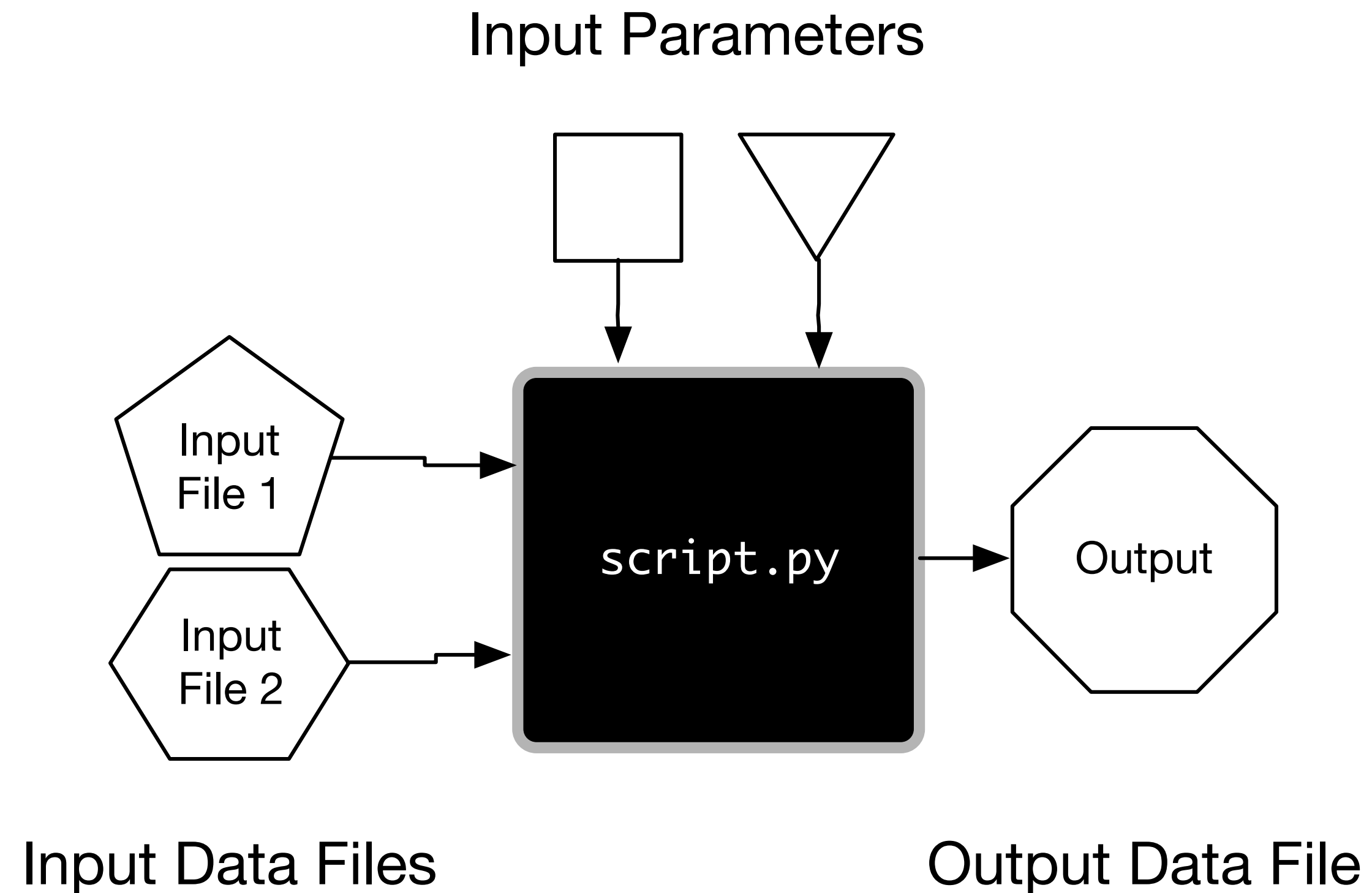
- Starting point is command line tool which:
 - Accepts input parameters
 - Accepts input files (data)
 - Writes defined set of results
- Generally processes one sample / dataset
- No queuing or LSF support

Core tool development



- Core tool typically stored on Github
- System libraries accessed implicitly
 - OS, installed software, etc
- Maintaining uniform environment important but difficult

