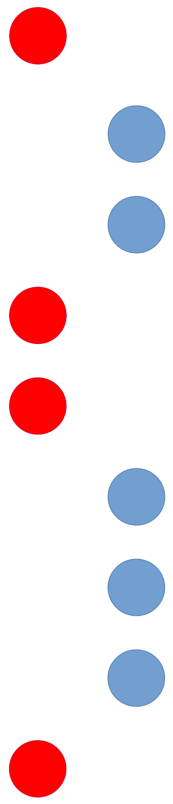
CI / CD introduction
Develoment process

Jean-Luc Delarbre

Git: branching model

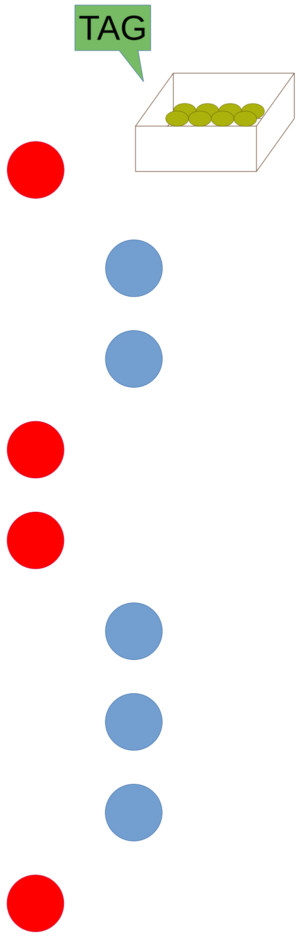
- **At least 2 branches:**
  - main (master) ⇨ client delivery
  - Develop ⇨ working branch
  - Often feature branches
- **Purpose: sort potatoes**

We want our potatoes not to rot



Industrial development requirement

- **Traceability**
  - Sort potatoes
    - Branching model, tag, release
    - Merge request
    - ...
- **Reliability**
- **Reproducibility**



Professional standard of development

- **Very often a missing practice**
  - Project bad example
    - No main branch: where are the delivery
    - Develop: hard to rebuild project
    - No link between doc and features in code

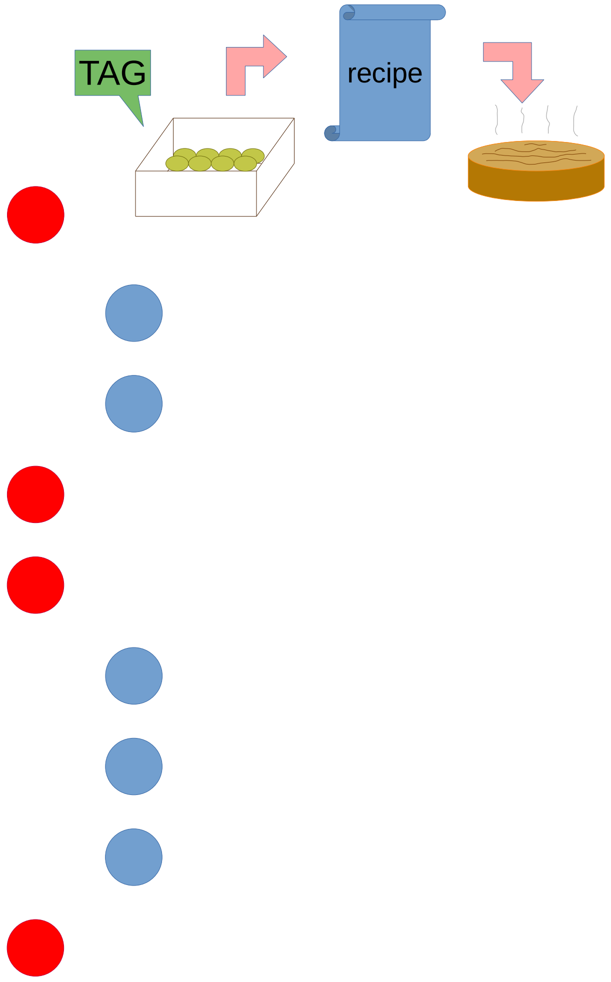
Professional development: how to

- **Use CI / CD of your DevOps platform (gitlab, github...)**
  - Define CI/CD configuration file
    - .gitlab-ci.yml
    - Build, test, deployment and documentation

Industrial development requirement

- **Traceability**
  - Sort potatoes
    - Branching model, tag, release
    - Merge request
    - …
- **Reliability**
  - Tests
- **Reproducibility**
  - Link delivery to build script

