Karlsruhe Institute of Technology

DevOps for Machine Learning in Academia

A personal view based on DEEP Hybrid DataCloud experience and HEAT project

Valentin Kozlov

STEINBUCH CENTRE FOR COMPUTING, KIT

GridKa School 2019 - The Art of Data

26-30 August 2019
KIT, Campus North, FTU

source: en.wikipedia.org

Outline

- What is DevOps?
- Why DevOps in Academia?
- DevOps techniques
- Examples:
  - DEEP Hybrid DataCloud
  - HeAT
- Summary

What is DevOps?

DevOps is *a set of software development practices* that combine *software development* (*Dev*) and *information technology operations* (*Ops*) to ***shorten*** the systems development ***life cycle*** while delivering features, fixes, and updates ***frequently*** in close alignment with ***business objectives***.

… Academics and practitioners have not developed a unique definition for the term "DevOps"

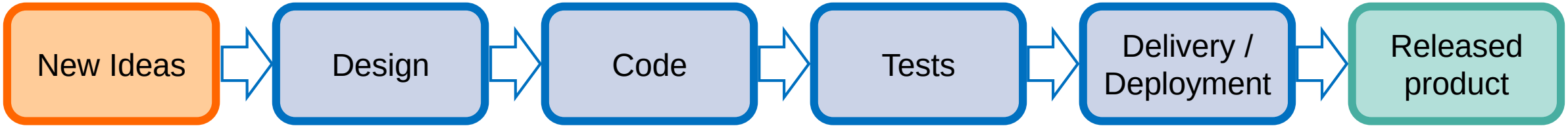
ATLASSIAN

DevOps is a *set of practices* that ***automates*** the processes between *software development* and *IT teams*, in order that they can *build*, *test*, and *release* software ***faster*** and more ***reliably***.

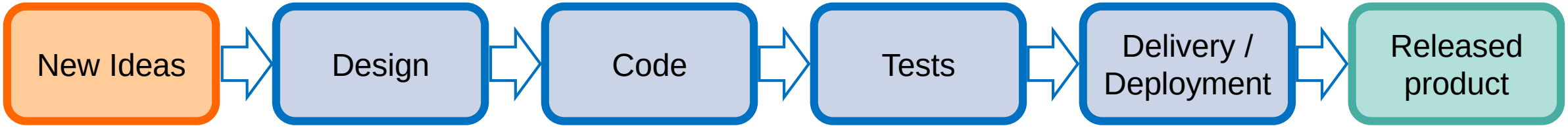
Keywords:

- ***set of practices***
- *release software* ***faster*** and ***reliably***
- ***automation*** *of the process*

Product pipeline



Product pipeline



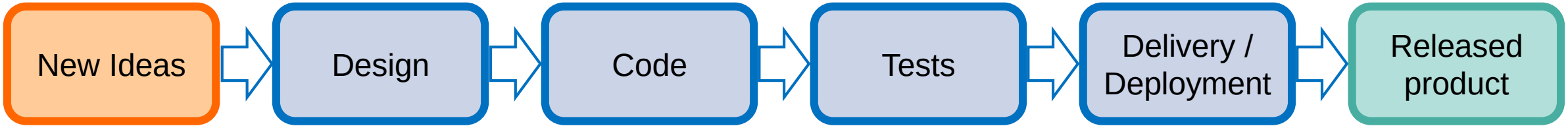
Product manager

Developer

Tester

Ops

Product pipeline



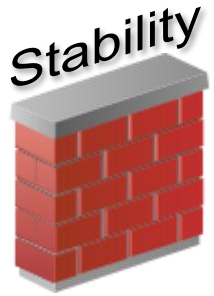
Product manager

Developer

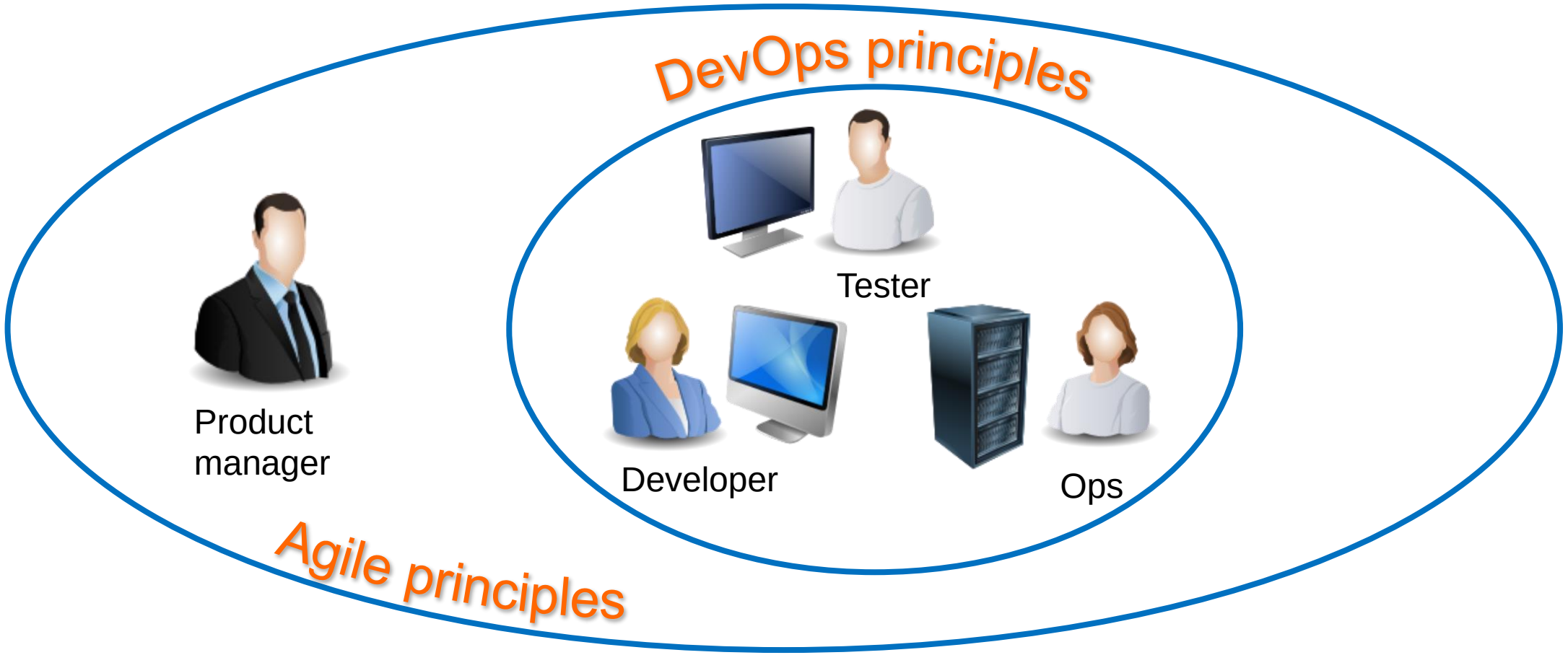
Tester

Ops

vs.

