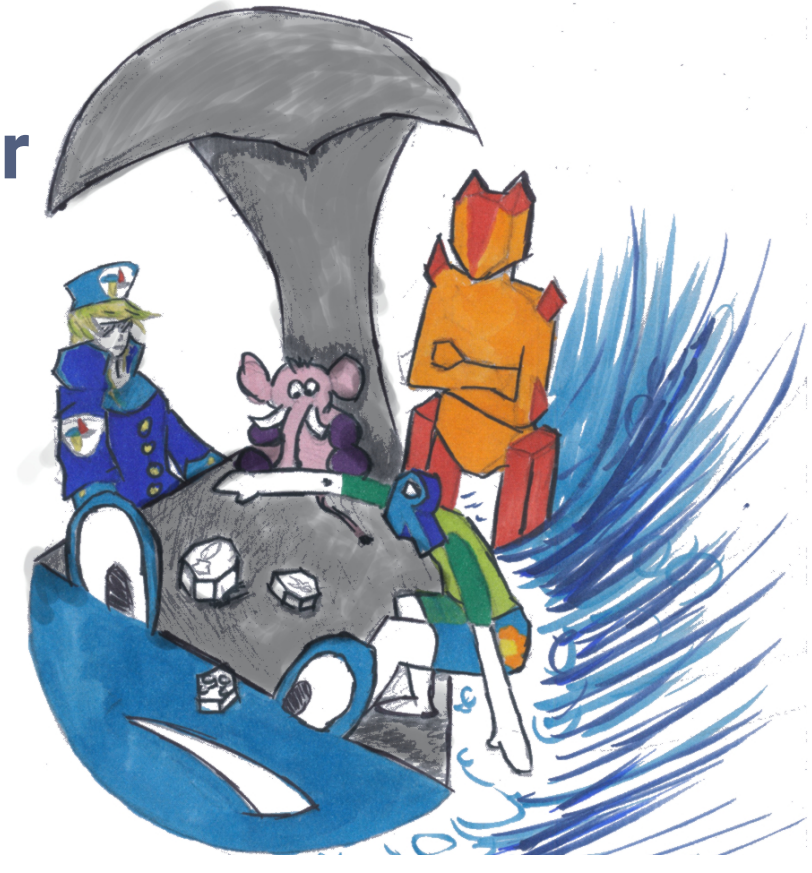
From Airflow to Docker

R in an Open Source Production Environment

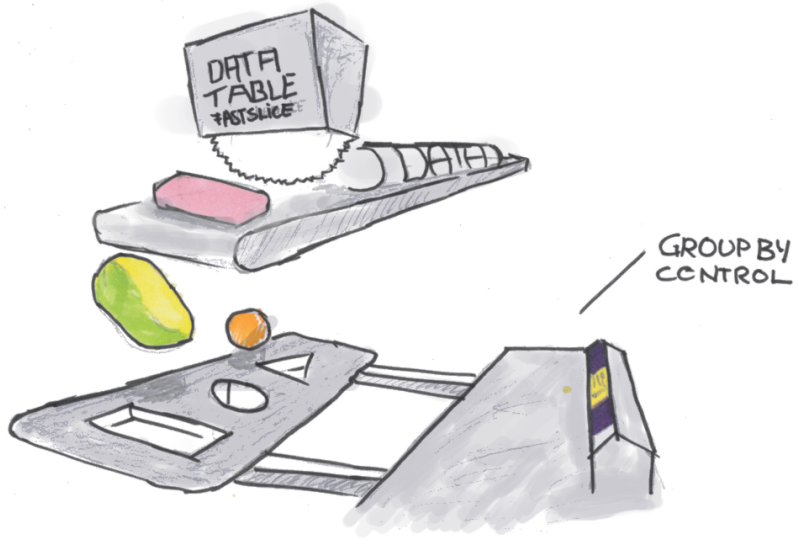
Matt Bannert (@whatsgoodio)

KOF, ETH Zurich

December 3, 2020



What is Production?

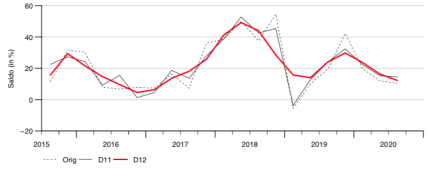


Repeated (regular) runs of the same task resulting in a pre-defined, quality controlled result.

