ETH-RAT

SDSC

RENKU - 連句

Reproduce, Reuse, Recycle Research

Rok Roškar and the SDSC Renku team

Renku-Reana workshop @ CERN | 26.06.2018

Goals of Renku

1.
Provide the means to create **reproducible** data science

2.
Facilitate the **sharing** and **reuse** of research artefacts

3.
Foster a **collaborative environment** for interactive prototyping

4.
Enable the **discovery** of relevant data and methods

5.
Allow **federated access** across institutions giving each the freedom to impose its own access controls over resources

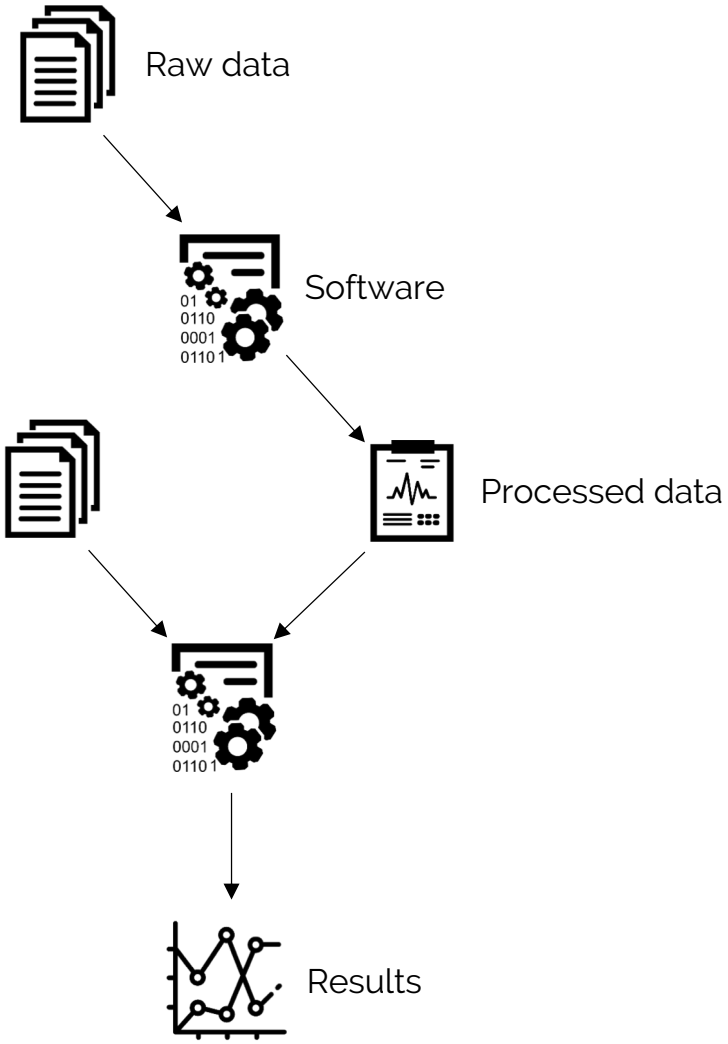
SDSC

Capturing, recording and utilizing the lineage of results is the core of Renku

Terminology

- We borrow the **Renku** name from the Japanese word for *linked-verse poetry*
- A "**ku**" is a verse in a renku poem
- We use "**ku**" to mean a piece of the data analysis process – includes discussion, code, and results