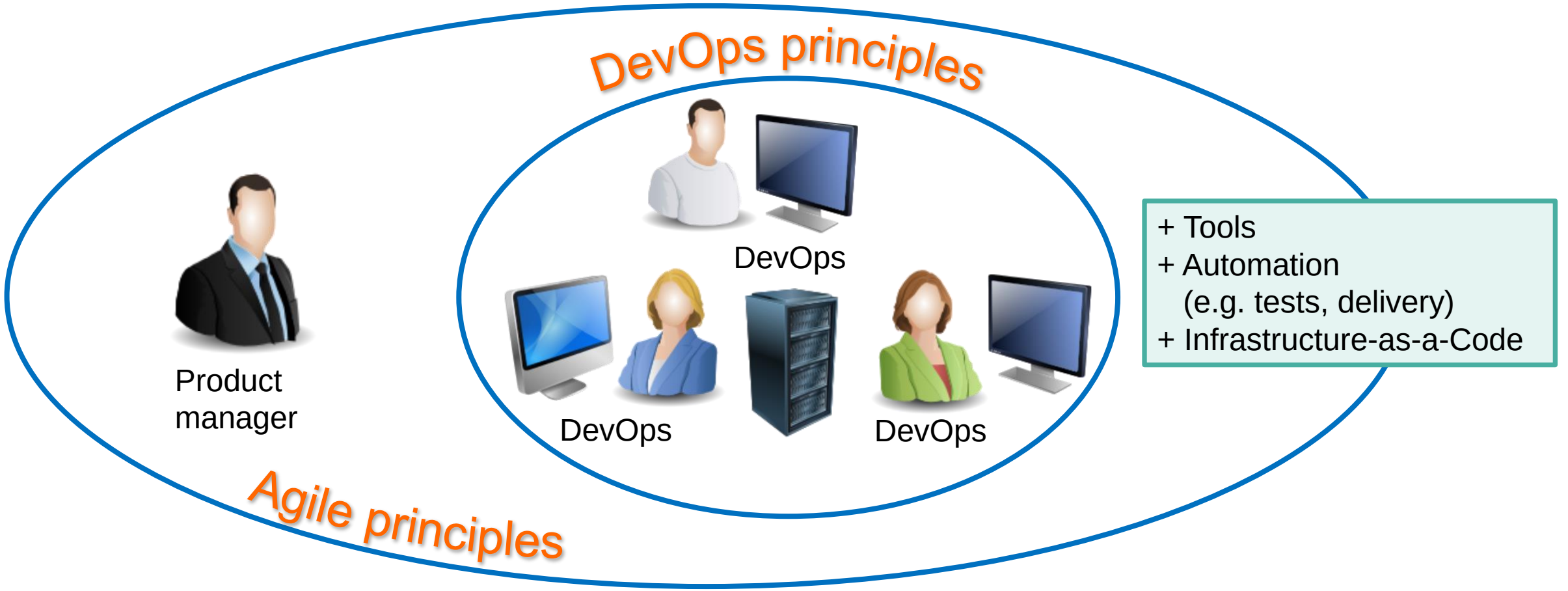


Dev +Tests + Ops → Work together



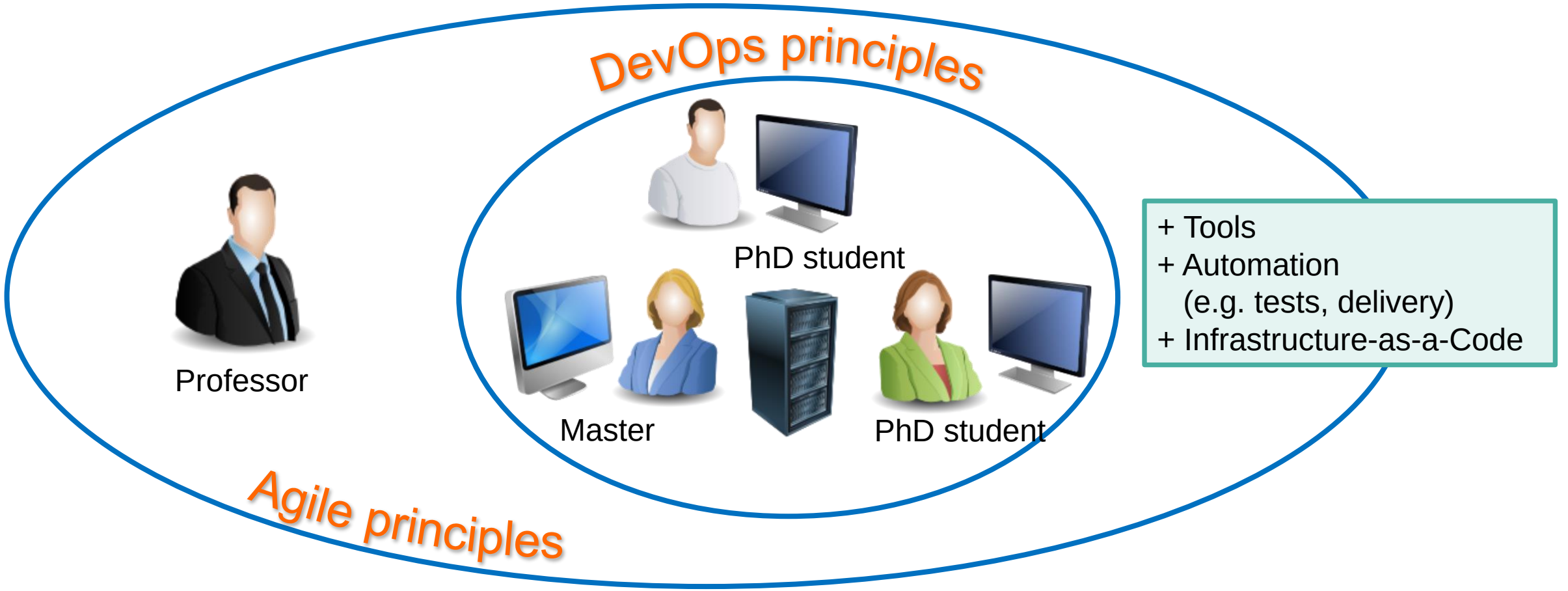
Iterative developments with a releasable product *at every iteration*

Dev +Tests + Ops → Work together



Iterative developments with a releasable product *at every iteration*

Dev +Tests + Ops → Work together



Iterative developments with a releasable product *at every iteration*

Why DevOps for Scientific Development?

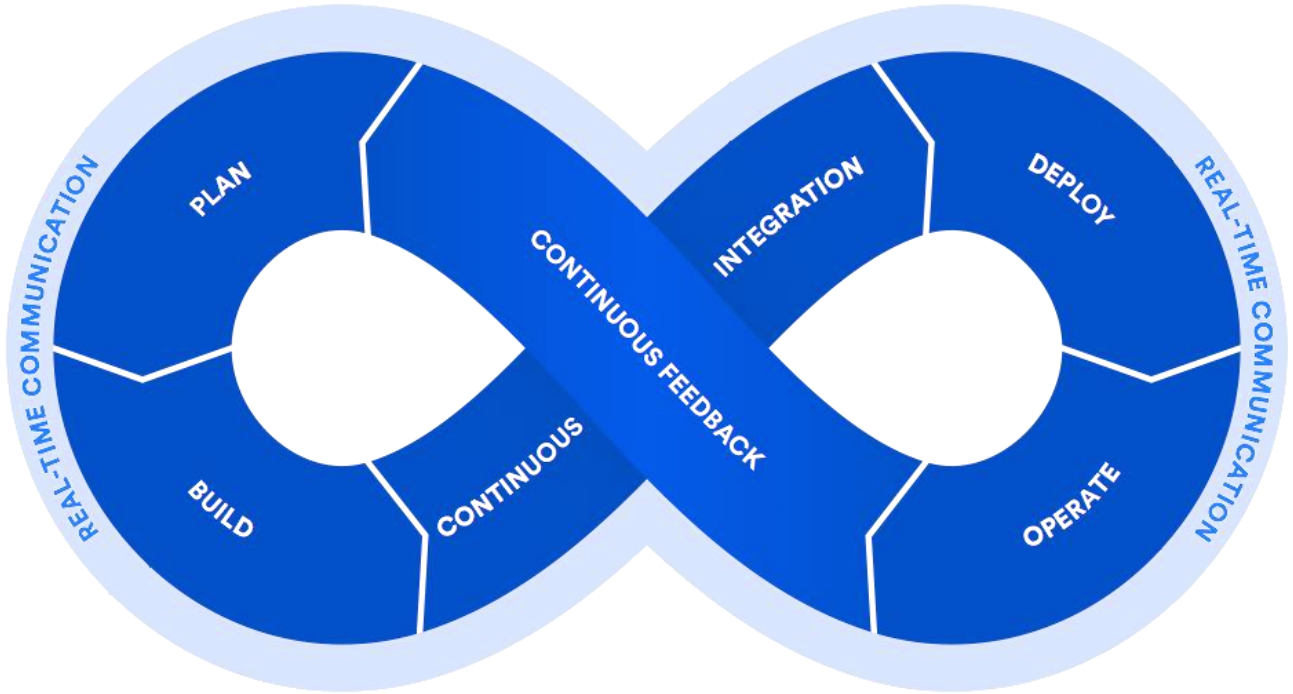
DevOps as

- a *set of practices*
- for a *reliable* software to be delivered *faster*
- a '*single person* skills' to maintain the product pipeline

this is … what we often do/need in science:

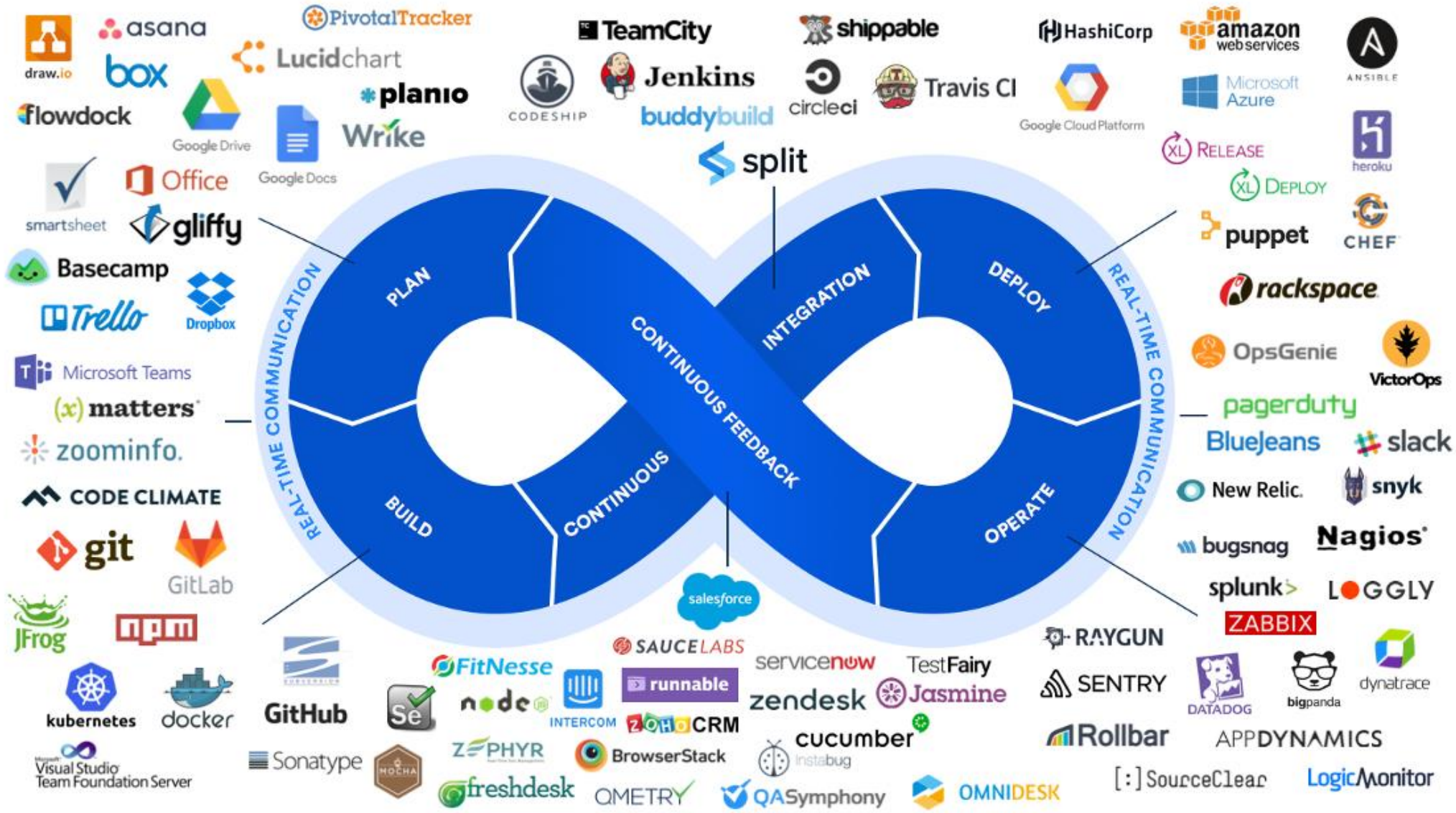
DevOps		Science
Faster	→	Art of experimentation
Reliability of software	→	Reliability of software ☺
'Single person' skills (supported by DevOps tools)	→	• e.g. a master / PhD project • Often a domain knowledge is required to develop and test the code • Overcome bottlenecks where IT ops skills needed
Automation of tests (e.g. unit tests)	→	Better structured and readable code, so that a student project can be continued
Code Quality checkers	→	Maintainable code
Infrastructure as a Code, Containerization	→	Scientific reproducibility

Product pipeline → DevOps loop



source: https://marketplace.atlassian.com/categories/devops?utm_source=wac_marketplace_landing

Marketplace for DevOps (Atlassian)

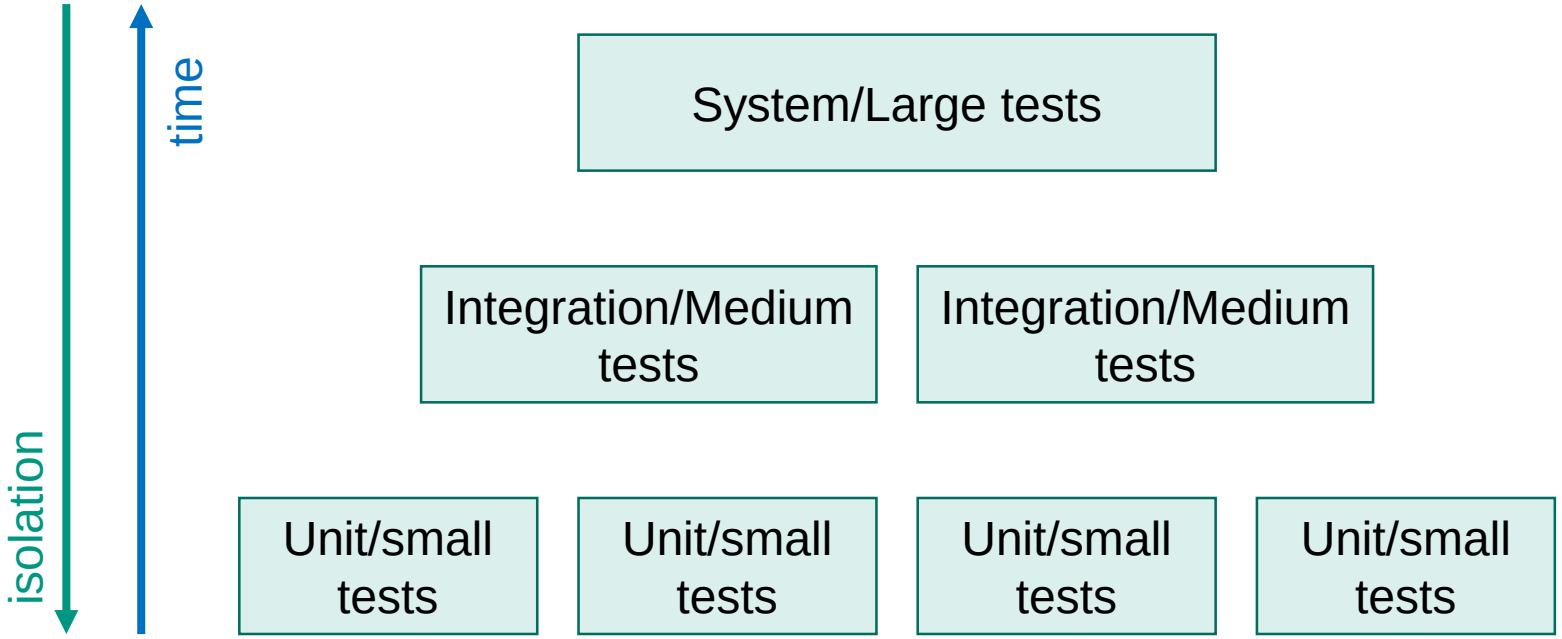


source: https://marketplace.atlassian.com/categories/devops?utm_source=wac_marketplace_landing

DevOps: key techniques

	Technique	Examples
Plan	Tools to document your plans (Agile user stories) and design	Google docs, Wikipages
Build	Version Control Test stack (Test coverage)	GitHub
CI / CD	Continuous Integration (CI) : Code merged in repository → Automated builds and code tests performed Continuous Delivery (CD) : CI success → Deploy in testing environments and perform integration tests Continuous Deployment (CD'): CD success → Deploy to production environment (e.g. as a stable release)	Travis CI, Jenkins
Deploy	Cloud: Infrastructure as a code Docker containers	Puppet, Ansible udocker (HPC)
Ope-rate	Tools to monitor your application, infrastructure	Zabbix

Test stack pyramid



e.g. System performance tests, Large size, even longer to execute, even less (10% *)

Testing a combination of functions, Medium size, longer to execute, less tests (20% *)

Testing a single function. Small, fast to execute, many (70% *)

*) As a good first guess at Google: https://testing.googleblog.com/2015/04/just-say-no-to-more-end-to-end-tests.html

Unit testing in ML

- https://medium.com/@keeper6928/how-to-unit-test-machine-learning-code-57cf6fd81765
- Machine learning testing framework for TensorFlow: https://github.com/Thenerdstation/mltest