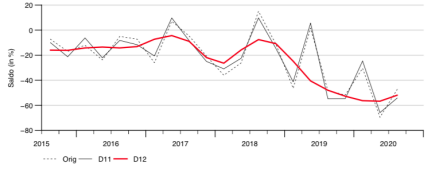
--- KOF definition of production

KOF Output

Noga 3-Steller 462 (Grosshandel mit landwirtschaftlichen Grundstoffen und lebenden Tieren)
Einkaufspreise, erwartete Änderung nächste 3 Monate



Noga 3-Steller 462 (Grosshandel mit landwirtschaftlichen Grundstoffen und lebenden Tieren)
Ertragslage, Änderung letzte 3 Monate



ETH zürich

KOF Konjunkturforschungsstelle

Industrie

Konjunkturumfrage
Enquête conjoncturelle

Moderater Anstieg von Nachfrage und Produktion erwartet

Hausse modérée de la demande et de la production attendue

Februar / Février 2020

Monatsumfrage / Enquête mensuelle

Noga 3-Steller 461 (Handelsvermittlung)
Nachfrage, erwartete Änderung nächste 3 Monate

Periode	Anteil +			Anteil =			Anteil -			Saldo		
	Orig	D11	D12	Orig	D11	D12	Orig	D11	D12	Orig	D11	D12
2019 Q3	22.0	27.7	27.1	69.5	67.0	66.2	8.5	8.1	9.3	13.6	16.5	16.7
2019 Q4	33.9	28.8	25.9	57.6	58.0	62.0	8.5	13.1	12.2	25.4	15.8	9.4
2020 Q1	0.0	-0.9	19.1	78.7	74.3	57.1	21.3	21.8	24.4	-21.3	-19.0	-12.0
2020 Q2	16.0	16.1	12.3	38.0	44.7	52.2	46.0	41.3	37.7	-30.0	-25.9	-30.0
2020 Q3	0.0	6.2	8.8	57.8	54.8	53.6	42.2	42.0	43.5	-42.2	-38.7	-37.5

Noga 3-Steller 461 (Handelsvermittlung)
Verkaufspreise, erwartete Änderung nächste 3 Monate

Periode	Anteil +			Anteil =			Anteil -			Saldo		
	Orig	D11	D12	Orig	D11	D12	Orig	D11	D12	Orig	D11	D12
2019 Q3	0.0	9.6	10.8	71.2	63.5	66.9	28.8	26.3	26.2	-28.8	-14.8	-10.3
2019 Q4	18.6	9.1	13.6	52.5	67.8	68.6	28.8	24.2	21.6	-10.2	-15.7	-4.7
2020 Q1	22.0	20.5	19.4	72.0	74.5	65.0	6.0	9.9	12.4	16.0	13.5	3.3
2020 Q2	30.0	30.1	21.4	62.0	52.2	63.0	8.0	11.9	13.8	22.0	14.2	5.9
2020 Q3	0.0	11.9	17.4	71.1	63.2	65.2	28.9	25.7	20.8	-28.9	-12.1	0.7

Noga 3-Steller 461 (Handelsvermittlung)
Wettbewerbsposition, Änderung letzte 3 Monate

Periode	Anteil +			Anteil =			Anteil -			Saldo		
	Orig	D11	D12	Orig	D11	D12	Orig	D11	D12	Orig	D11	D12
2019 Q3	20.3	22.8	22.4	57.6	53.6	57.5	22.0	25.9	20.9	-1.7	-6.5	0.1
2019 Q4	20.3	17.9	15.0	57.6	61.6	61.8	22.0	24.1	26.4	-1.7	-4.1	-10.7
2020 Q1	0.0	-1.4	7.4	68.0	71.8	68.2	32.0	28.4	29.4	-32.0	-28.4	-26.3
2020 Q2	10.0	11.2	4.8	42.0	38.0	69.2	48.0	45.8	27.3	-38.0	-34.7	-29.1
2020 Q3	0.0	2.8	7.3	97.8	93.8	70.0	2.2	6.0	14.4	-2.2	-6.5	-13.0

Docker

Apache Airflow

Gitlab CI

Docker in One Slide

Single purpose, application focused virtualization.

- **Images**: blueprints for **containers**.
- **Registries**: store images.
- **Docker files** are text based configs from which images are created.
- Images can be stacked, so we can build on existing images.