Capture the scientific process

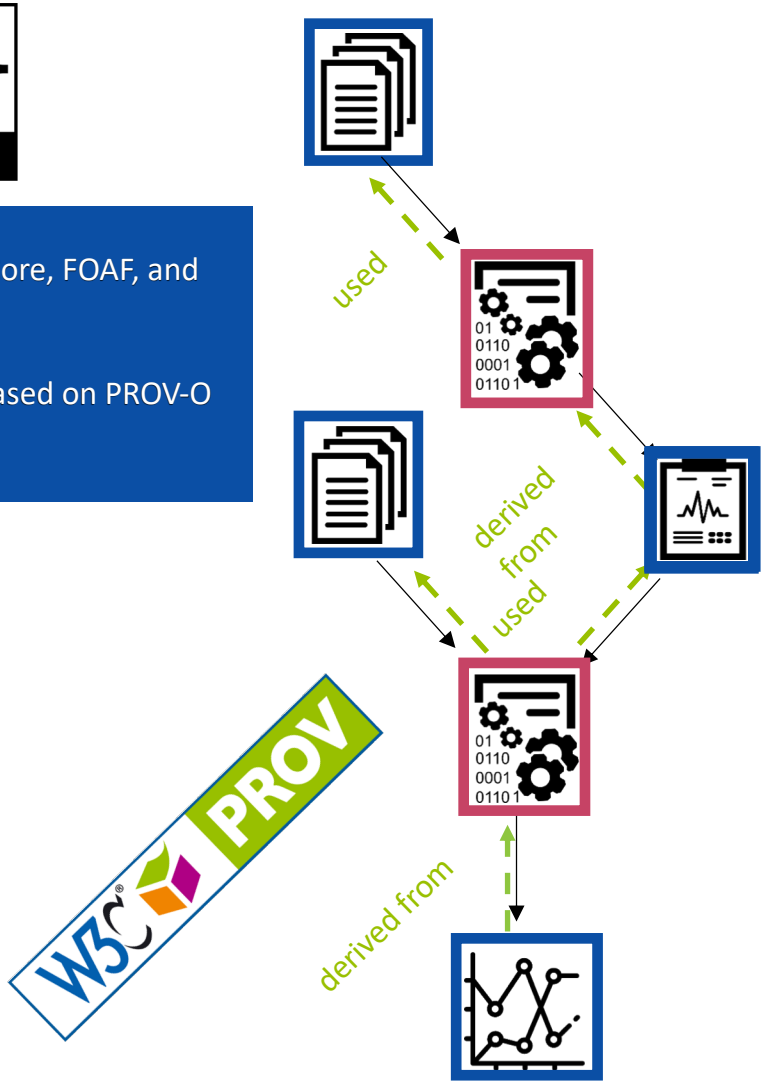


1. Lineage is recorded into a knowledge graph
2. Steps can be repeated and reused
3. Version control is built-in for data, code, and workflows
4. Lineage accessible via simple tools

Metadata

JSON-LD

- Metadata use Dublin Core, FOAF, and Schema.org
- Provenance graph is based on PROV-O W3C recommendation