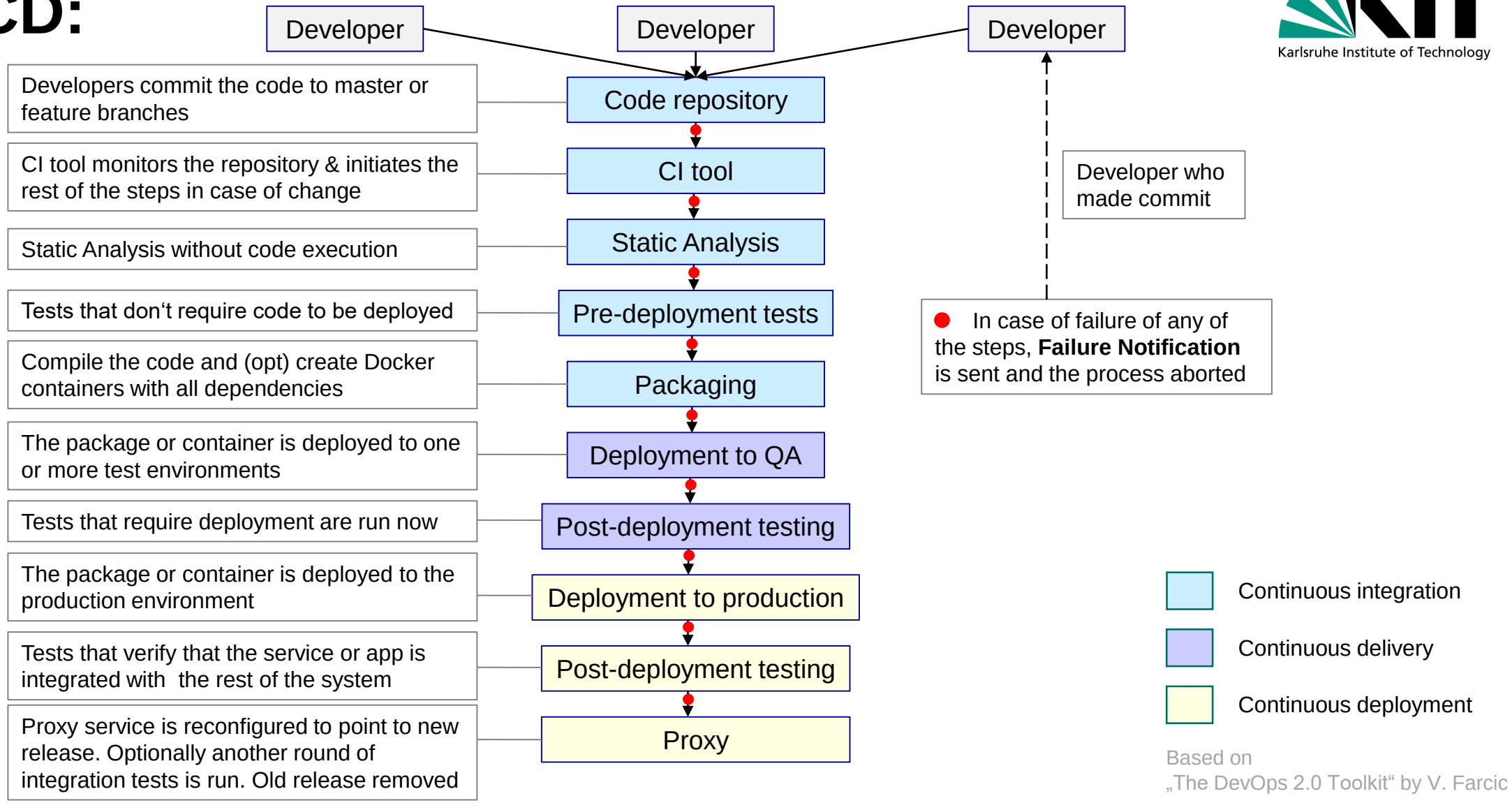
```
def test_convnet():
  image = tf.placeholder(tf.float32, (None, 100, 100, 3)
  model = Model(image)
  sess = tf.Session()
  sess.run(tf.global_variables_initializer())
  before = sess.run(tf.trainable_variables())
  _ = sess.run(model.train, feed_dict={
              image: np.ones((1, 100, 100, 3)),
              })
  after = sess.run(tf.trainable_variables())
  for b, a, n in zip(before, after):
      # Make sure something changed.
      assert (b != a).any()
```

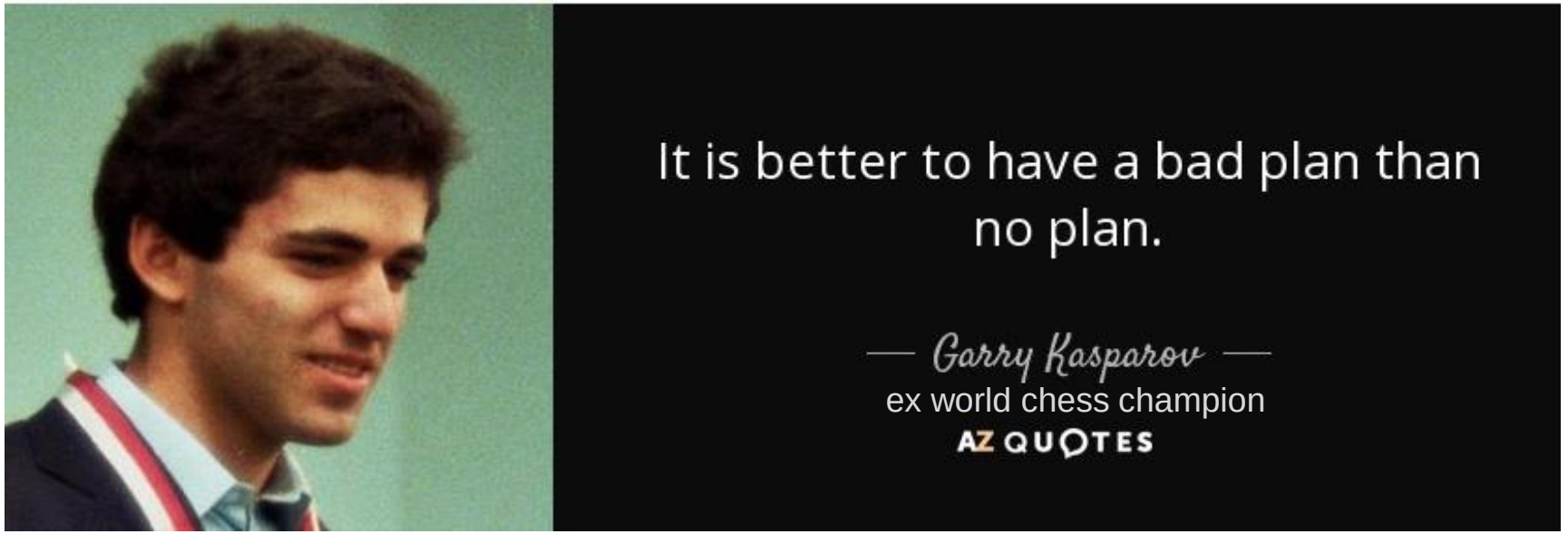
Eliminates:

- Sometimes variables are initialized but neural network layer is not (properly) connected, i.e. corresponding weights will not be updated
- Some layers (functions) have to be set into trainable state

→ You check that all trainable variables are indeed updated after one iteration.

CI / CD:


Developer
Developer
Developer
Developers commit the code to master or feature branches
Code repository
CI tool monitors the repository & initiates the rest of the steps in case of change
CI tool
Static Analysis without code execution
Static Analysis
Tests that don‘t require code to be deployed
Pre-deployment tests
Compile the code and (opt) create Docker containers with all dependencies
Packaging
The package or container is deployed to one or more test environments
Deployment to QA
Tests that require deployment are run now
Post-deployment testing
The package or container is deployed to the production environment
Deployment to production
Tests that verify that the service or app is integrated with the rest of the system
Post-deployment testing
Proxy service is reconfigured to point to new release. Optionally another round of integration tests is run. Old release removed
Proxy
Developer who made commit
In case of failure of any of the steps, **Failure Notification** is sent and the process aborted
Continuous integration
Continuous delivery
Continuous deployment
Based on „The DevOps 2.0 Toolkit“ by V. Farcic

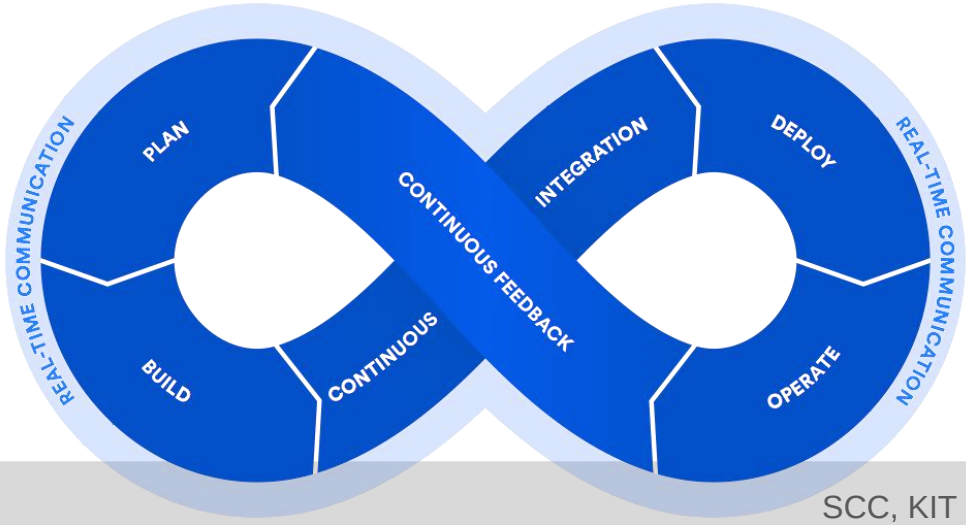


We may rephrase it as ..