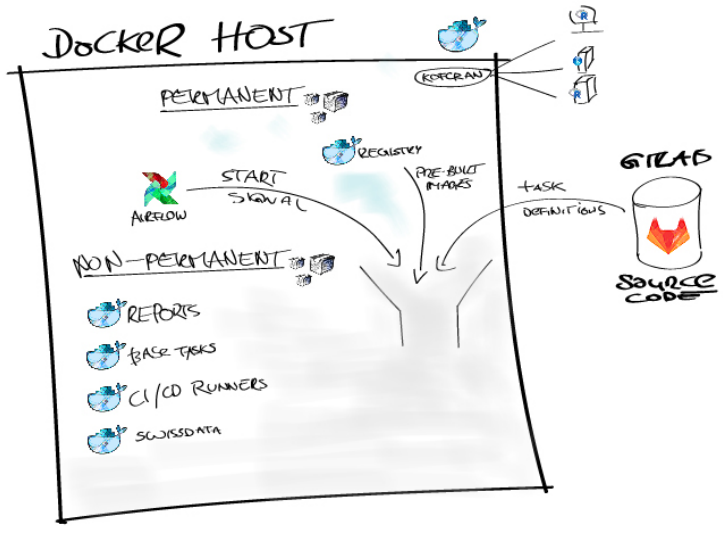
KOF Main Components: Docker Containers

permanent containers:

- PostgreSQL
- Apache Airflow
- miniCRAN
- custom docker registry
- APIs (plumber, servr, express.io)

non-permanent containers:

- a universal base image
- task specific images



KOF Main Components: Gitlab CI

on push to default branch

- build (R) packages
- push to miniCRAN
- deploy to servers
- push files to docker host / volumes (rsync)
- (rebuild docker images)

KOF Main Components: Gitlab CI

on push to default branch

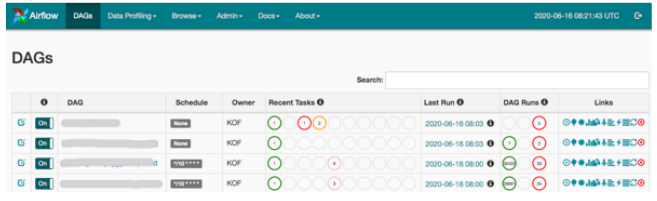
- build (R) packages
- push to miniCRAN
- deploy to servers
- push files to docker host / volumes (rsync)
- (rebuild docker images)

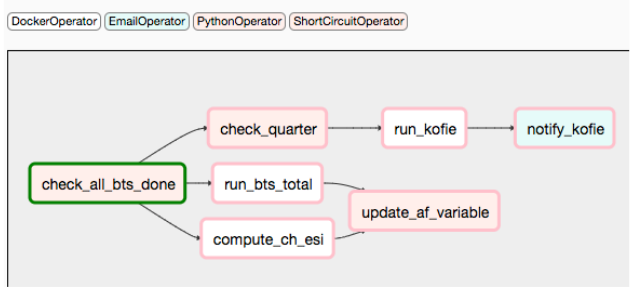
.gitlab-ci.yaml

```
[...]

stages:
  - buildncheck
  - kran
  - deploy

check:
  image: rocker/tidyverse:3.6.0
  stage: buildncheck
  artifacts:
    untracked: true
  script:
    # we don't need it and it causes a hidden file NOTE
    - rm .gitlab-ci.yml
    - R -e 'devtools::install(".", repos = "miniCRAN")'
    - R -e 'devtools::check(error_on = "error")'
    - R CMD build . --no-build-vignettes --no-manual

[...]

- ssh -t gitlabci@someserver.com 'sudo /usr/bin/R -e \\
  "install.packages(\"thatpack\", repos = \"miniCRAN\")"'
```

KOF Main Components: Apache Airflow

- SO questions tagged airflow: 4K+
- workflow scheduler
- monitor / overview dashboard
- can trigger processes locally, on VMs, docker, Kubernetes etc.





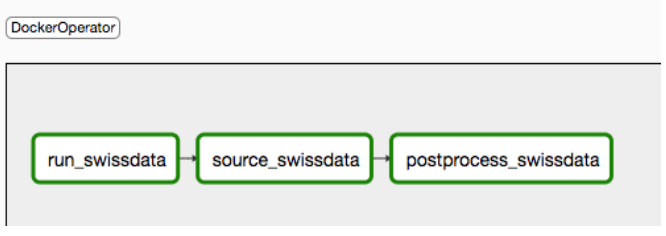
Applied Example: The swissdata project

Gather data from various public sources,
transform into homogeneous time series format,
read into PostgreSQL database.

--- the swissdata project in a nutshell.