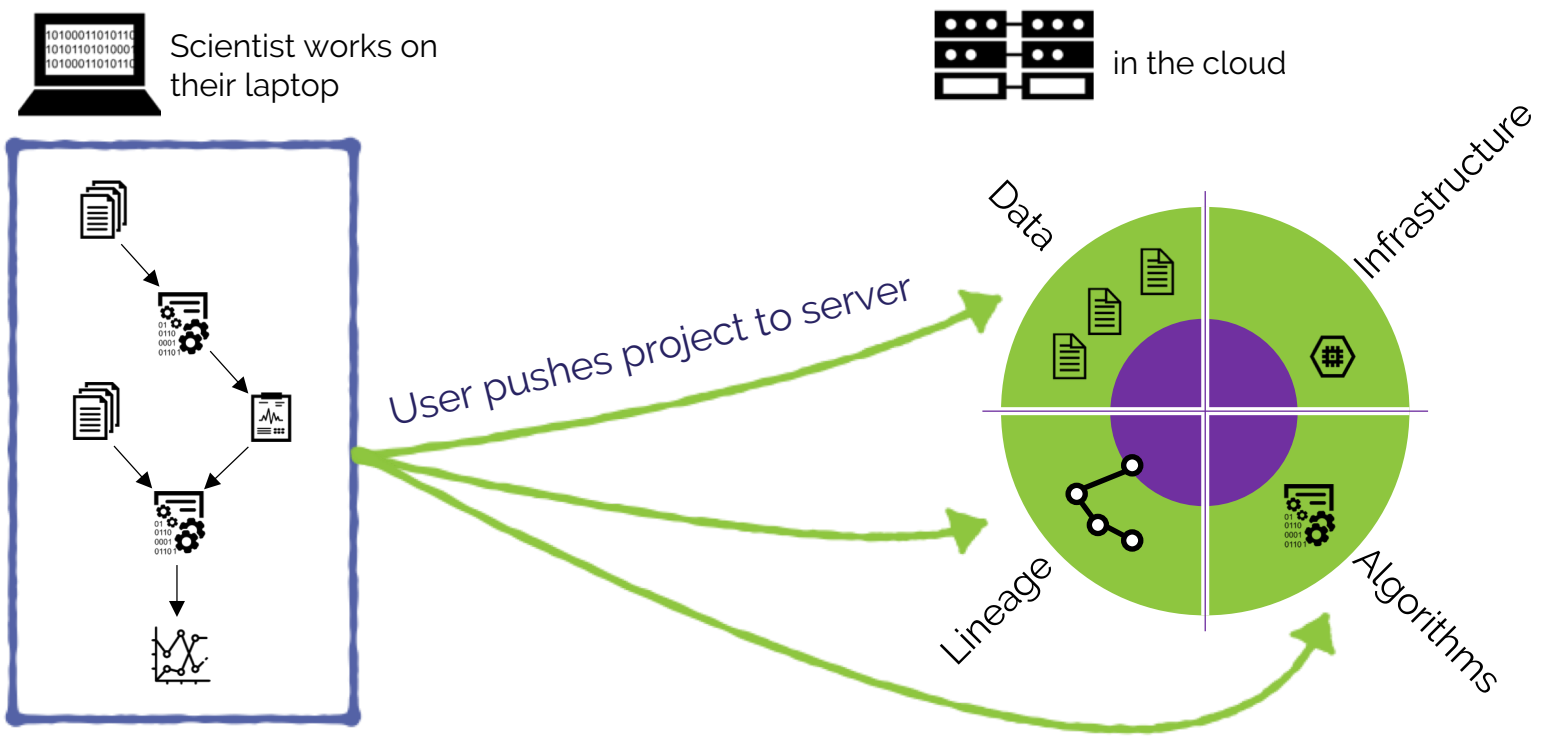


COMMON WORKFLOW LANGUAGE

- CWL for representing all computational steps
- Capture individual steps from user input
- Tools for constructing workflows from basic pieces
- Rely on container technologies to ensure reproducibility

SDSC

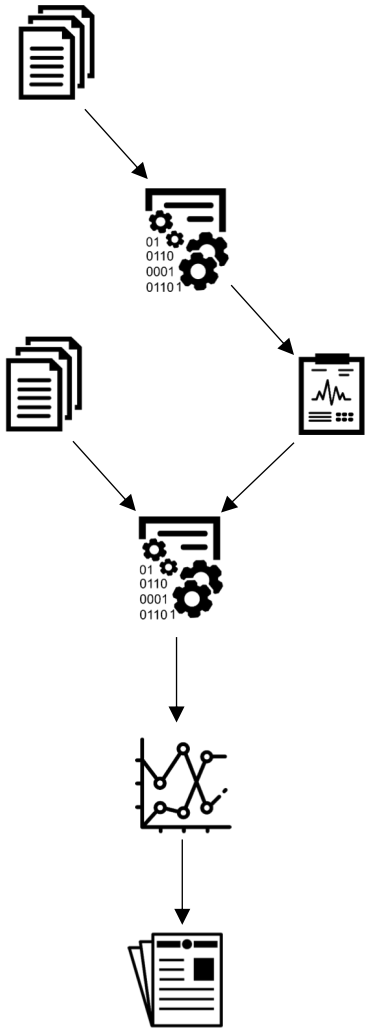
Verify and share results



1. Results automatically verified
2. Data lineage captured “live”
3. Knowledge graph populated
4. Shareable interactive environment created

