



# DevOps adoption in scientific applications: DisVis and PowerFit cases

Pablo Orviz<sup>1</sup>, Mikael Trellet<sup>2</sup>, Alexandre Bonvin<sup>2</sup>

<sup>1</sup>Instituto de Física de Cantabria, Santander, Spain

<sup>2</sup>Utrecht University, Utrecht, The Netherlands



## Plan

- Good development practices in scientific applications
- DevOps culture
- DisVis and PowerFit use cases
- Our continuous delivery pipeline

Good practices in  
software development



# Good development practices in scientific applications

## Why?

Scientific/academic software development has similar goals than professional/commercial software

- User satisfaction
- Bug-free end-product
- Coding environment structured
- Seamless release of new features

However, until now, and as far as we know, only few scientific software seem to have a development policy that would professional good development practices.

# Good development practices in scientific applications

## When?

Several steps of software development must follow good practices:

- Code writing (dependencies, comments, etc.)
- Code checking (e.g. PEP8)
- Code compilation
- Code testing (performances, results, etc.)
- Code release (e.g. SCM)
- Documentation

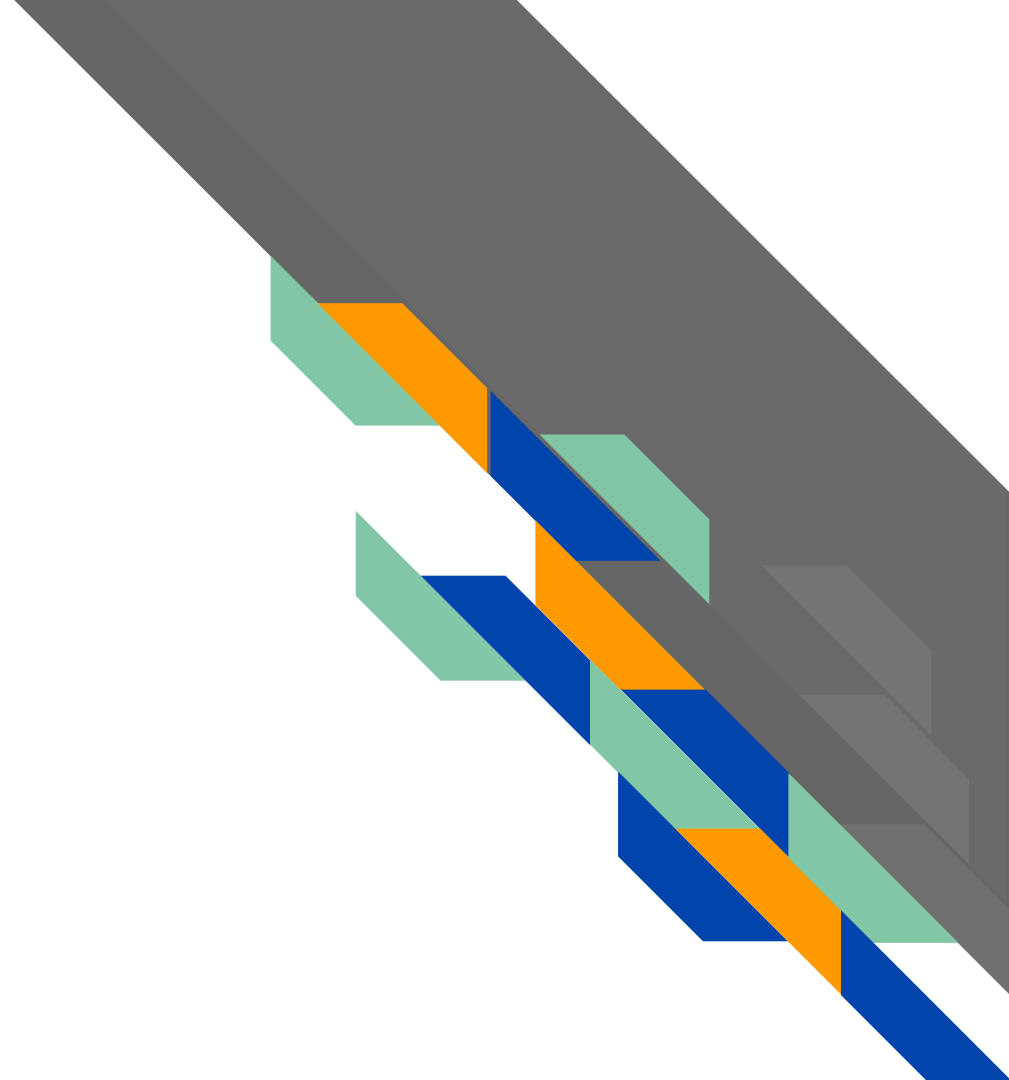
# Good development practices in scientific applications