Examples: The Swissdata Project

Directed Acyclic Graphs (DAG)




```
[...]

dag = DAG('swissdata', description='Run swissdata',
          schedule_interval = '0 9,17 * * *',
          default_args = default_args, catchup=False)

with dag:
    run_swissdata = DockerOperator(
        task_id = 'run_swissdata',
        image = 'some-docker-registry.ch/swissdata:0.1.0',
        api_version = 'auto',
        auto_remove = True,
        force_pull = True,
        volumes = [
            'swissdata-output:/output'
        ],
        docker_url = "unix:///var/run/docker.sock"
    )

    source_swissdata = DockerOperator(
        task_id = 'source_swissdata',
        image = 'some-docker-registry.ch/kofbase:0.1.0',
        user='some-base-user',
        api_version = 'auto',
        auto_remove = True,
        force_pull = True,
        volumes = [
            'swissdata-output:/swissdata',
            'kofbase-tasks:/tasks'
        ],
        docker_url = 'unix:///var/run/docker.sock',
        environment = {
            'PG_PASSWORD': pg_password
        },
        command = 'source_swissdata'
    )

    postprocess_swissdata = DockerOperator(

    ...

    )

run_swissdata.set_downstream(source_swissdata)
source_swissdata.set_downstream(postprocess_swissdata)
```

Task Specific Image

```
run_swissdata = DockerOperator(
        task_id = 'run_swissdata',
        image = 'some-docker-registry.ch/swissdata:0.1.0',
        api_version = 'auto',
        auto_remove = True,
        force_pull = True,
        volumes = [
            'swissdata-output:/output'
        ],
        docker_url = "unix:///var/run/docker.sock"
    )
```

Standard Image and a Task File

```
source_swissdata = DockerOperator(
        task_id = 'source_swissdata',
        image = 'some-docker-registry.ch/kofbase:0.1.0',
        user='some-base-user',
        api_version = 'auto',
        auto_remove = True,
        force_pull = True,
        volumes = [
            'swissdata-output:/swissdata',
            'kofbase-tasks:/tasks'
        ],
        docker_url = 'unix:///var/run/docker.sock',
        environment = {
            'PG_PASSWORD': pg_password
        },
        command = 'source_swissdata'
    )
```

The Role of R

What Does R Do in This?

- miniCRAN + R packages take care of dependencies
- Downloading, reading and processing data using packages such as *{readxl}*, *{httr}*, *{rvest}*, *{xml2}*, *{jsonlite}*, *{pxR}*, *{tsbox}*, *{yaml}*, ...
- R is used as database interface layer *{timeseriesdb}*

The Role of R

What Does R Do in This?

- miniCRAN + R packages take care of dependencies
- Downloading, reading and processing data using packages such as *{readxl}*, *{httr}*, *{rvest}*, *{xml2}*, *{jsonlite}*, *{pxR}*, *{tsbox}*, *{yaml}*, ...
- R is used as database interface layer *{timeseriesdb}*

Is R any Good at This?

The Role of R

What Does R Do in This?

- miniCRAN + R packages take care of dependencies
- Downloading, reading and processing data using packages such as *{readxl}*, *{httr}*, *{rvest}*, *{xml2}*, *{jsonlite}*, *{pxR}*, *{tsbox}*, *{yaml}*, ...
- R is used as database interface layer *{timeseriesdb}*

Is R any Good at This?

- dependencies are managed reasonably well, weaknesses are mitigated by docker

The Role of R

What Does R Do in This?

- miniCRAN + R packages take care of dependencies
- Downloading, reading and processing data using packages such as *{readxl}*, *{httr}*, *{rvest}*, *{xml2}*, *{jsonlite}*, *{pxR}*, *{tsbox}*, *{yaml}*, ...
- R is used as database interface layer *{timeseriesdb}*

Is R any Good at This?

- dependencies are managed reasonably well, weaknesses are mitigated by docker
- R interfaces incredibly well, there is an R package for everything (15K+ and counting).

The Role of R

What Does R Do in This?

- miniCRAN + R packages take care of dependencies
- Downloading, reading and processing data using packages such as *{readxl}*, *{httr}*, *{rvest}*, *{xml2}*, *{jsonlite}*, *{pxR}*, *{tsbox}*, *{yaml}*, ...
- R is used as database interface layer *{timeseriesdb}*

Is R any Good at This?

- dependencies are managed reasonably well, weaknesses are mitigated by docker
- R interfaces incredibly well, there is an R package for everything (15K+ and counting).
- R is inclusive: it's interpreted. It runs on any major OS. Amazing resources to reach carpentry level. It's free.

Resources

- Rocker Project
- Apache Airflow
- miniCRAN
- gitlab CI

Stay in Touch!

@whatsgoodio: https://twitter.com/whatsgoodio

Email: bannert [at] kof.ethz.ch

see you @user2021global (July 2021)