Discover and understand the work of others

Graph-based search…

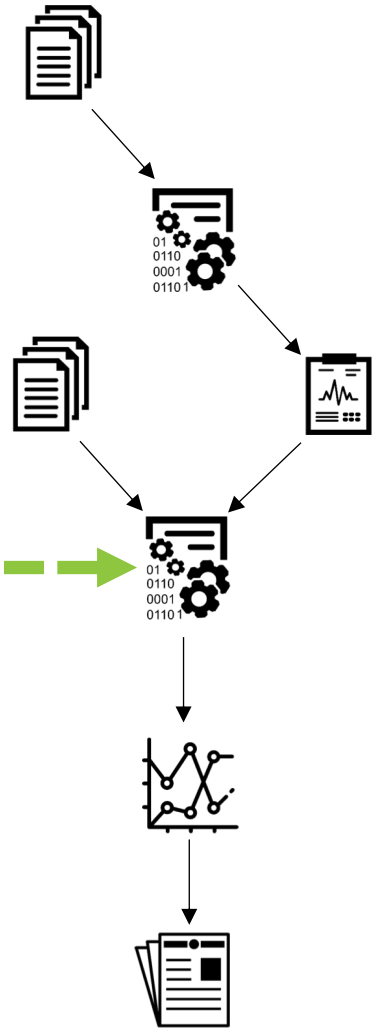


Ex: search for **data** and explore the tools to efficiently generate results

Discover and understand the work of others

Graph-based search…

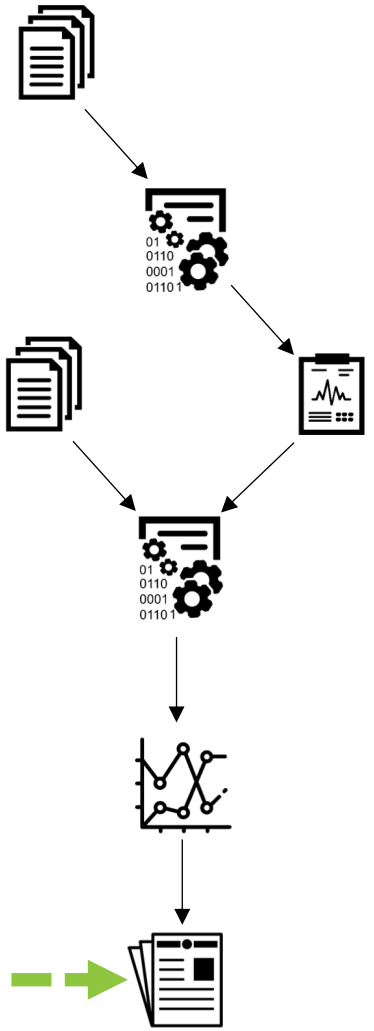
Ex: search for an **algorithm** and see its applications



