Kevin Wang

Structure of SC_CPC_workshop

"Open-source" does not mean putting codes that only you can run on GitHub.

Kevin Wang*

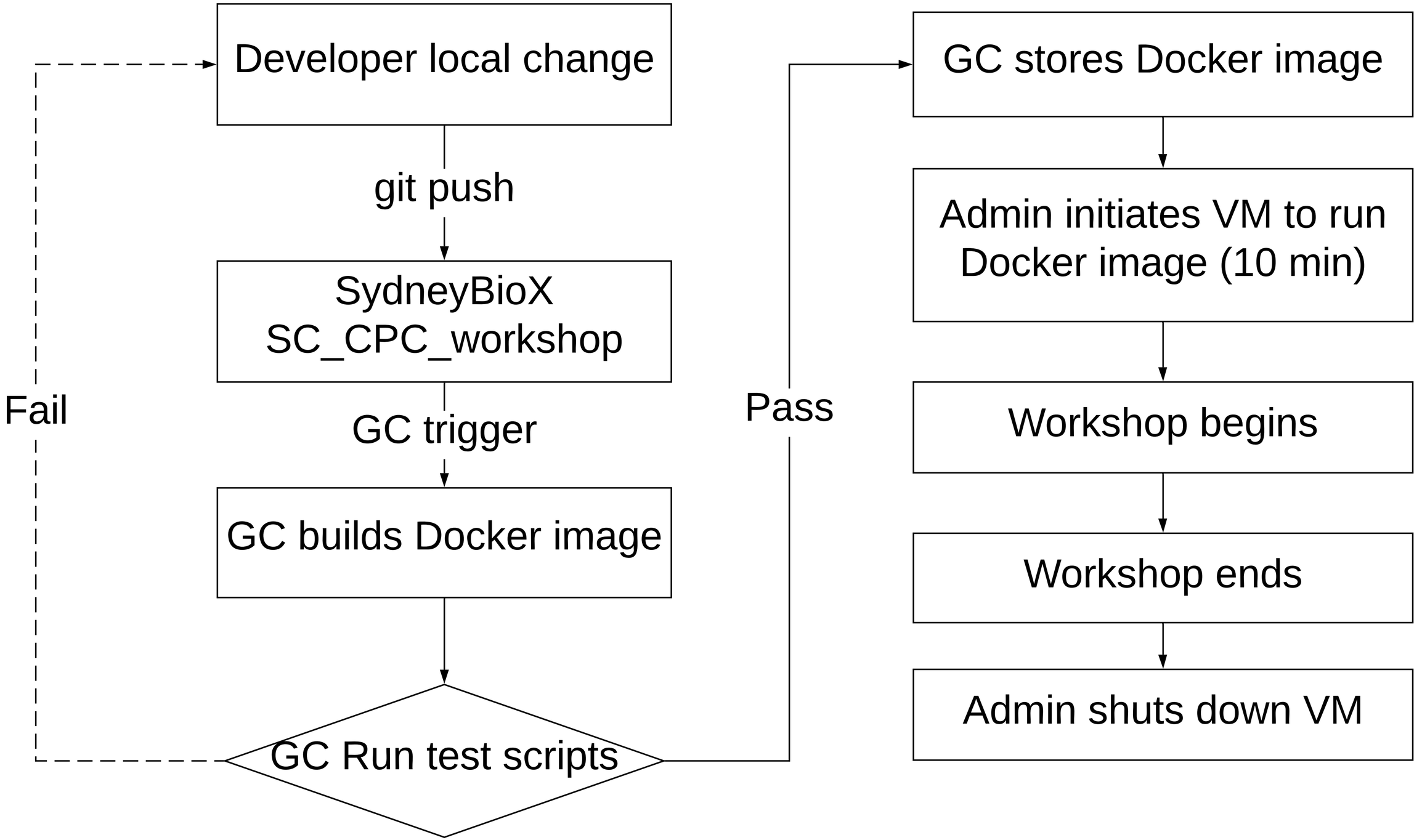
*That's me!