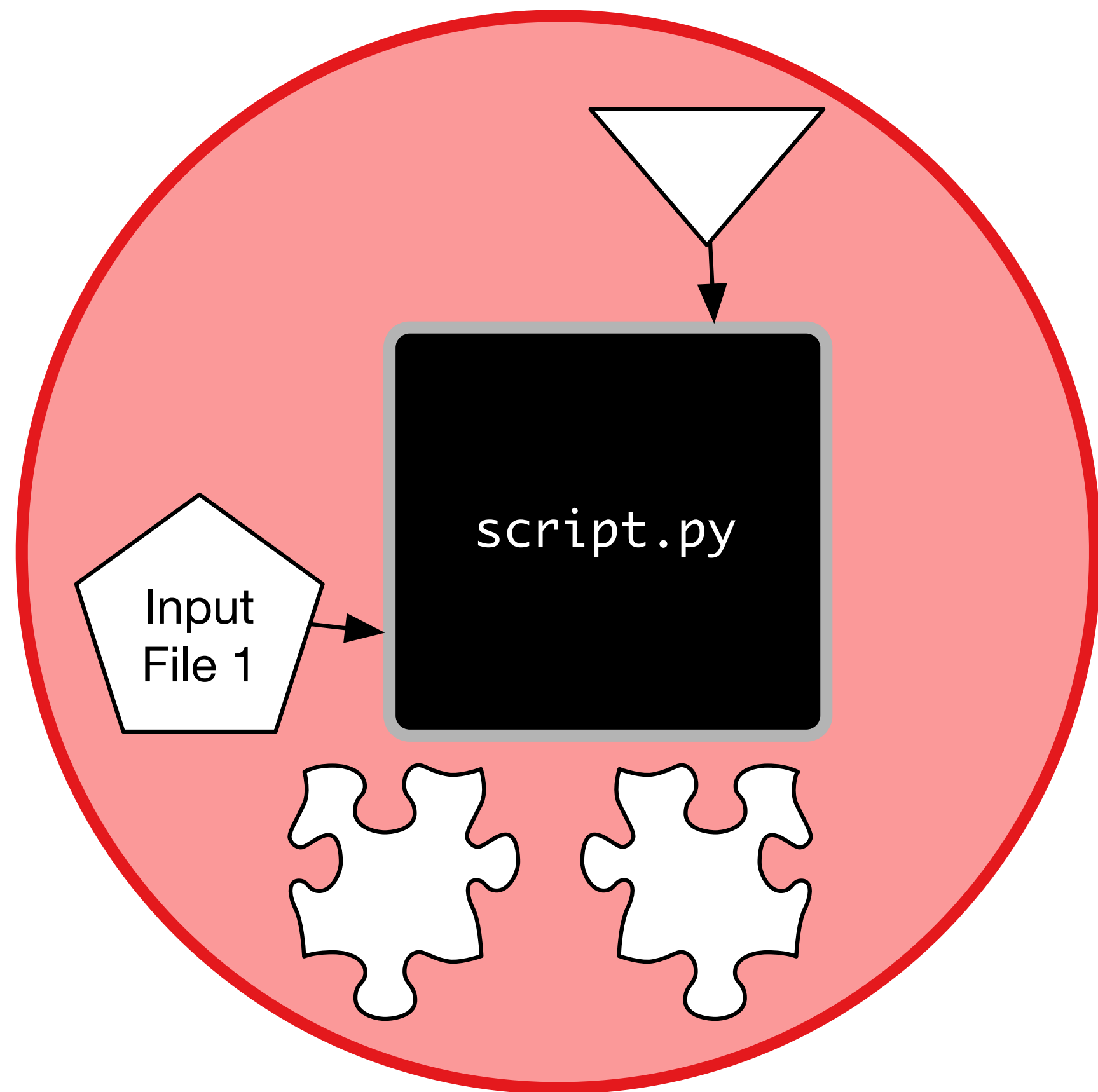
Core tool lessons



- Script must accept any input filename
 - Cannot assume what it is
- Files are staged, directories are not
 - Can pass .tar.gz if necessary
- Output path independent of input path
 - Default output filenames useful

```
python script.py -a -x vcf -o output.dat input1.bed input2.bam
```

