Container for reproducible data analysis

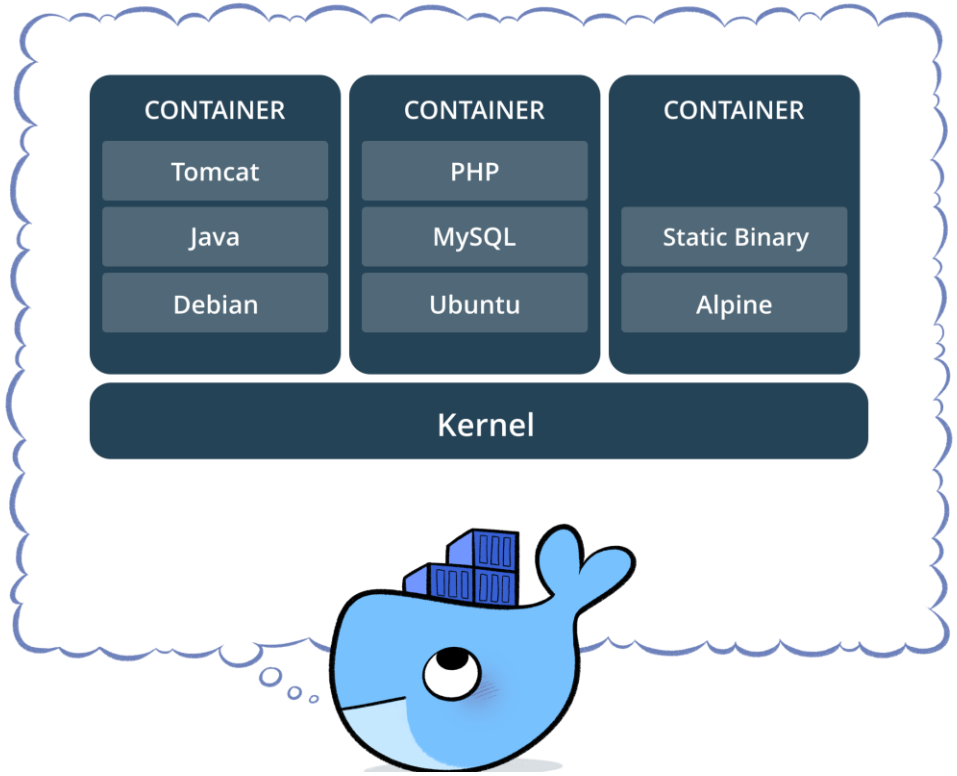
Use-cases

SIS-Containers | Q&A workshop 2022

Container introduction

- **Image** is a lightweight, standalone, executable package of software that includes everything needed to run an application.
- **Container** is a running container image.
- **Popular container engines**

(Singularity)