## How?

Nowadays, plethora of tools, servers, libraries, help users to maintain a good software development environment

- Code writing ⇒ Developer documentation
- Code checking ⇒ IDE, style conventions checking tools
- Code compilation ⇒ Automated deployment
- Code testing ⇒ Unit tests, integration tests
- Code release ⇒ Integration with SCM tools
- Documentation ⇒ Comments/docstrings scanning tools

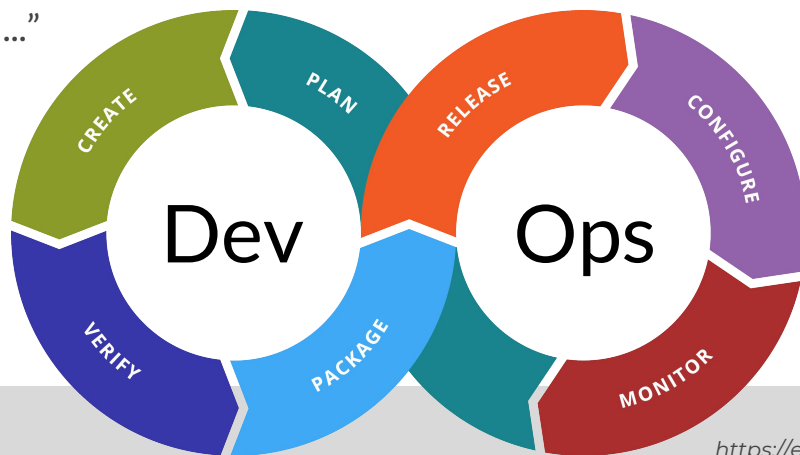
DevOps culture



# DevOps culture for software development

What is the DevOps culture?

- Wikipedia: “... a software engineering culture and practice that aims at unifying **software development (Dev)** and **software operation (Ops)**. The main characteristic of the DevOps movement is to strongly advocate **automation** and **monitoring** at all steps of software construction, from **integration, testing, releasing to deployment ...**”





# Continuous Integration/Delivery/Deployment

**Continuous Integration (CI)**

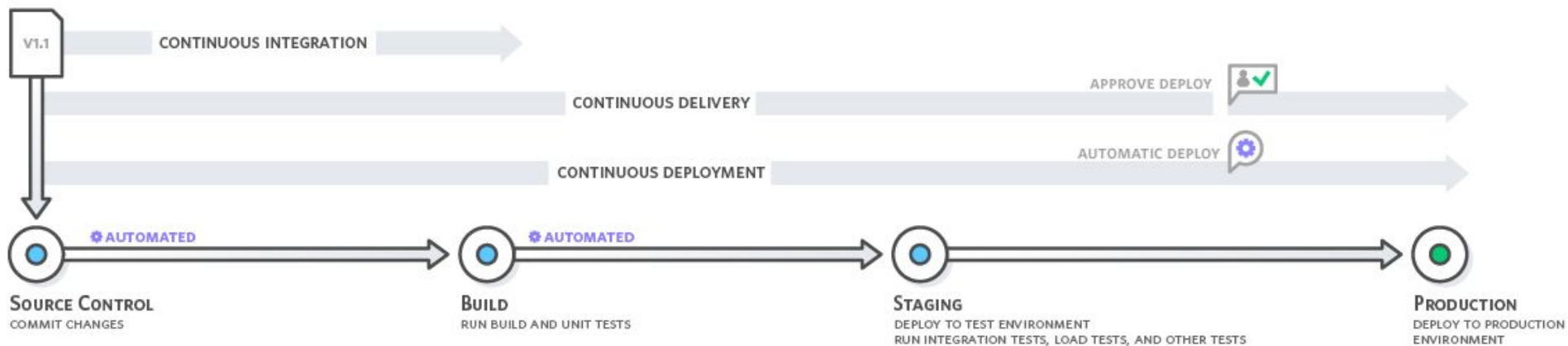
Code merged in repository  $\Rightarrow$  Automated builds and code tests performed