Discover and understand the work of others

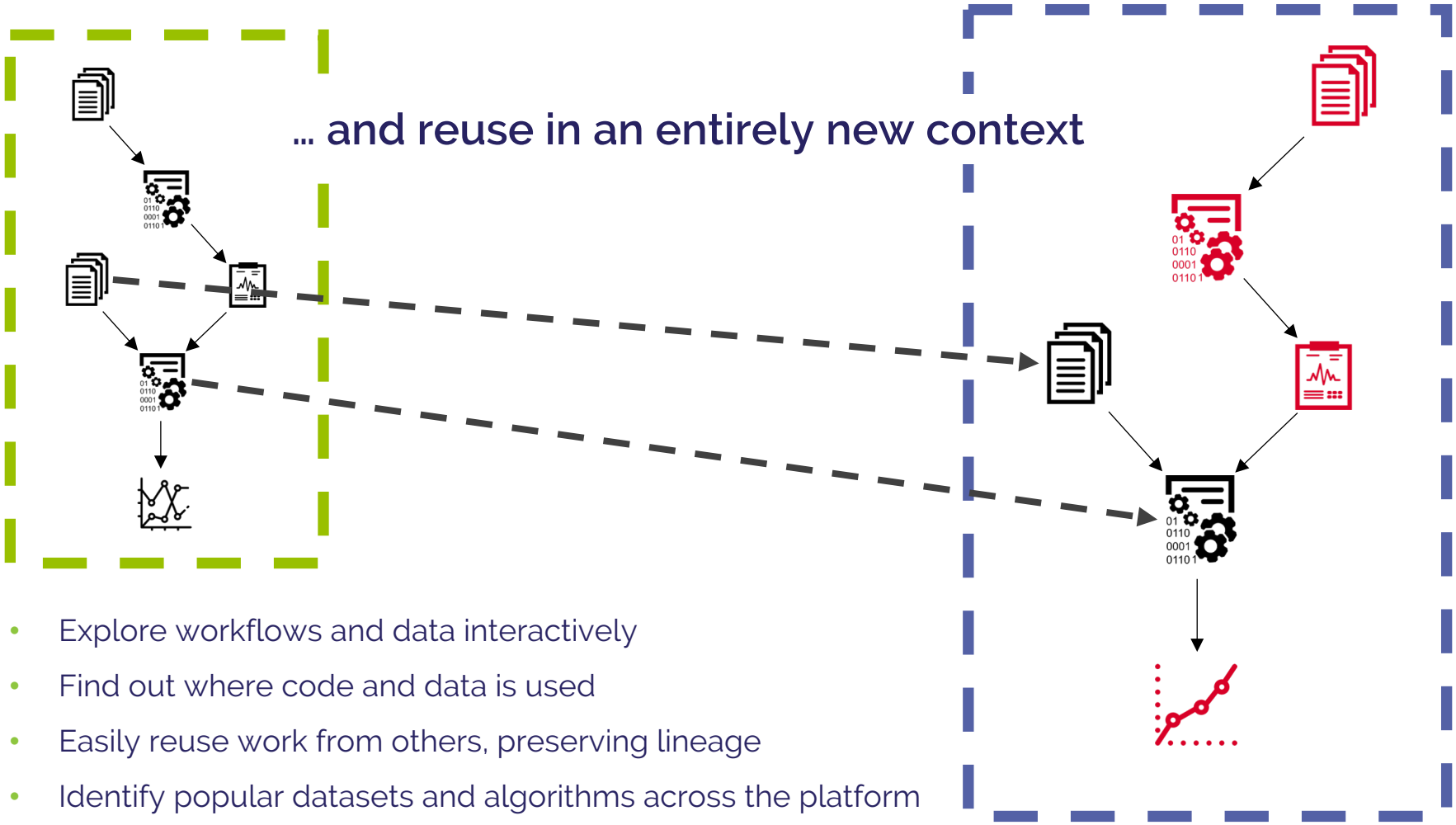
Graph-based search…

Ex: search for a **publication**, obtain a full view of how the results were obtained



Reuse and repeat

Search for relevant data or algorithms…



- Explore workflows and data interactively
- Find out where code and data is used
- Easily reuse work from others, preserving lineage
- Identify popular datasets and algorithms across the platform