It is better to have a simple DevOps pipeline than no pipeline

Advance your pipeline iteratively

- Analyze your current way of software development
- Think what improvement would bring you the most benefit
- Implement



Example #1

DEEP Hybrid DataCloud

https://deep-hybrid-datacloud.eu/

https://marketplace.deep-hybrid-datacloud.eu/

https://docs.deep-hybrid-datacloud.eu

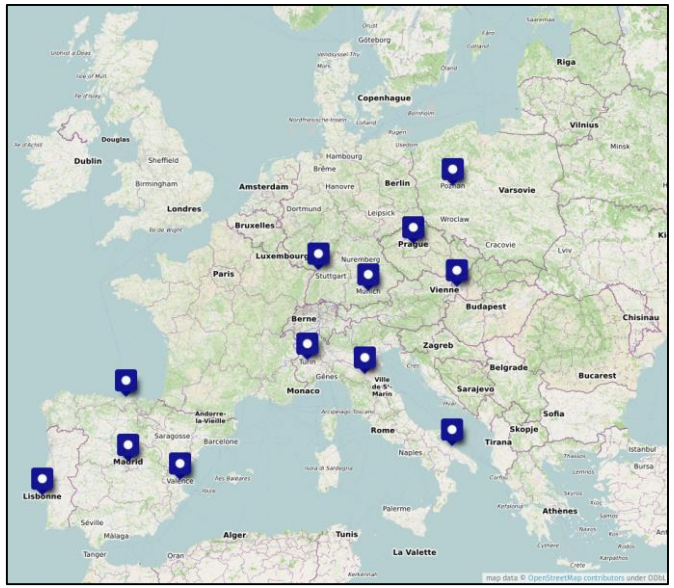
https://www.youtube.com/playlist?list=PLJ9x9Zk1O-J_UZfNO2uWp2pFMmbwLvzXa

DEEP Hybrid DataCloud: context

Karlsruhe Institute of Technology

Designing and **E**nabling **E**-Infrastructures for intensive data **P**rocessing in a Hybrid DataCloud

- Started as a spin-off project (together with XDC) from INDIGO-DataCloud technologies
- H2020 project, EINFRA-21 call
- Runs November 1st 2017 – April 2020
- March 27, 2019 : mid-term Review (Luxembourg)
- 9 academic partners + 1 industrial partner: CSIC, LIP, INFN, PSNC, KIT, UPV, CESNET, IISAS, HMGU; Atos



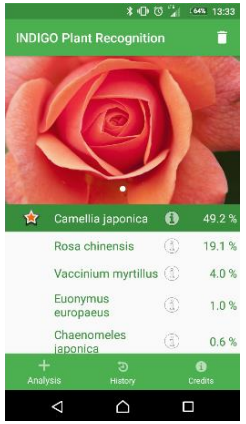
Ease and lower the entry barrier for scientists
in using intensive computing techniques, e.g. deep learning, to exploit very large data sources

- Build ready to use modules and offer them through an open catalog or marketplace (Docker images)
- Implement common software development techniques also for scientist's applications (DevOps)
- Transparent execution on e-Infrastructures (pilot testbed, orchestration)

DEEP pilot use-cases

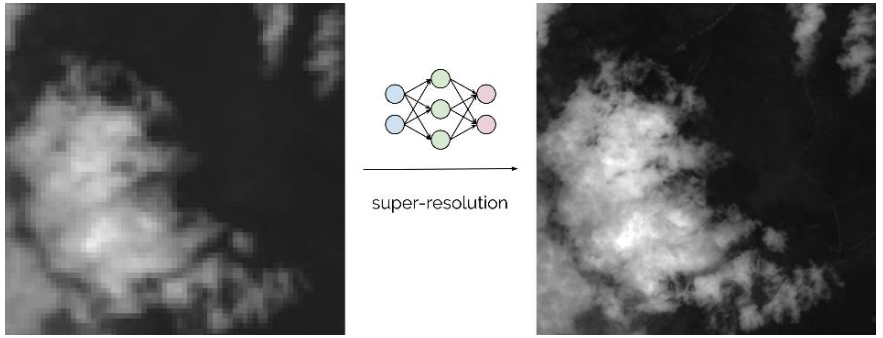
Image Classification

Plants, Conus marine snails, Seeds, Phytoplankton (CNN, TF)



Satellite Imagery

Super-resolution service (e.g. DSen2; TF)



Retinopathy

DL to analyze color fundus retinal photography images (CNN, TF)

DR=0

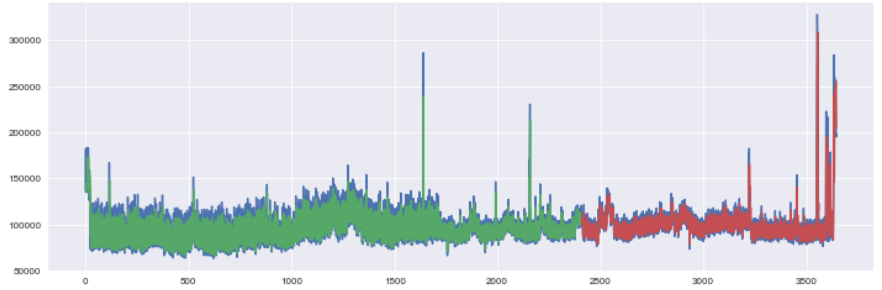
DR=4

DEEP Leading Interaction: dog's breed detection

to test the infrastructure and provide working examples

Massive Online Data Streams: Online analysis of data streams

Intrusion detection systems (LSTM/GRU, TF)



dataset, prediction (train), prediction (test)

DevOps tools from a DEEP user perspective

Karlsruhe Institute of Technology

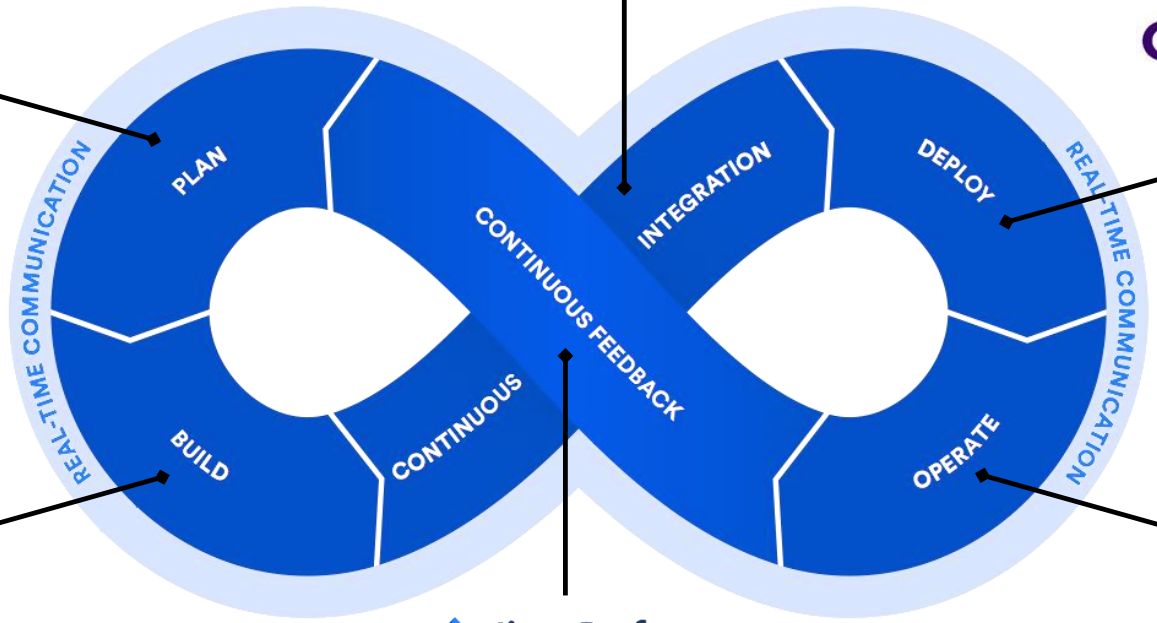
Jenkins
+ jenkins-pipeline-library

Confluence
Jira Software
(user stories)

PaaS Orchestrator
INDIGO - DataCloud
OASIS TOSCA

Alien4Cloud

UDOCKER



DEEP Data Science Template
COOKIECUTTER

DEEPaaS API

metadata.json

GitHub

docker

Jira Software

GitHub
issue tracking

User-defined logging

Data Science Template

1. $ pip install cookiecutter
2. $ cookiecutter https://github.com/indigo-dc/cookiecutter-data-science
3. answer questions
4. Two directories created:
   <user_project> and <DEEP-OC-user_project>
   - This are also git repositories.
   - Both have ‘master’ and ‘test’ branches.
   - Dockerfile, Jenkinsfile, and
     python files necessary for DEEPaaS API integration
     are pre-populated
   - .stestr.conf, tox.ini for testing in Python