**Continuous Delivery (CD)**

CI success  $\Rightarrow$  Deploy in testing environments and performs integration tests

**Continuous Deployment (CD')**

CD success  $\Rightarrow$  Deploy to production environment (e.g. as a stable release)

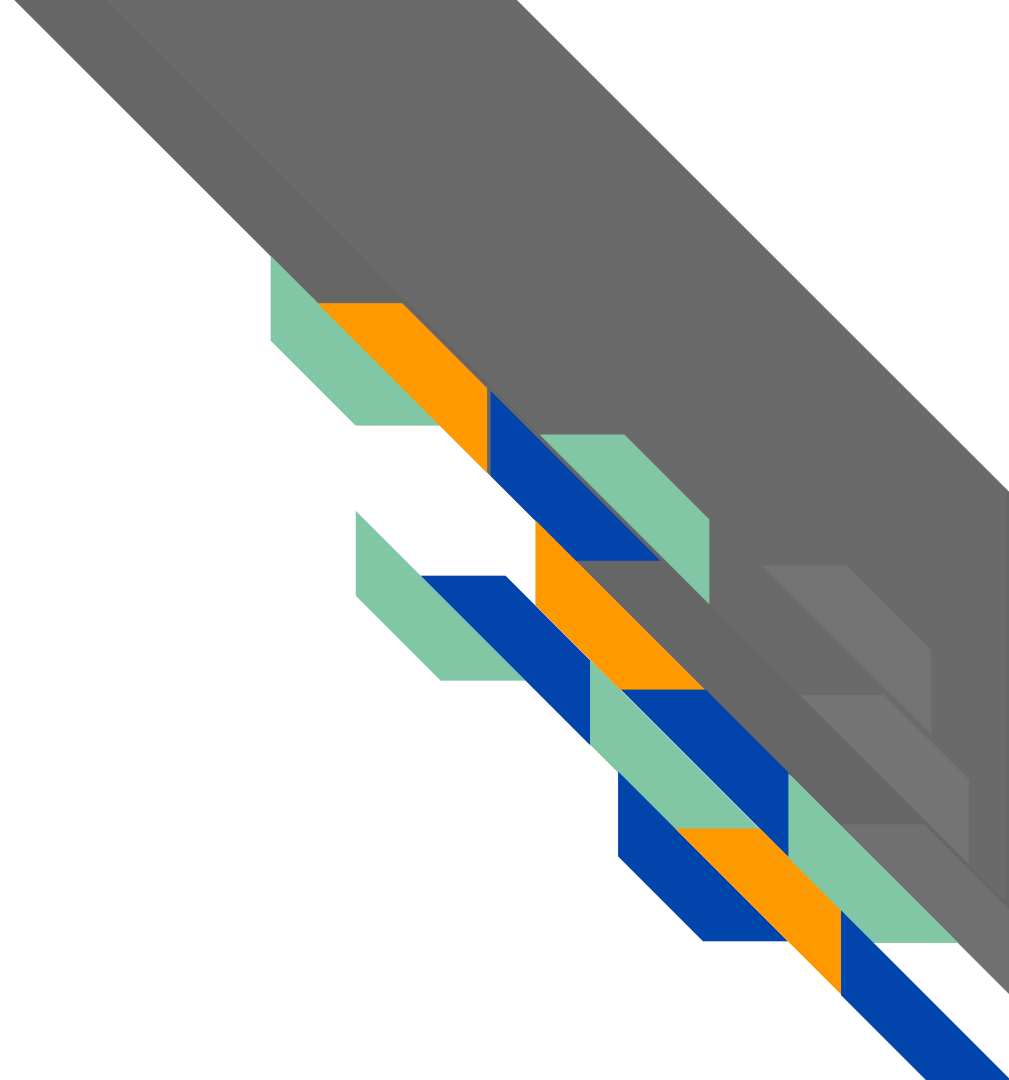




# DevOps culture for scientific/academic software development

- **Limited** number of contributors  $\Rightarrow$  Development/operations often made by the **same person/people**
- Not all scientists are familiar with **best practices** in operations
- **Reproducibility** must be at the core of Science, setting up **automated** and **systematic** checking points reduces chances of results divergence