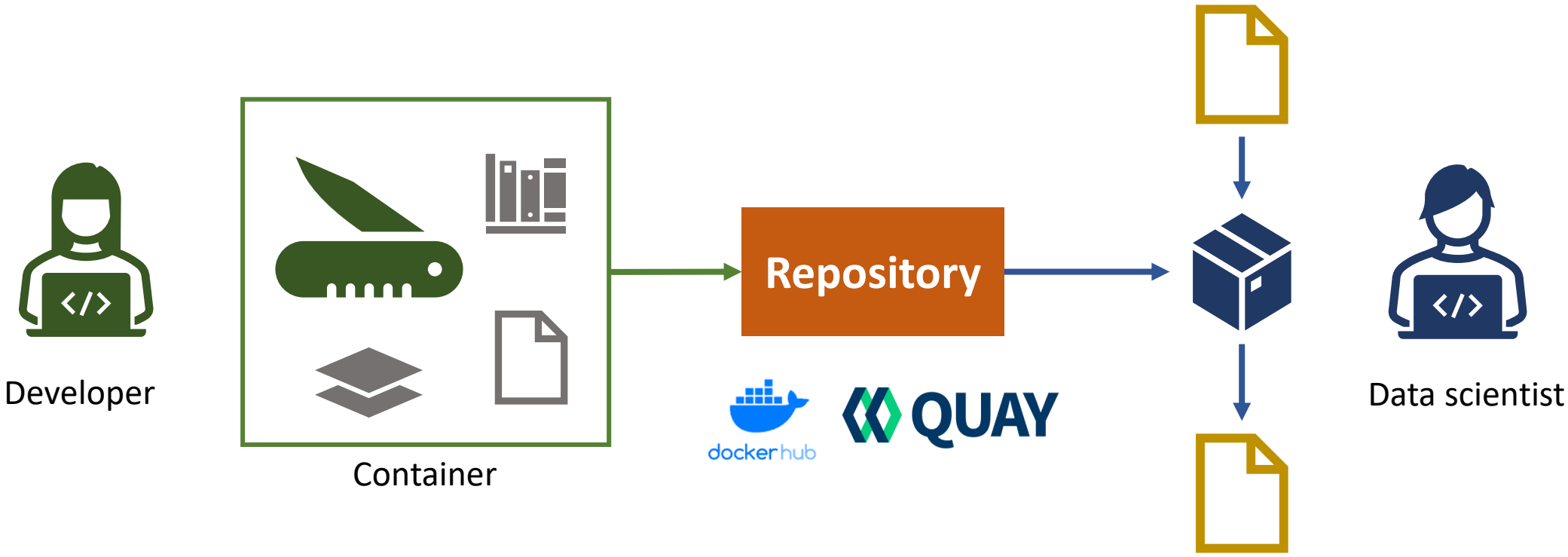


https://blog.usejournal.com/what-is-docker-in-simple-english-a24e8136b90b

Use case 1: container for data processing

Containerization of tool, environment, libraries, etc.

Use container for reproducible data processing



Building and using container

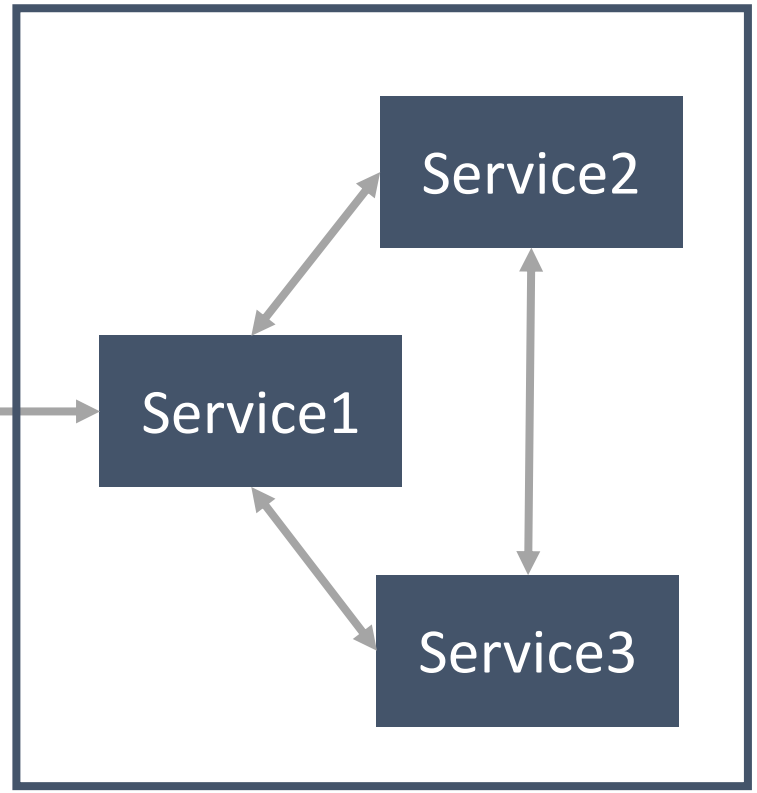
```
# FROM statement defines base image.
FROM ubuntu:22.10

# Copy files into container
COPY myscript.py /home/myscript.py

# Run tools
RUN apt-get update && apt-get install -y python

# Define default command
CMD python /home/myscript.py
```