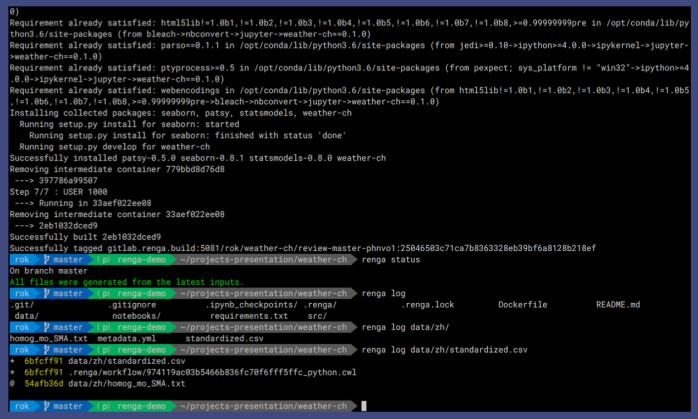
SDSC

Typical Renku Flow

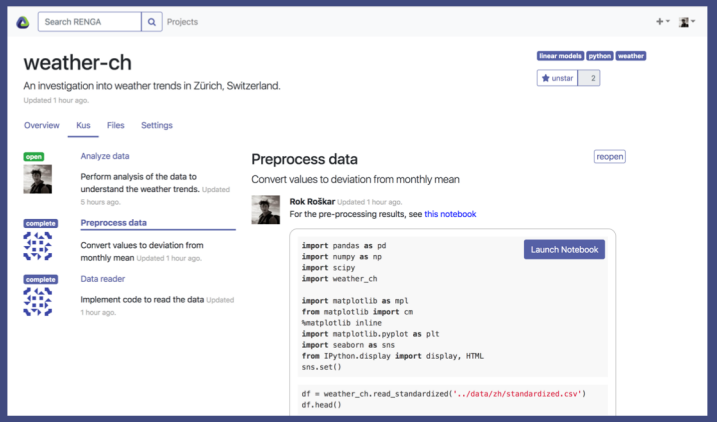
1. Initialize a project in the terminal with CLI
2. Import data
    1. URL/git/local filesystem
3. Execute some work generating a lineage of results
4. Push to server
    1. Optionally verify last step (CI for the workflow/step)
    2. Build an image for the commit
5. Share link to launch hosted notebook
    1. Create changes, commit back the notebook
6. Inspect workflow
    1. Adapt inputs/parameters
    2. Rerun workflow(s) on the cloud infrastructure
    3. Create new workflows from existing pieces (could be from other projects!)

Building easy-to-use tools on top of trusted technologies

Renku consolidates the open-source data science and software engineering technologies into a single platform



