gin-proc snakemake microservice workflow [DRAFT]

GIN CI VM

GIN CI User forms

GIN forward to form on CI

a) Auto-create .gin-ci-conf.yaml
   - select build type
   - select required files
b) Update existing config file

Commit config file

Local

GIN

gin-proc

repo [master]

repo [master]

snakemake CI container (b)

Cluster

repo [master]

change in 'in.dat'
commit -m "gin-proc build"
push / PR

snakefile
codefile.py
in.dat

.gin-proc-conf.yaml

webhook (a)

- gin get repo
- parse .gin.ci-conf.yaml
- gin get submodules
- install dependencies
- run snakemake

execute rules
gin get content (c)

snakemake conda

authentication (d)

Repo cache, results

- out.dat

backflow (e)

Caching repositories
Intermediate results
Zip download results

Job queue
Container registry
Statistics
RunLogs
DAG (f)

gin-proc features

(a) [user]/[repo]/[branch]/[cibuild]
    e.g. mdenker/reponame/master/snakemake
(b) container persistence and reuse
(c) timestamp handling: local vs. commit(b) snakemake specific remotes
    possible gin remote
(d) authentication via gin token exchange
(e) backflow of out.in to the gin repo
    - new branch + pull request
    - downloadable zip file
(f) snakemake build graph visualization

snakefile example:
rule:
  input: gin://in.dat
  output: gin://out.dat -m "job success commit message"
  shell: python codefile.py in.dat out.dat