- Cost of **defect solving** is reduced if detected **early** in the software development process
- Allows for **faster** and **more frequent** updates, **reduces time** between algorithms improvement and their availability for users
- More **frequent updates**  $\Rightarrow$  **smaller changes**  $\Rightarrow$  **less risk of disruption** in operations
- Automating **test/build/stage** steps allow to focus on the **core** development, the **algorithm(s)**

DisVis & PowerFit



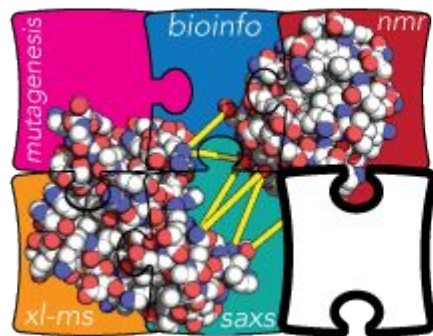
# DisVis/PowerFit test cases

## Bonvin Lab



DisVis

Prodigy



**ADDOCK**  
High-Ambiguity Driven Docking

PowerFit

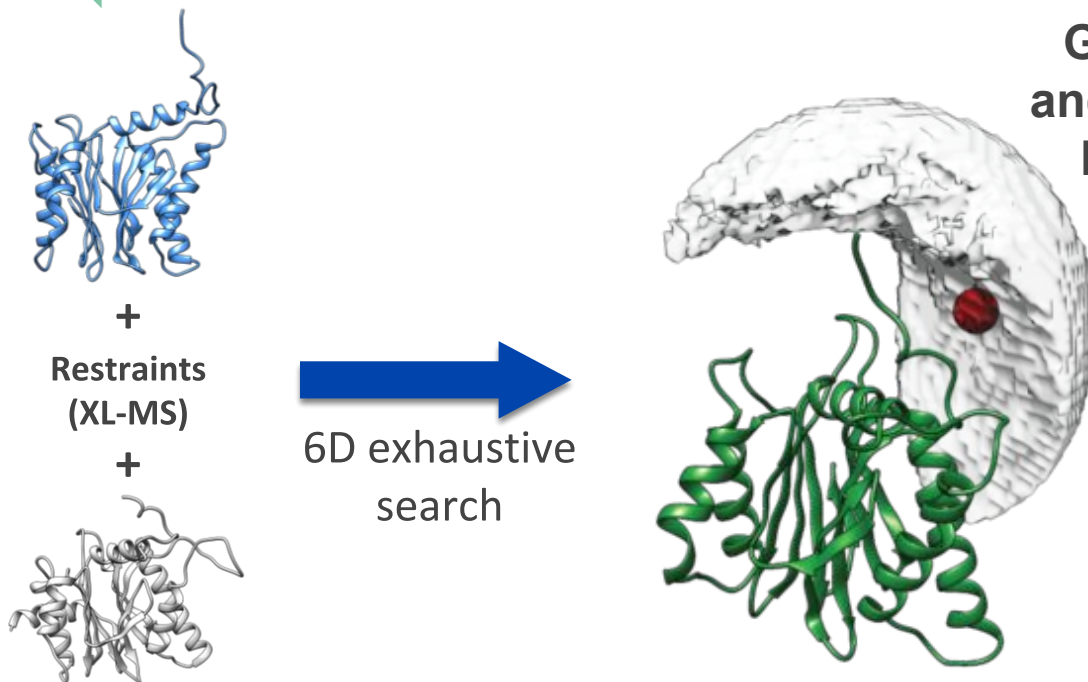
CPORT

SpotON

[haddock.science.uu.nl](http://haddock.science.uu.nl)