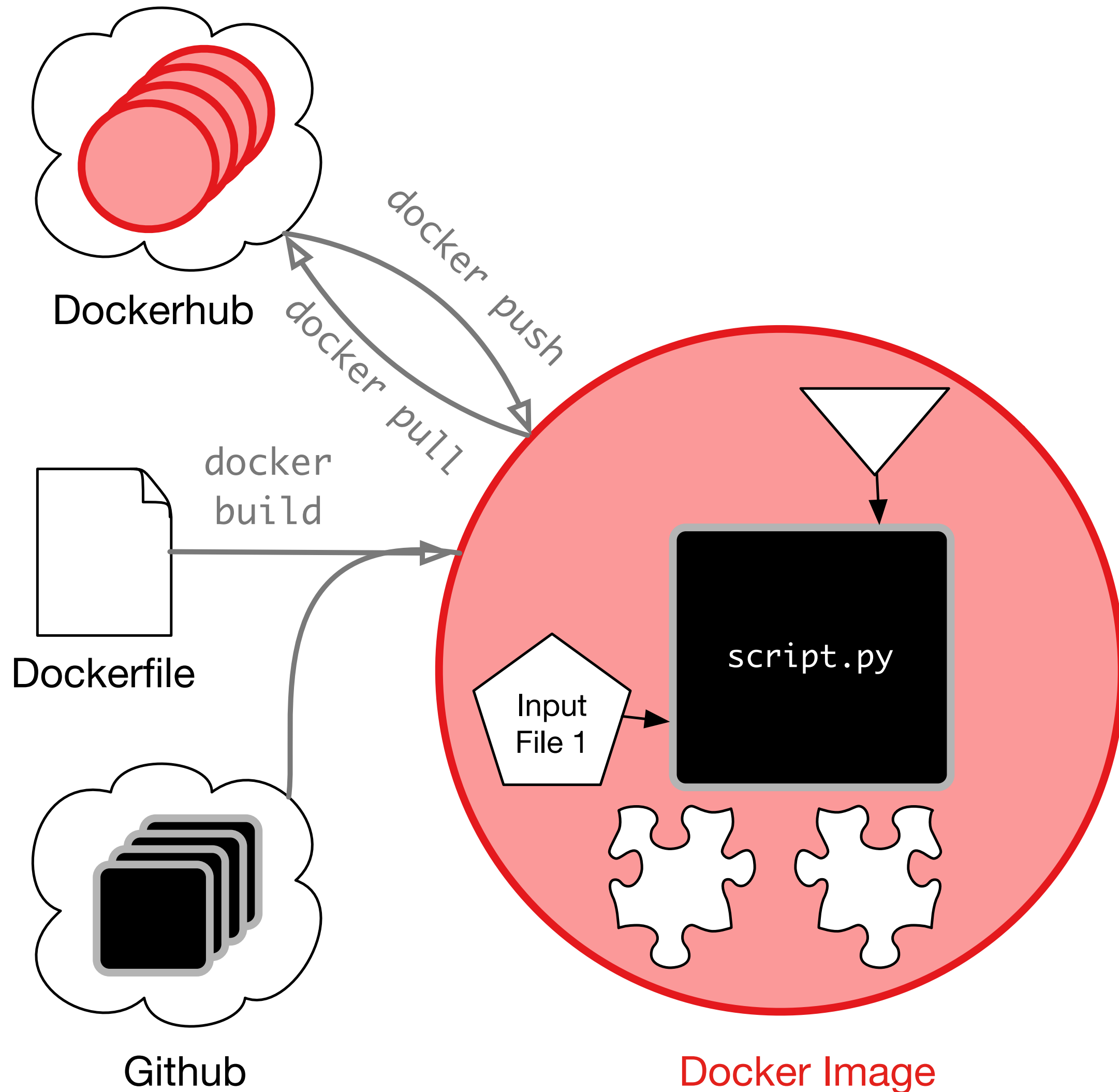
Docker



Docker Image

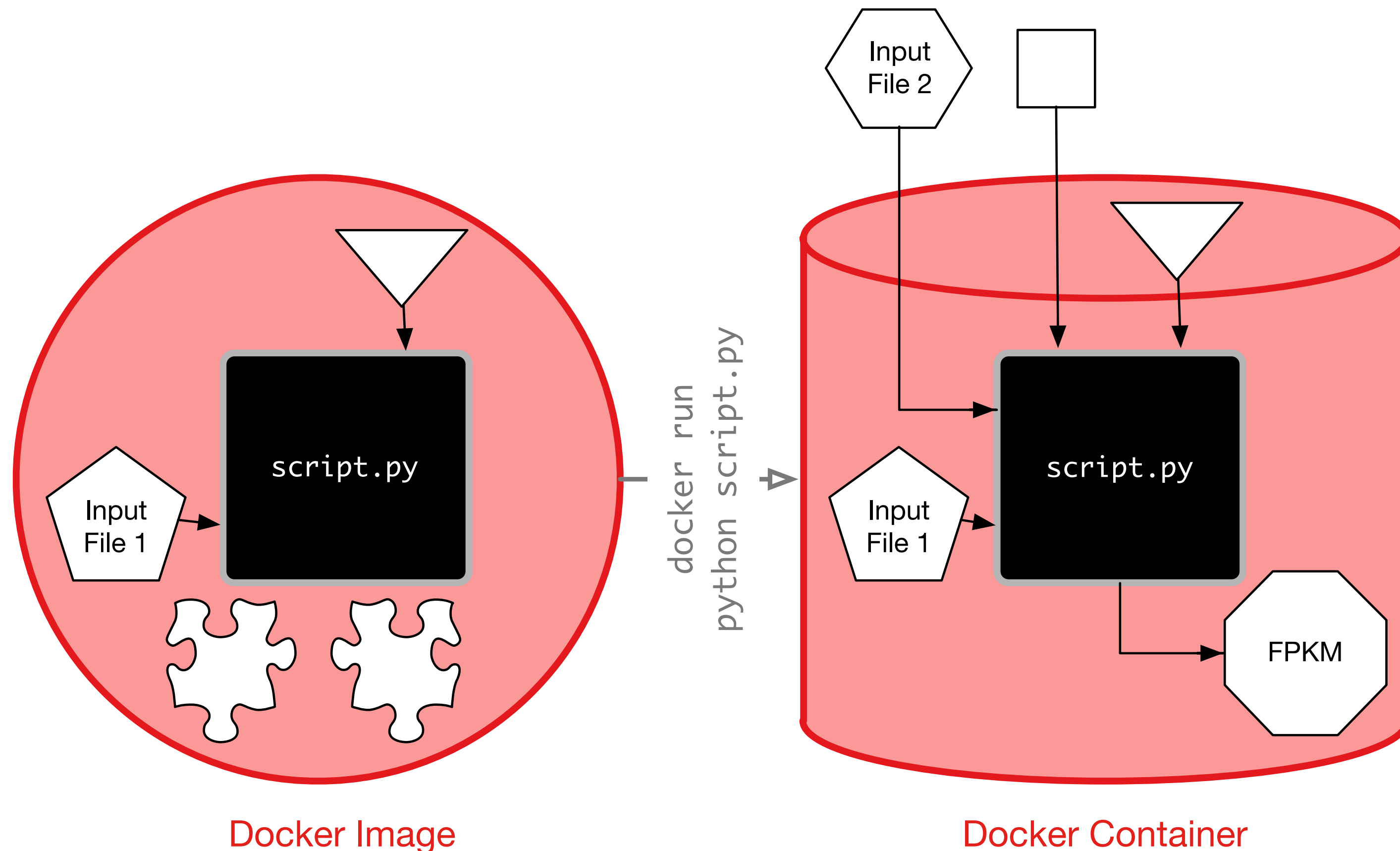
- Provides uniform environment for executing tool
- Libraries, tool incorporated into docker image
- Some (small) datasets can also be built into docker image