Command-line interface



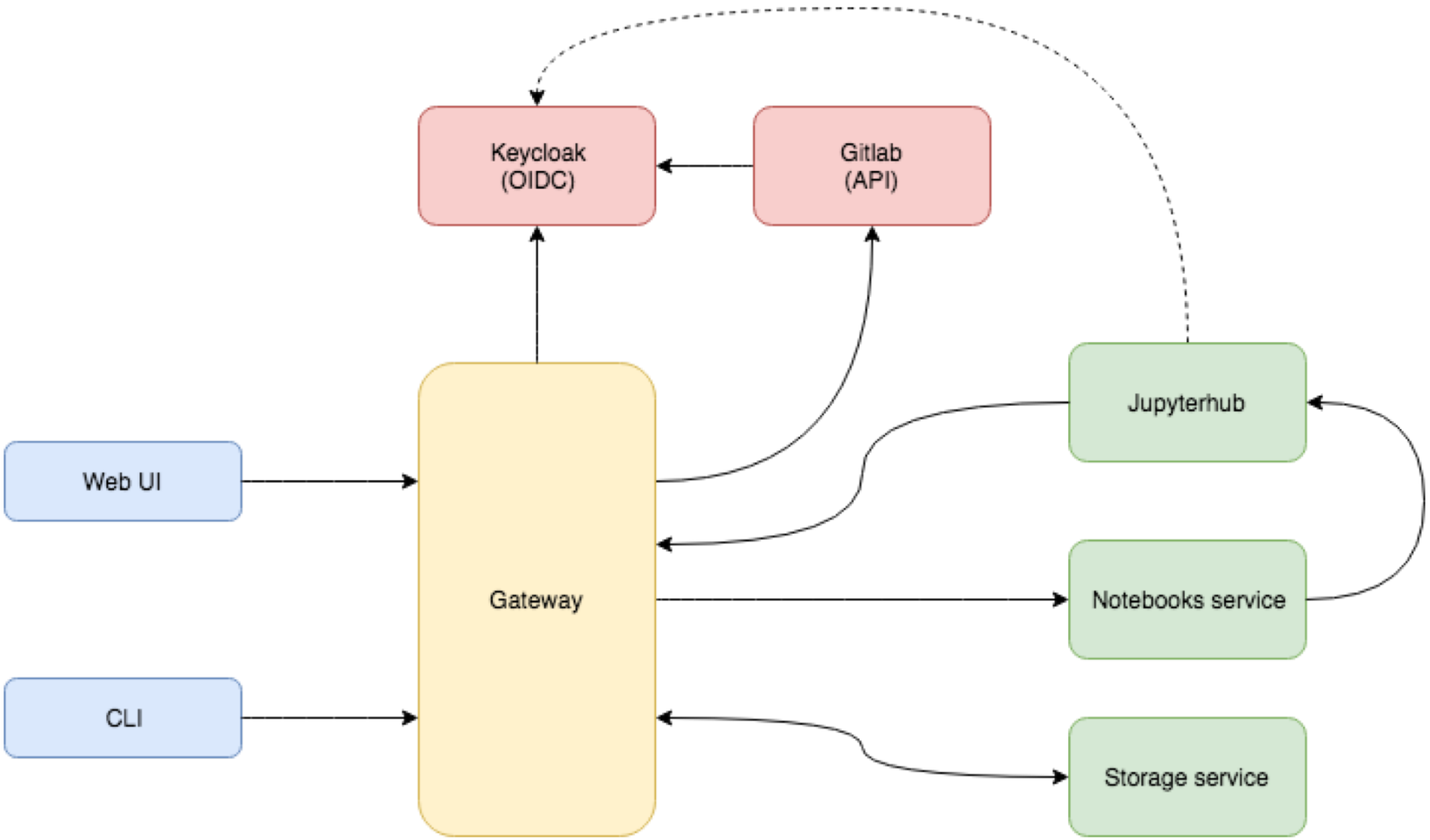
Web-based front-end

SDSC

Platform components and technology

- **`renku-ui:`** react.js + bootstrap
- **`renku-python:`** click-based python CLI and API client
- **`renku-notebooks:`** flask app interacting with jupyterhub
- **`renku-storage:`** git-lfs & S3 API with scala/play
- **`renku-gateway:`** API Gateway (in-progress)
- **`renku-graph:`** Knowledge Graph (in progress)
- Workflow execution (CWL+???)
- Repository management + CI (vanilla GitLab)
- Authentication (Keycloak)
- Runtime (docker + kubernetes)

Basic Architecture



Deployment

- Deployed via kubernetes helm charts (docker-compose option for development)
- Main chart in top-level repo, but each service can be standalone
- Development charts pushed on every commit to a chart registry for continuous deployment
- Stable chart repository (upon release)
- Cloud (SWITCH OpenStack) + HPC environments (CSCS)

https://swissdatasciencecenter.github.io/helm-charts

Development and Roadmap

- Q1+Q2 2018: Redeveloped ground-up
  - Provide a user interface for project/discussions
  - Create interactive environments on-demand
  - Track lineage within and across projects
  - Automatic update of results based on lineage
  - Basic “cloud” workflow with interactive notebooks including custom images
- Q3 2018: solidify the base, fill in the gaps
  - Hosted platform for a limited audience
  - Collect use-cases, obtain feedback (bugfixes!)
  - Index data into an instance-wide Knowledge Graph
  - Capture component interactions on-line via events
  - Basic support for (re)executing workflows in the cloud through the UI
- Q4 2018:
  - Introduce access controls
  - Exploit the Knowledge Graph for user queries
- 2019 → federation

Current status

Platform is under very **active** development:

https://github.com/SwissDataScienceCenter

`renku` — services & deployment recipes
`renku-python` — CLI and Python API
`renku-ui` — Web front-end
`renku-notebooks` — external JH service
`renku-storage` — storage service (LFS+S3)