Example containerfile

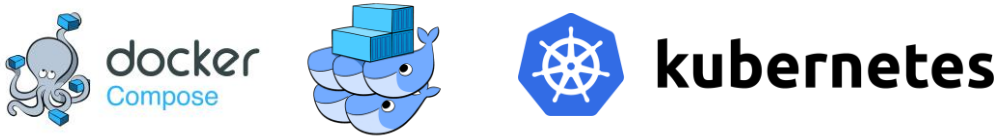
https://github.com/docker/getting-started

Use case 2: deploy services

Reseacher
Access to data
Interactive analysis
Run tools



Containerized deployment



Example services and frameworks

Deployment of services

1. **(Develop application)**
2. **Prepare cloud environment**
   OpenStack VMs, commercial cloud provider, managed Kubernetes server,
3. **Configuration**
   Networking, manage incomming traffic, define volumes, add certificates
4. **Define services**
   Compose file, Helm charts
5. **Deploy services**

```yaml
services:
  frontend:
    image: awesome/webapp
    ports:
      - "443:8043"
    networks:
      - front-tier
      - back-tier
    configs:
      - httpd-config
    secrets:
      - server-certificate

  backend:
    image: awesome/database
    volumes:
      - db-data:/etc/data
    networks:
      - back-tier

volumes:
  db-data:
    driver: flocker
    driver_opts:
      size: "10GiB"

configs:
  httpd-config:
    external: true

secrets:
  server-certificate:
    external: true

networks:
  # The presence of these objects is sufficient to define them
  front-tier: {}
  back-tier: {}
```

https://github.com/docker/getting-started

Use case 3: workflow management systems
Use container in workflows

snakemake

nextflow

Galaxy

{wdl}

COMMON WORKFLOW LANGUAGE

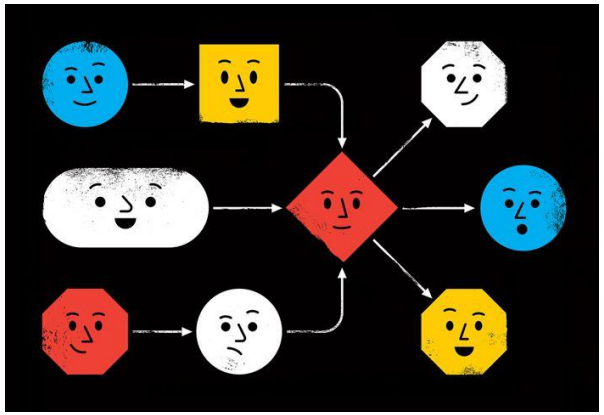
Goals for data analysis

- Reproducible - record steps and parameter
- Portable - execute on local computer or cluster or cloud
- Maintainable - organize, monitor, re-executed
- Sharable - share and re-use code