Docker: Making an image



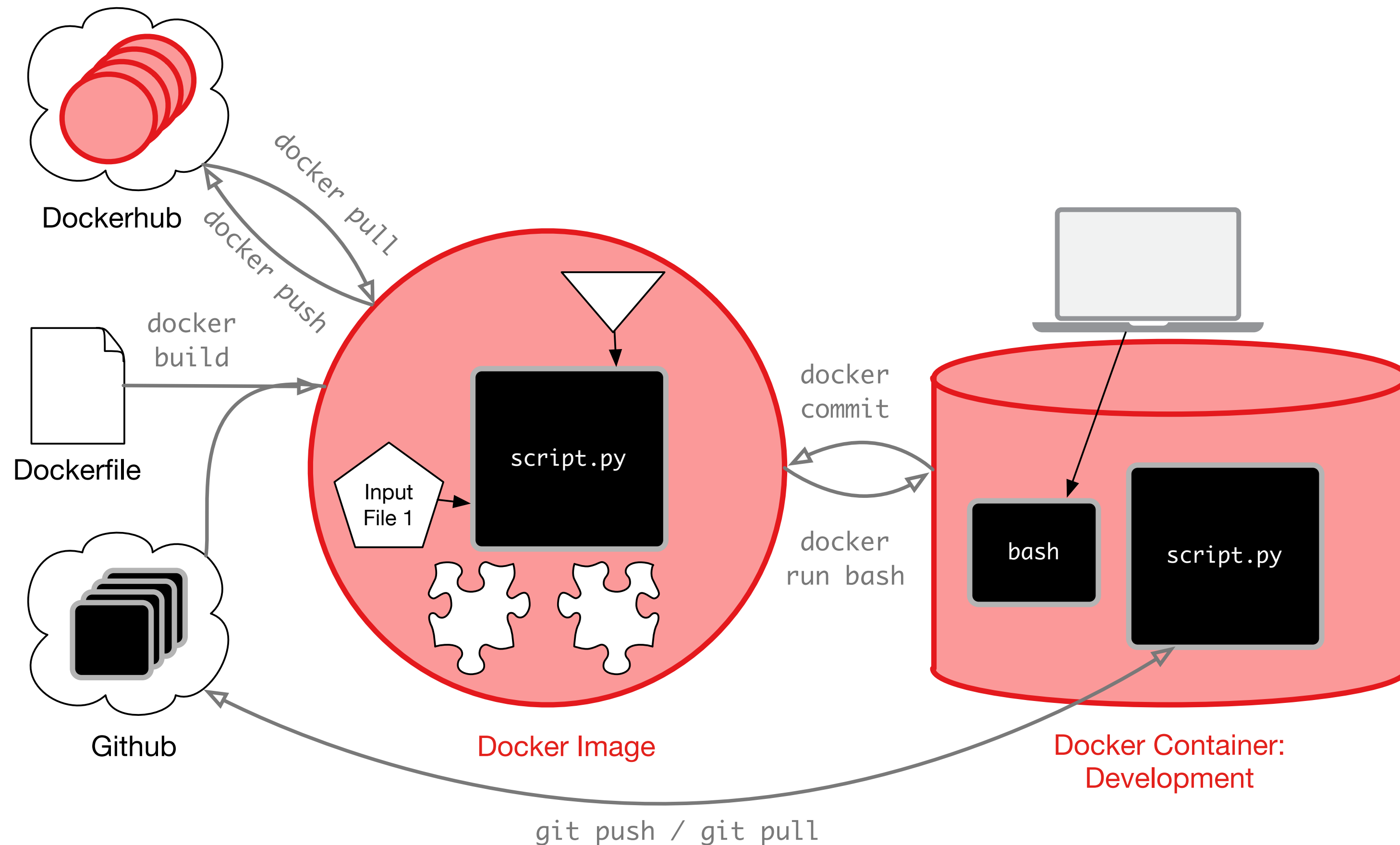
- Docker image typically built from Dockerfile
- Installs OS, all libraries
- Pulls tool code from Github
- Docker images can be stored on Dockerhub
- Push and pull images as needed