<user_project>

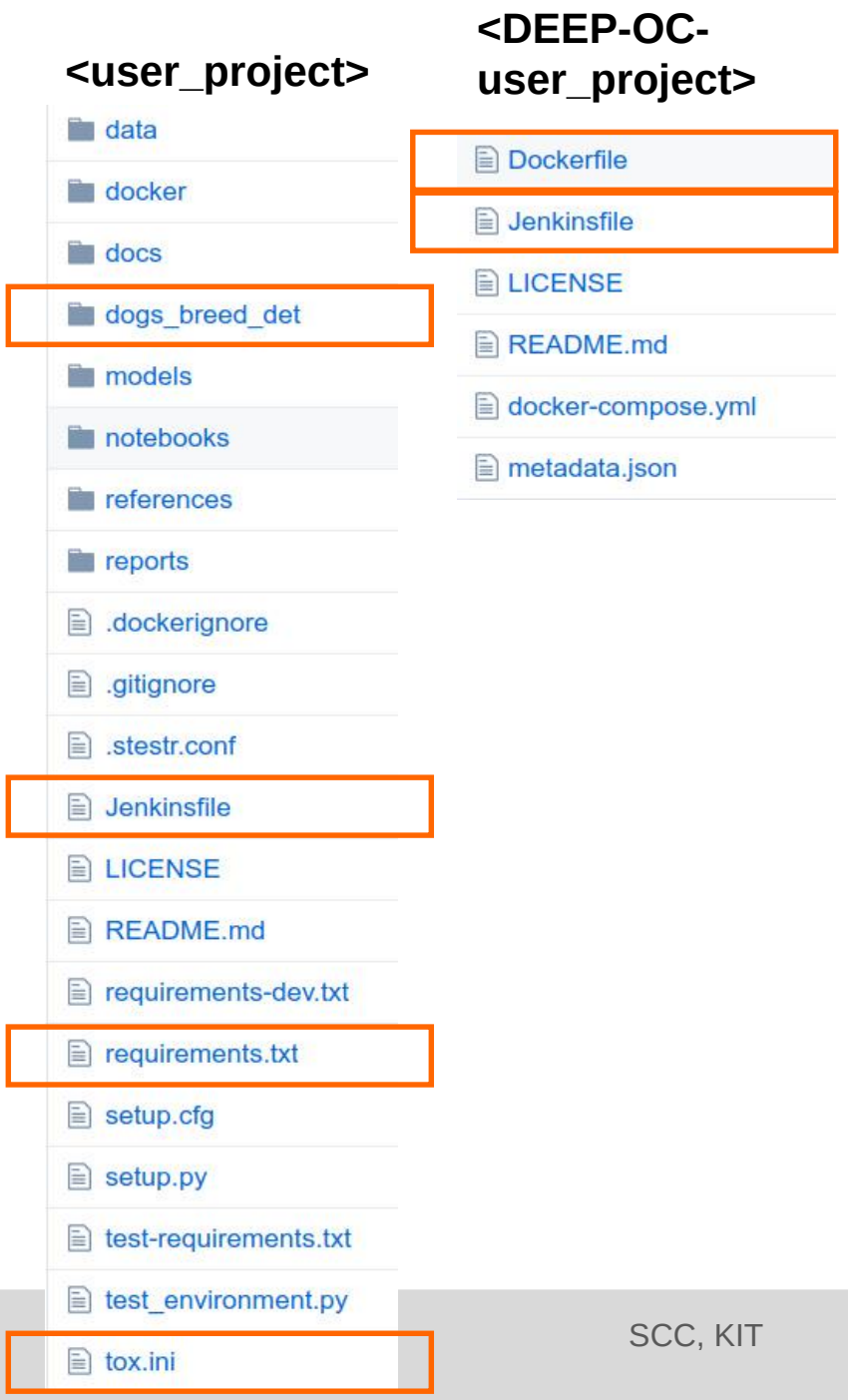
- data
- docker
- docs
- dogs_breed_det
- models
- notebooks
- references
- reports
- .dockerignore
- .gitignore
- .stestr.conf
- Jenkinsfile
- LICENSE
- README.md
- requirements-dev.txt
- requirements.txt
- setup.cfg
- setup.py
- test-requirements.txt
- test_environment.py
- tox.ini

<DEEP-OC-user_project>

- Dockerfile
- Jenkinsfile
- LICENSE
- README.md
- docker-compose.yml
- metadata.json

Data Science Template

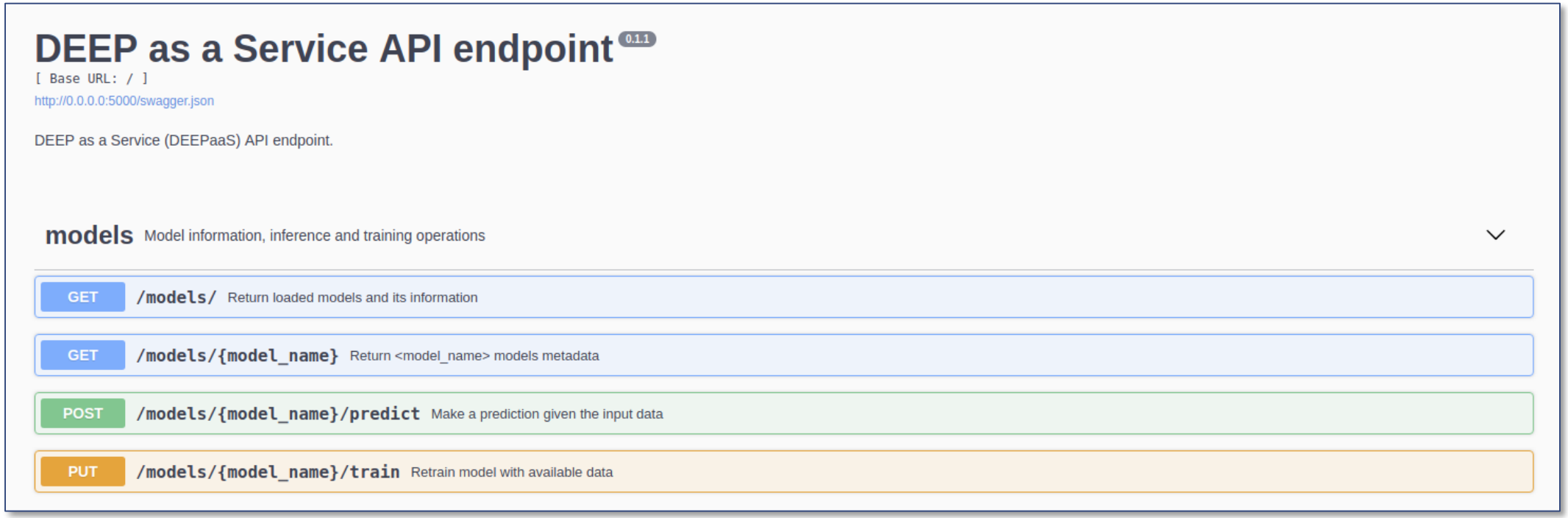
1. $ pip install cookiecutter
2. $ cookiecutter https://github.com/indigo-dc/cookiecutter-data-science
3. answer questions
4. Two directories created:
   <user_project> and <DEEP-OC-user_project>
   - This are also git repositories.
   - Both have ‘master’ and ‘test’ branches.
   - Dockerfile, Jenkinsfile, and
     python files necessary for DEEPaaS API integration
     are pre-populated
   - .stestr.conf, tox.ini for testing in Python



DEEPaaS API

get_metadata()	Return metadata from the exposed model
get_train_args(*args)	Function to expose to the API what are the typical training parameters
train(*args)	Function to train your model.

predict_file(path, **kwargs)	Perform a prediction from a file in the local filesystem
predict_data(*args, **kwargs)	Perform a prediction from the data passed in the arguments.
predict_url(*args)	Perform a prediction from a remote URL



Jenkins CI/CD pipeline

PR + approve

Code version control

github.com/deephdc

test branch
master branch

Code testing & Quality Control

Jenkins

flake8 / PEP8

unit tests

Metrics gathering

Bandit security scanner

BANDIT

Docker build & push to registry

Jenkins

docker hub

:cpu-test, :gpu-test
:latest, :cpu, :gpu

DEEP Open Catalog

Jenkins CI/CD reports

Full project name: Pipeline-as-code/DEEP-OC-org/dogs_breed_det/test

- Changes
- Full Stage View
- coverage.py report
- Bandit report
- Open Blue Ocean
- GitHub
- PyLint Warnings
- Coverage Report
- SLOCCount Results
- Embeddable Build Status

Recent Changes

Build History trend

find

#30	Aug 11, 2019 10:32 PM
#29	Aug 11, 2019 10:24 PM
#28	Aug 11, 2019 10:07 PM
#27	Aug 11, 2019 9:47 PM
#26	Aug 11, 2019 9:39 PM
#25	Aug 11, 2019 9:36 PM
#24	Aug 11, 2019 9:28 PM
#23	Aug 11, 2019 5:31 PM
#22	Aug 11, 2019 6:48 AM
#21	Aug 10, 2019 10:23 AM
#20	Aug 9, 2019 10:55 PM
#19	Aug 8, 2019 9:20 PM
#18	Aug 6, 2019 9:03 AM
#17	Aug 5, 2019 10:18 PM
#16	Aug 5, 2019 9:19 PM