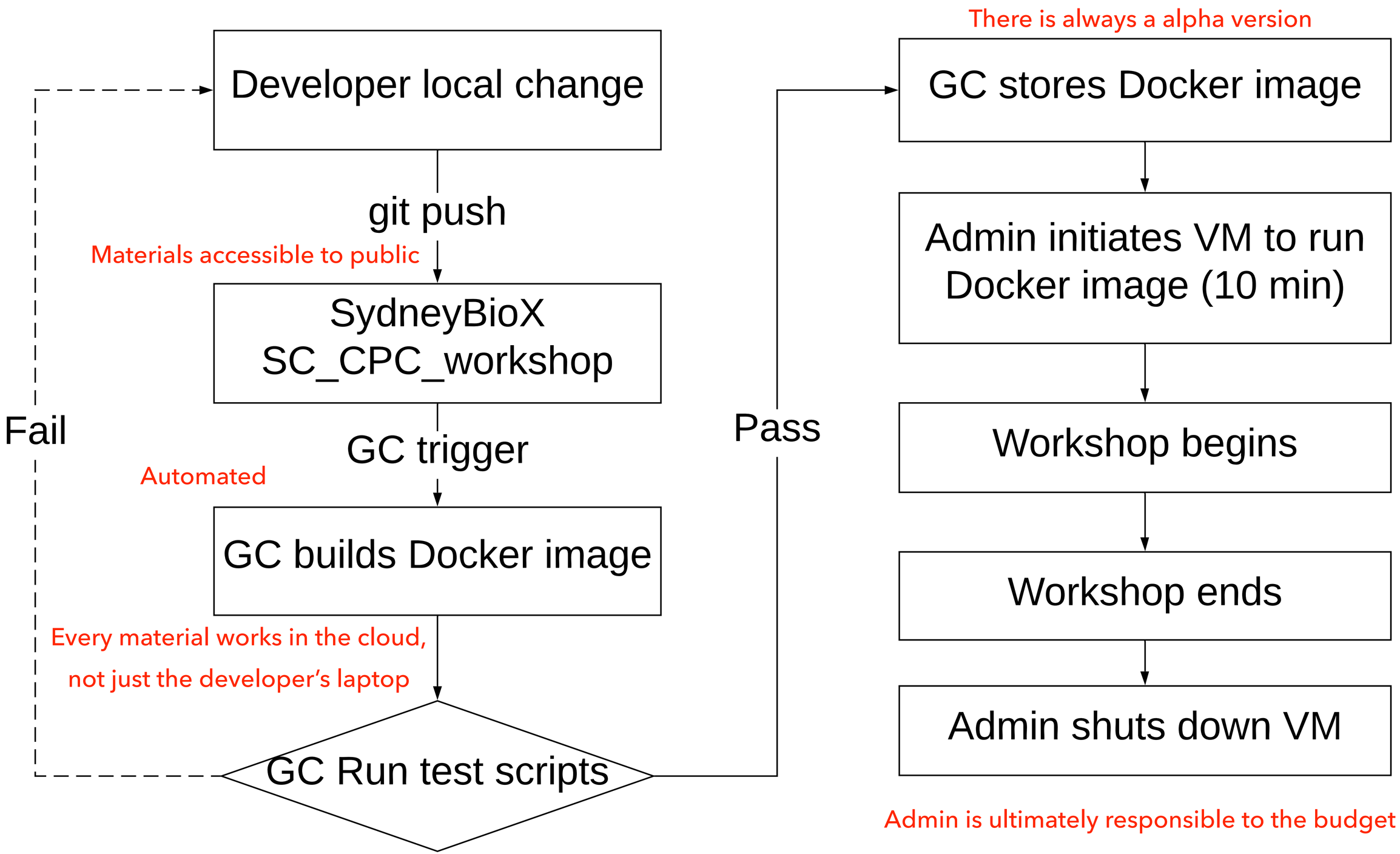
https://github.com/SydneyBioX/SC_CPC_workshop

- All teaching materials are identical to https://github.com/SydneyBioX/SingleCellPlus, except:
  - `Dockerfile` and other set-up scripts
  - `deployment` folder with all the passwords
  - `.gitignore` (to prevent public access to sensitive information)

Google Cloud is a collection of services

- Our GC account currently hosts
  - data.zip (Storage Bucket)
  - Docker images (Container Registry)
  - Virtual machines (VM instances)


Developer local change
git push
SydneyBioX
SC_CPC_workshop
GC trigger
GC builds Docker image
GC Run test scripts
Fail
Pass
GC stores Docker image
Admin initiates VM to run
Docker image (10 min)
Workshop begins
Workshop ends
Admin shuts down VM


There is always a alpha version
Developer local change
git push
Materials accessible to public
SydneyBioX
SC_CPC_workshop
GC trigger
Automated
GC builds Docker image
Every material works in the cloud,
not just the developer's laptop
GC Run test scripts
Fail
Pass
GC stores Docker image
Admin initiates VM to run
Docker image (10 min)
Workshop begins
Workshop ends
Admin shuts down VM
Admin is ultimately responsible to the budget

Advantage of this structure

- Community: Multiple developers can commit to GitHub and trigger a GC build
- Task clarity: Developers only need to push*, admin only need to deploy
- Rigorous: Everything that is available in a Docker image must be functional
- Accountable: Admin is solely responsible for deployment and budgeting
- Trackable: Every version is linked to a GitHub commit
- Reliable: Admin always have access to a working copy of Docker + website

*Developers also need to build the website

Specific files and functionalities

Building Docker image

- `Dockerfile` list out build commands:
  - docker pull from Bioconductor Docker image
  - Add and run build scripts into Docker (install.R and docker_setup.sh)
  - Add `docker_test.R`, `user_setup.R` and `omg.R`

`install.R`, `docker_setup.sh` and `docker_test.R`

- `install.R` controls how all R packages should be installed
- `docker_setup.sh` does two things:
  - git pull from SydneyBioX/SC_CPC_workshop
  - download data.zip from https://storage.googleapis.com/scp_data/data.zip
- `docker_test.R` renders the main RMarkdown files to check if they can be successfully compiled

`user_setup.R` and `omg.R`

- When the workshop begins, attendees should type `source("/home/user_setup.R")` into the R console.
- This copies all the files needed to run that workshop.
- Anything goes wrong during the workshop
  - Minor: ask attendees to fix it themselves
  - Major: type `source("/home/omg.R")` into the console (this is the last resort)

Initiating the workshop

- `deployment/password.Rmd` uses
  - number of users
  - number of virtual machines needed

to generate passwords and store those as `deployment/users.csv`. Print and distribute to attendees.

- `deployment/GCE.R` must be ran by a person authorised to `scpworkshop` GC account. Basic configurations (CPU, RAM or HD) are needed.