Docker: Containers



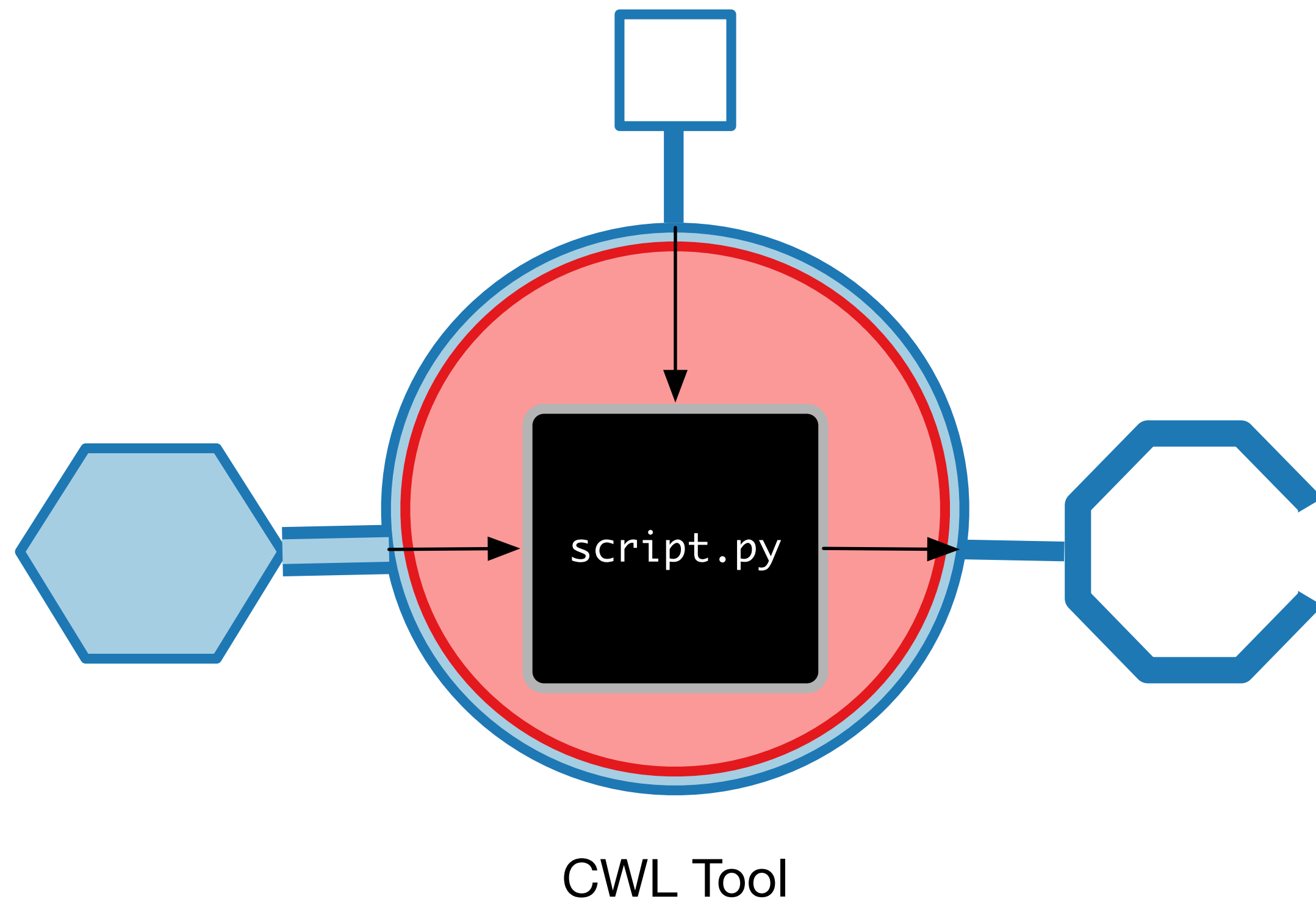
- Docker image is *recipe* for creating Docker container
- Container has allocated memory, disk space
- Typically, container executes tool script and runs to completion