Enlarge Configure

Code Coverage

Packages	100%
Files	100%
Classes	100%
Lines	49%
Conditionals	100%

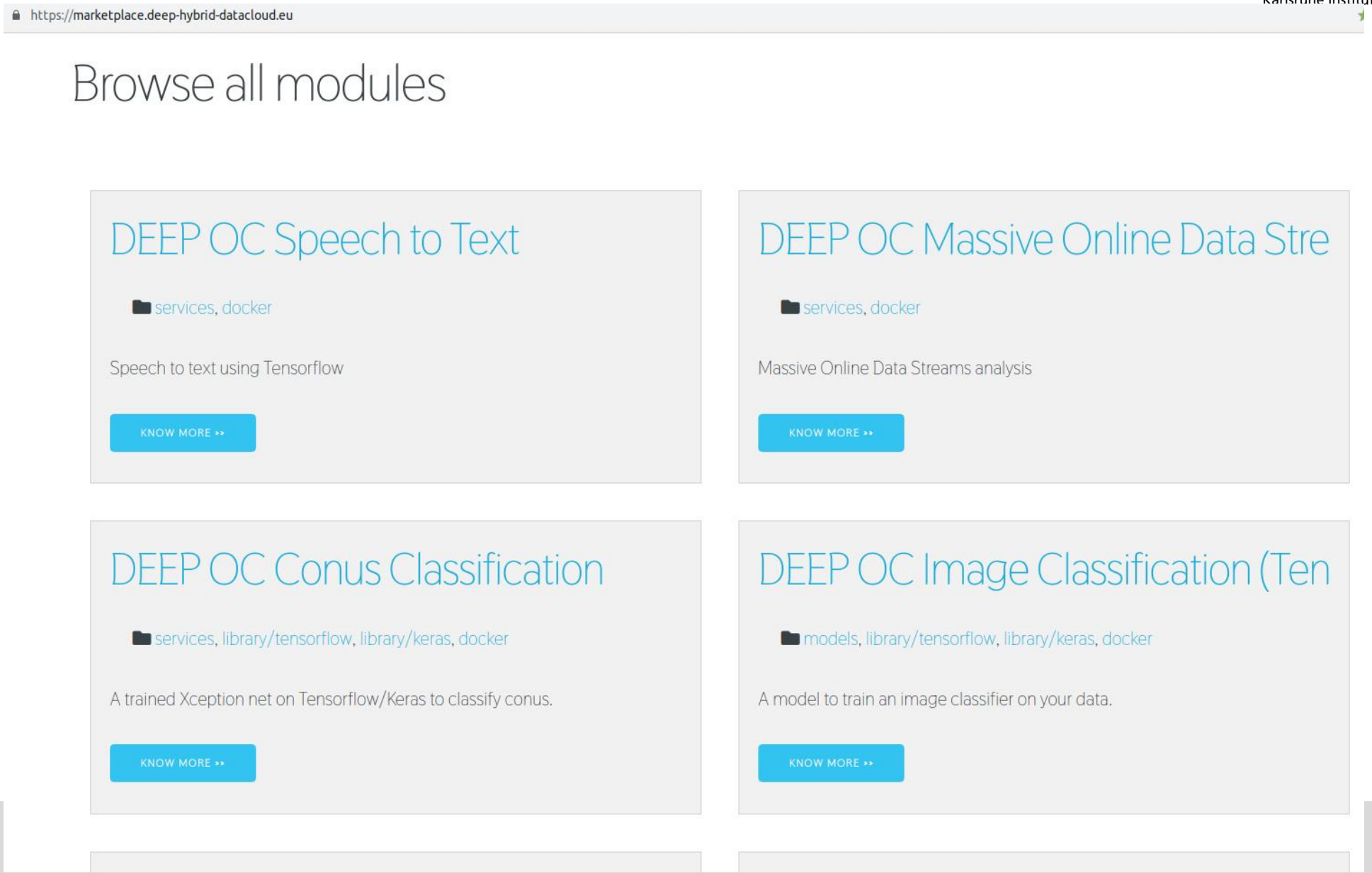
Stage View

Average stage times:
(Average full run time: ~28min 8s)

	Declarative: Checkout SCM	Code fetching	Style analysis: PEP8	Unit testing coverage	Metrics gathering	Security scanner	Re-build Docker images	Declarative: Post Actions
	7s	1s	2min 42s	47s	6s	28s	2min 21s	647ms
#30 Aug 12 00:32 1 commit	10s	3s	3min 24s	3min 38s	45s	3min 16s	16min 26s	570ms
#29 Aug 12 1 commit	7s	1s	3min 23s	33s	349ms	81ms	67ms	551ms

DEEP Open Catalog

https://marketplace.deep-hybrid-datacloud.eu



DEEP Open Catalog

DEEP OC Massive Online Data Streams

GET THIS MODULE | DEEPHDC/DEEP-OC-MODS

By DEEP-Hybrid-DataCloud Consortium | Created: Tue 19 February 2019 - Updated: Wed 14 August 2019

services, docker

License: Apache 2.0

build passing

This use case analyzes online data streams in order to generate alerts with time-bounded constrains and in real-time. The main study is focused on building additional intelligent module using NN and DL techniques in co-function with underlying Intrusion Detection Systems (IDS) supervising traffic networks of compute centers. Preserving old data for historical purposes, security analysts will be able to supervise generated alerts and to enhance cyber security [1, 2] for such centers when large IT infrastructures and devices products a huge amount of data streaming continuously and dynamically.

The principle of the solution is proactive time-series prediction [5] adopting NNs as well as

Run locally on your computer

Using Docker

You can run this module directly on your computer, assuming that you have Docker installed, by following these steps:

Deployment of artefacts

OASIS TOSCA

Topology and Orchestration Specification for Cloud Applications
TOSCA template → Simple Profile in YAML.

Alien 4 Cloud

GUI to compose TOSCA templates, deployment in a cloud

orchent – CLI to perform deployments using TOSCA templates

PaaS Orchestrator Dashboard – Web GUI for deployments using TOSCA templates

HPC environment:

UDOCKER

a basic user tool to execute simple docker containers in user space ***without*** requiring root privileges.

- a docker like CLI
- provides GPU access
- MPI / Infiniband can be used

Example #2

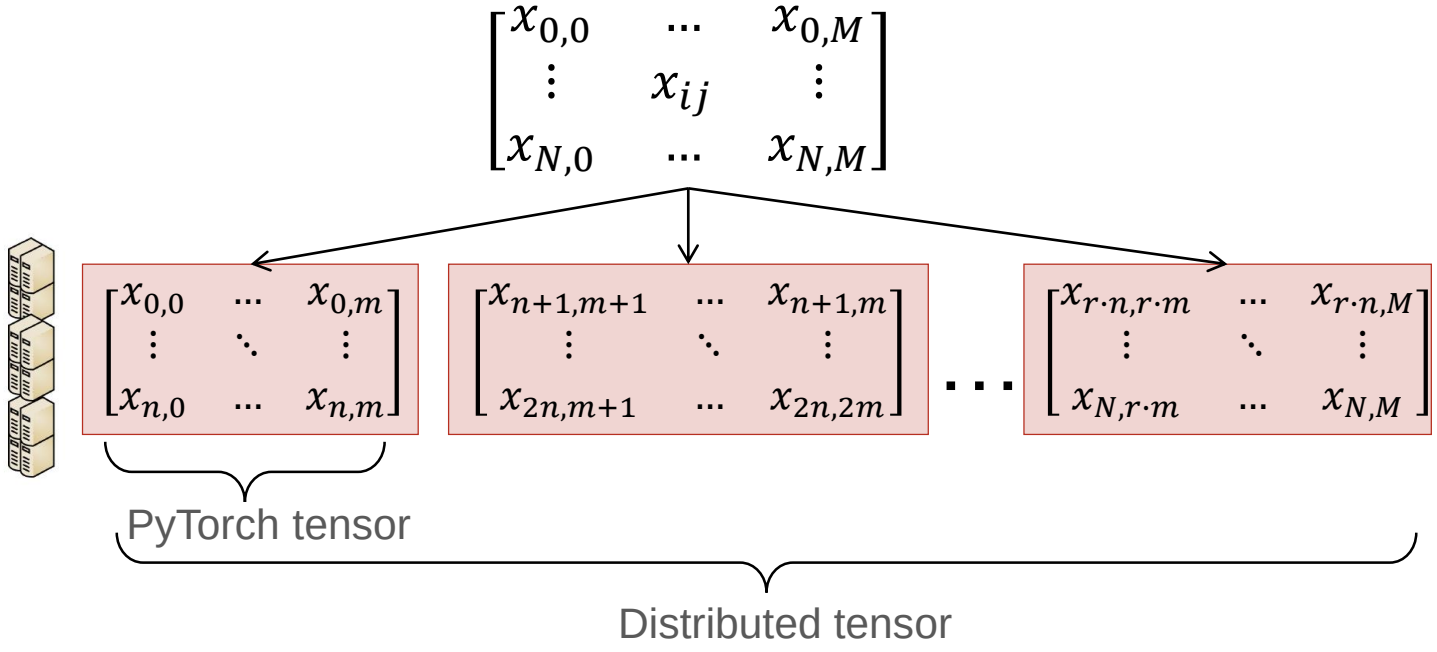
- Helmholtz Analytics Toolkit

https://github.com/helmholtz-analytics/heat/

HeAT

Scientific data analytics library for HPC systems build on top of **PyTorch**

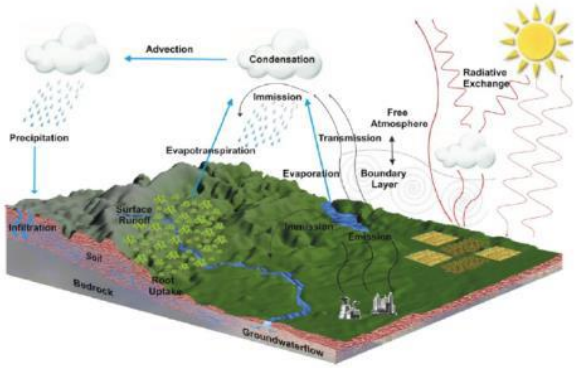
- Operates on heterogeneous hardware like GPU/CPU systems
- Allows computation on ***distributed*** systems
- ***Distributed tensor data object***: operations like basic scalar functions, linear algebra algorithms, slicing or broadcasting operations