    ⇨ pipeline



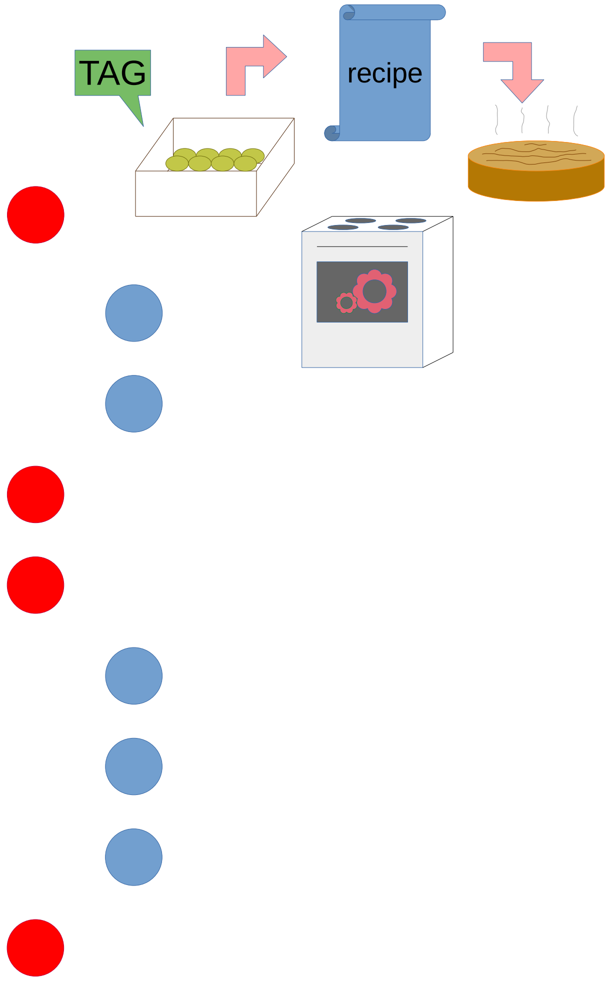
Professional development: how to

- **Use CI / CD of your DevOps platform (gitlab, github…)**
  - Define CI/CD configuration file
    - .gitlab-ci.yml
- **Docker**
  - Reproducible and portable build environment
    - Dockerfile: environment as code
      - Better than documentation
      - Reusable competence across projects vs specific development environment

Industrial development requirement

- **Traceability**
  - Sort potatoes
    - Branching model, tag, release
    - Merge request
    - ...
- **Reliability**
  - Tests
- **Reproducibility**
  - Link delivery to build script

    ⇨ pipeline
  - Registry



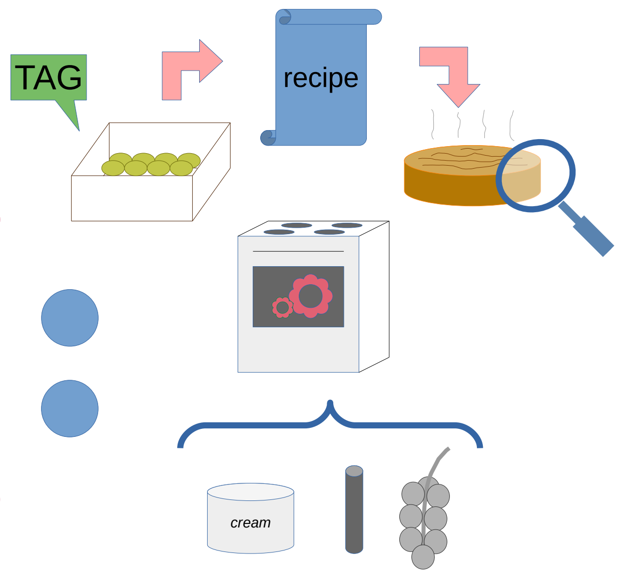
Professional development: how to

- **Use CI / CD of your DevOps platform (gitlab, github…)**
  - Define CI/CD configuration file
    - .gitlab-ci.yml
- **Docker**
  - Reproducible and portable build environment
    - Dockerfile: environment as code
      - Better than documentation
      - Reusable competence across projects vs specific development environment
- **Package manager**
  - Often third parts are needed
  - Maven, pip, npm, conan

Industrial development requirement

Automated

- **Traceability**
  - Sort potatoes
    - Branching model, tag, release
    - Merge request
    - …
- **Reliability**
  - Tests
- **Reproducibility**
  - Link delivery to build script
    ⇨ pipeline
  - Registry
  - Package manager



- No poisoned dish
- One click delivery
- Collaborative work
  - Developers
  - others