Docker: Development



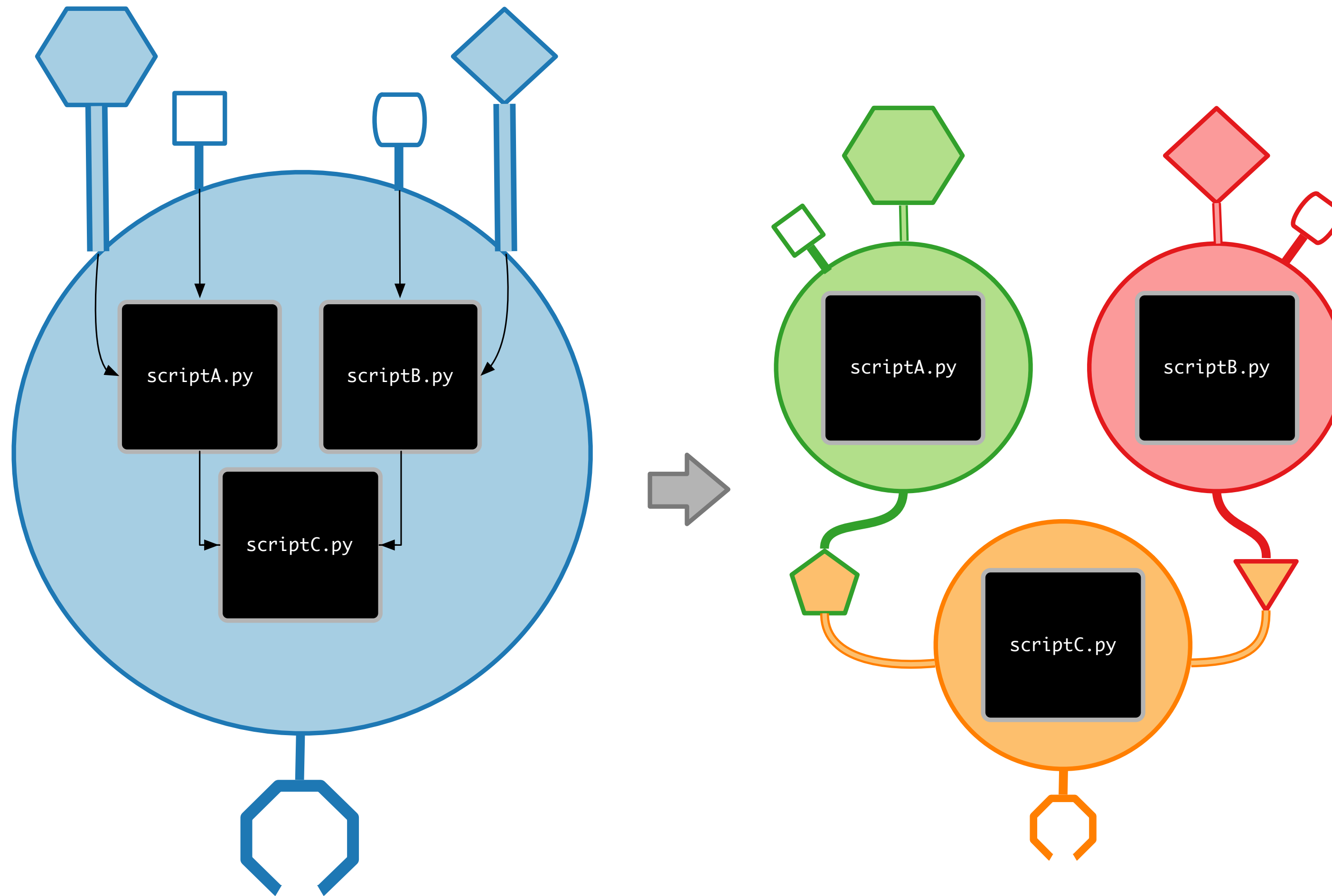
- Alternative is to run `bash` in container - good for development
- Can execute ``python script.py`` in same environment as running container
- Like SSH to container
- Docker commit will incorporate changes to local image
- Good for testing, but incorporate changes to Dockerfile eventually

CWL: Tool



- CWL is a wrapper around Docker image which provides uniform interface to workflow engines
 - Allows parallelization, scheduling, restarting, etc.
- Define input data types, input parameters, output data
- Use Rabix Composer for writing CWL tool wrappers
 - Rabix Executor for running CWL tools

CWL: Workflow



- Complex tools are better broken up into multiple tools linked together into a workflow
- Use Rabix Composer for writing CWL workflows
- Use Rabix Executor for running workflows
- Other execution engines exist

Rabix Composer Tutorial

- Use Rabix Composer to wrap a sample program in CWL
 - <http://docs.rabix.io/tutorial-1-wrapping-samtools-sort>
 - Don't need to log into images.sbgenomics.com
- See TinDaisy: CWL wrapper around SomaticWrapper
 - <https://github.com/ding-lab/tin-daisy>