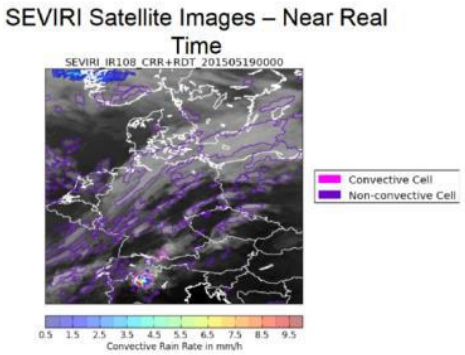


3 Helmholtz centers:

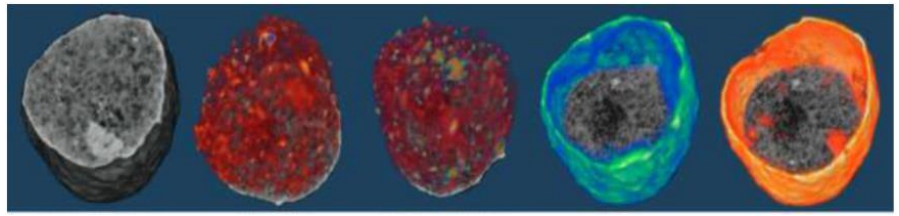
HeAT Use cases

Earth System Modelling

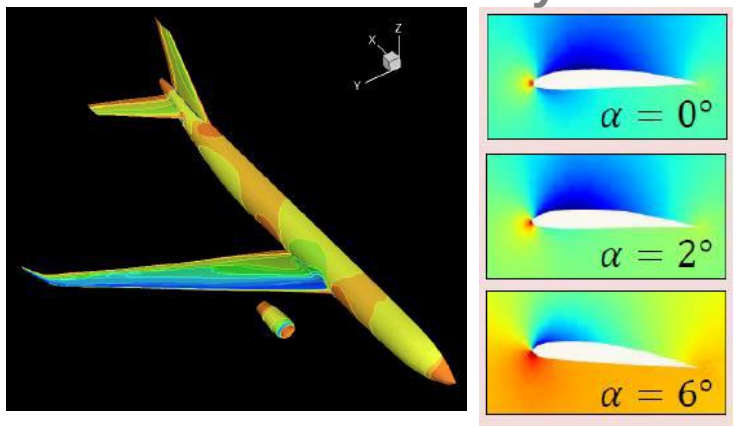




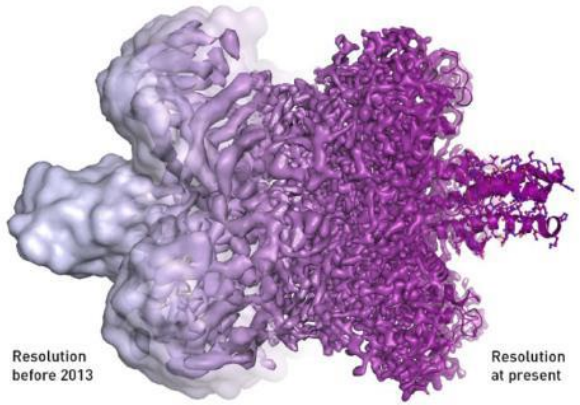
Research with Photons



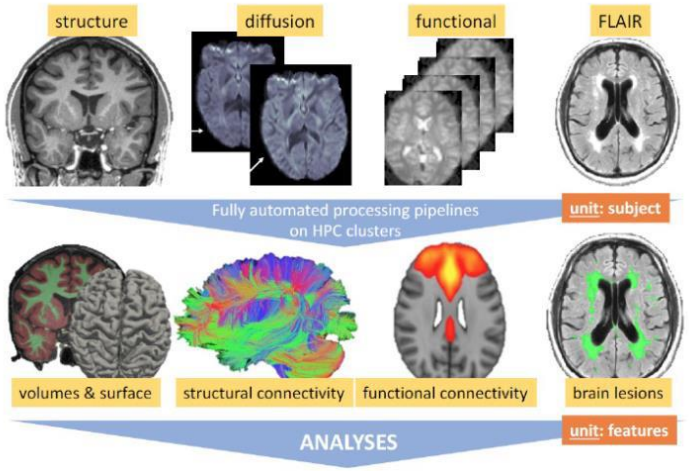
Aeronautics and Aerodynamics



Structural Biology



Neuroscience



HeAT CI/CD pipeline

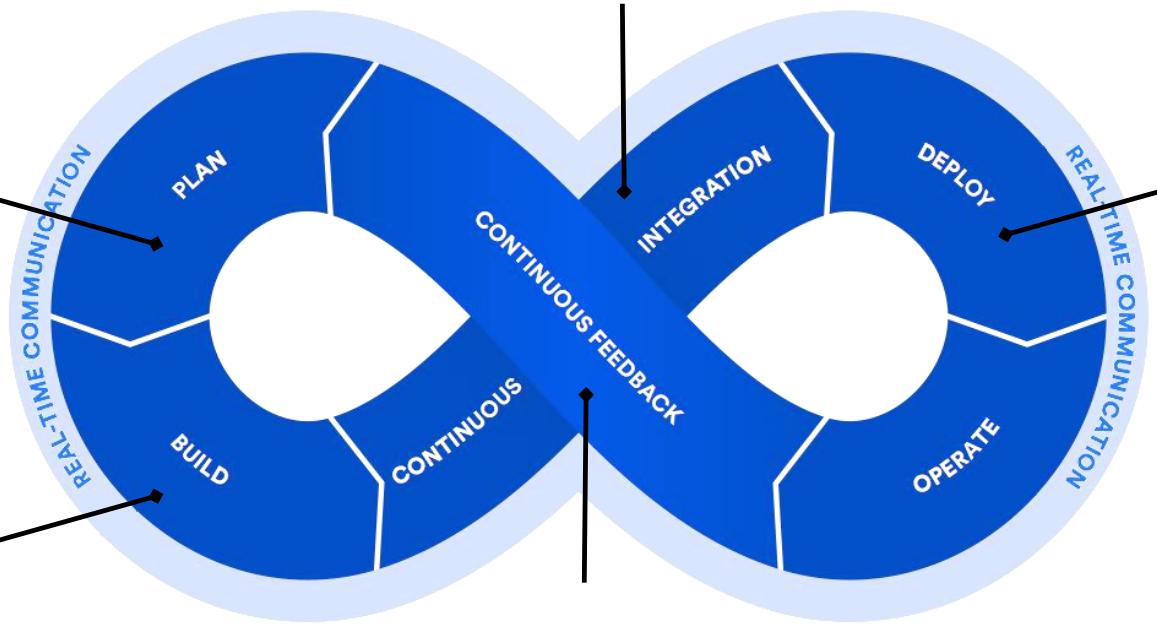
Travis CI

docker

codecov.io

Mattermost
for discussions

GitHub
sprint
planning



SPHINX
PYTHON DOCUMENTATION GENERATOR
readthedocs

python Package Index
(on release)

GitHub
Version control,
code review

GitHub
issue
tracking

HeAT CI/CD pipeline

GitHub

feature branch

New feature
+ unit tests

Pull Request (PR)

- Code Review
- Run unittests
- codecov.io (github plugin)

readthedocs

Travis CI

1. Install linux
2. Pull Docker images for unittests
3. Run tests

planed update to run more complex tests

GitLab runner

- Own CI hardware
- Trusted Approvers

slurm workload manager

HPC,
devoted account

Feature Combination matrix

- CPU
- GPU
- Parallel I/O

Summary

DevOps is a ***set of practices*** *to release software* ***faster*** *and* ***reliably****, and* ***automate*** *this process as much as possible.*

Key techniques are*:*

- *iterative development*
- *version control*
- *test stack*
- *CI/{Delivery | Deployment}*
- *Pipeline-as-a-Code*
- *Infrastructure-as-a-Code*
- *Monitoring*

This set of practices and available tools help ***a single person*** to maintain the ***full product pipeline*** → this is often the case in science!

While in Machine Learning it is not an easy task to run System/Performance tests, one should start with simpler (unit)tests, which already help a lot.

Credits and Links

- Agile: https://en.wikipedia.org/wiki/Agile_software_development
- Coursera course “Continuous Delivery & DevOps” by University of Virginia
- Online: Jez Humble: https://continuousdelivery.com (there‘re many, of course!)
  at medium.com: https://medium.com/search?q=DevOps
- Atlassian Marketplace: marketplace.atlassian.com/categories/devops?utm_source=wac_marketplace_landing
- GitHub Marketplace: https://github.com/marketplace
- Cookiecutter: https://github.com/cookiecutter/cookiecutter
- How to (unit) test machine learning code:
  - https://medium.com/@keeper6928/how-to-unit-test-machine-learning-code-57cf6fd81765
  - Machine learning testing framework for TensorFlow: https://github.com/Thenerdstation/mltest
  - TensorFlow Benchmarks: https://github.com/tensorflow/benchmarks/tree/master/scripts/tf_cnn_benchmarks
- Bandit security scanner: https://github.com/PyCQA/bandit , https://pypi.org/project/bandit/
- OASIS TOSCA: https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=tosca
- udocker: https://github.com/indigo-dc/udocker
- Deep Learning online books: livebook.manning.com/book/deep-learning-with-python (F.Chollet); www.deeplearning.ai/machine-learning-yearning (A. Ng); www.deeplearningbook.org (I. Goodfellow);