TOOLBOX

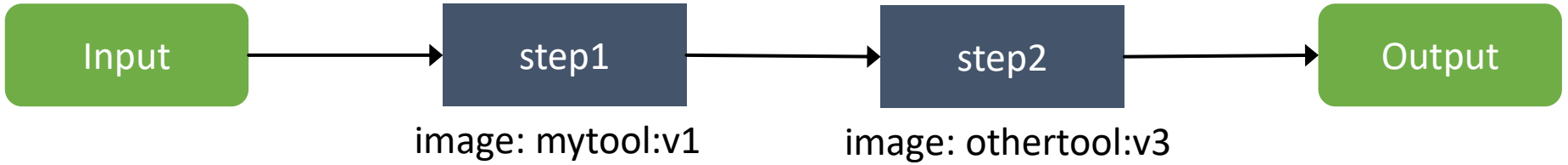
THAT'S THE WAY WE FLOW

Workflow systems turn raw data into scientific knowledge



NATURE, 02 SEPTEMBER 2019

Using container in workflow languages



COMMON WORKFLOW LANGUAGE

https://www.commonwl.org/user_guide/topics/using-containers.html

nextflow

https://www.nextflow.io/docs/latest/container.html

snakemake

https://snakemake.readthedocs.io/en/stable/snakefiles/deployment.html

{wdl}

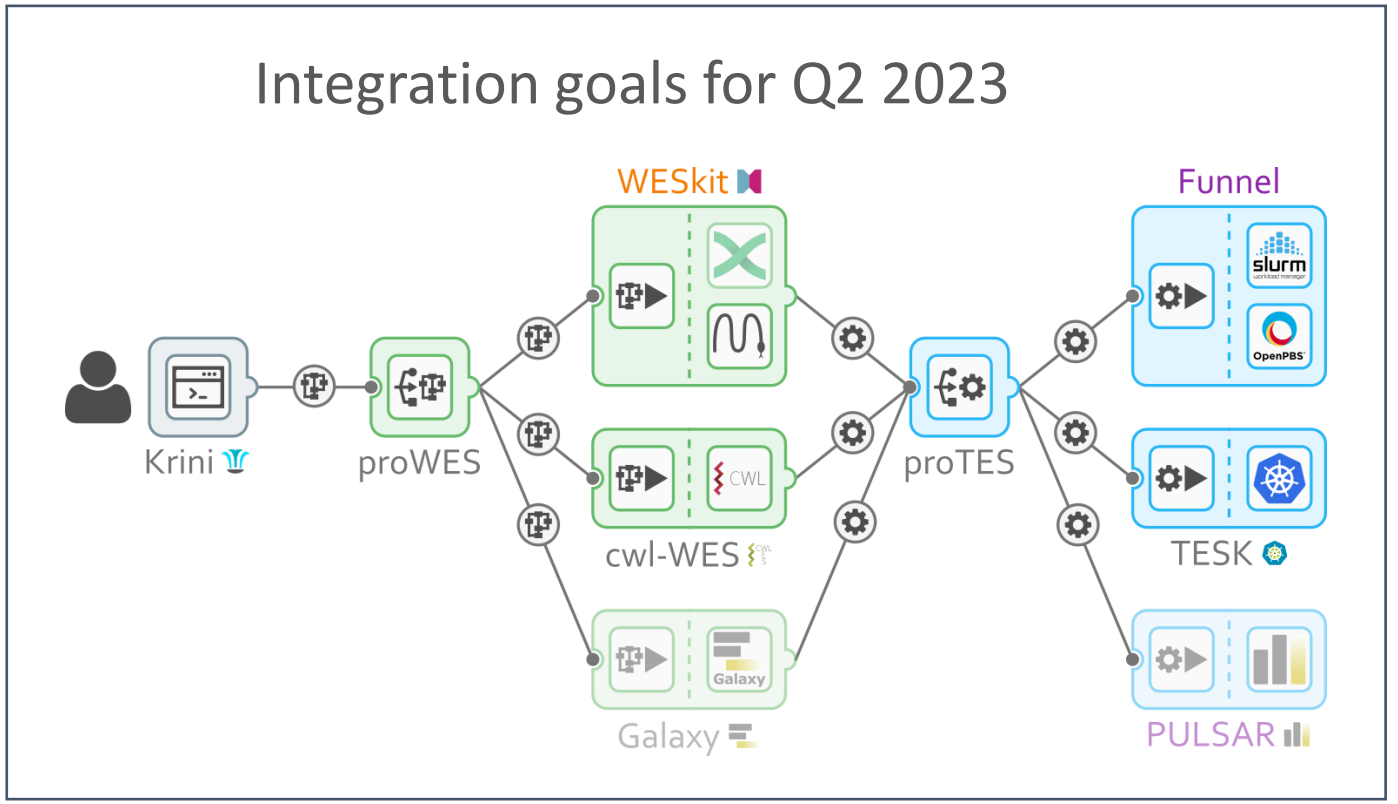
https://cromwell.readthedocs.io/en/stable/tutorials/Containers/

Next step: running workflows in cloud environment

ELIXIR Cloud & AAI

https://elixir-europe.github.io/cloud/

GA4GH cloud specs
WES: workfow execution service
TES: task execution service



(from Alex slides)

Summary

Typical container use-cases

1. Create und use container
2. Deploy services
3. Use container in workflows
4. Execute workflows via cloud
5. Others?