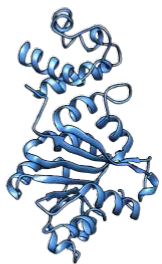
3D-DART

# DisVis - visualising accessible interaction space

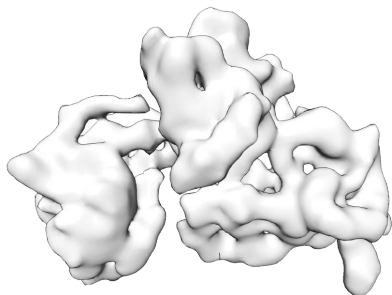


**Given 2 interacting structures  
and a set of distance constraints  
between them, are there any  
solutions that satisfy  $N$   
constraints?**

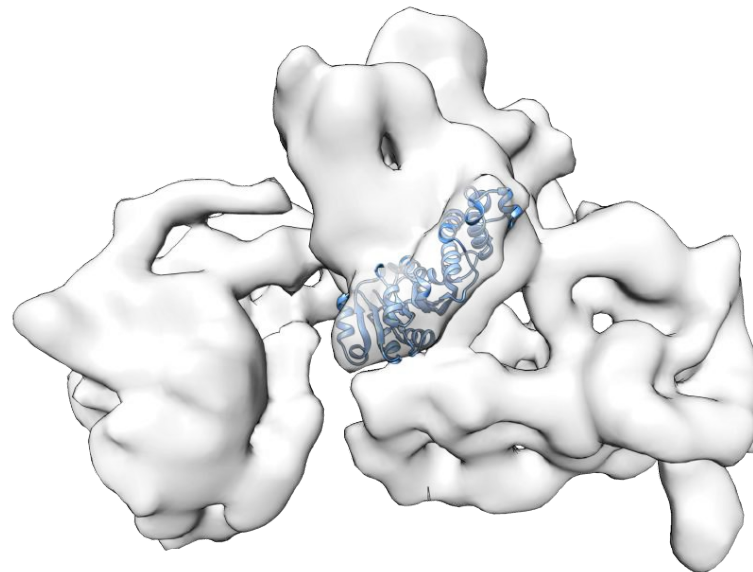
# PowerFit - fitting structures into 3D maps



3D structure  
+  
Low-resolution  
density map



6D exhaustive  
search



Best Fit of structure in  
density map

# GRID-enabled web portals



Home >> DISVIS >> Index

## DISVIS

GRID-enabled web portal @BonvinLab

HADDOCK CPORIT **DISVIS** POWERFIT PRODIGY 3D-DART

About Submit Register Example Help/Manual Support

WELCOME TO THE GRID-ENABLED DISVIS WEBSERVER! >>

DisVis visualizes the accessible interaction space!

A B

bioexcel  
INDIGO - DataCloud  
MoBrain  
esi

**DISVIS WEBSERVER**

**REGISTRATION:** To use the DisVis server you must have registered for an account. If you do not have an account yet you can [register here](#)

**Submit your job to:**

- DISVIS GPU accelerated Grid server
- DISVIS server

<https://milou.science.uu.nl/services/DISVIS>



Home >> POWERFIT >> Index

## POWERFIT

GRID-enabled web portal @BonvinLab

HADDOCK CPORIT DISVIS **POWERFIT** PRODIGY 3D-DART

About Submit Register Example Help/Manual Support

WELCOME TO THE GRID-ENABLED POWERFIT WEBSERVER! >>

PowerFit fits your 3D structures in any map!

PowerFit automatically fits high-resolution atomic structures into cryo-EM densities.

To this end it performs a full-exhaustive 6-dimensional cross-correlation search of the atomic structure and the density. It takes as input a cryo-EM density with its resolution, a high-resolution atomic structure corresponding to the density, and a correlation function as its input. It performs a pre-filter and/or a neighborhood search.

You must have registered for an account. If you do not have an account yet you can [register here](#)

PowerFit GPU accelerated Grid server

**REFERENCE FOR USE OF THE SERVER**

When using the PowerFit server please cite:

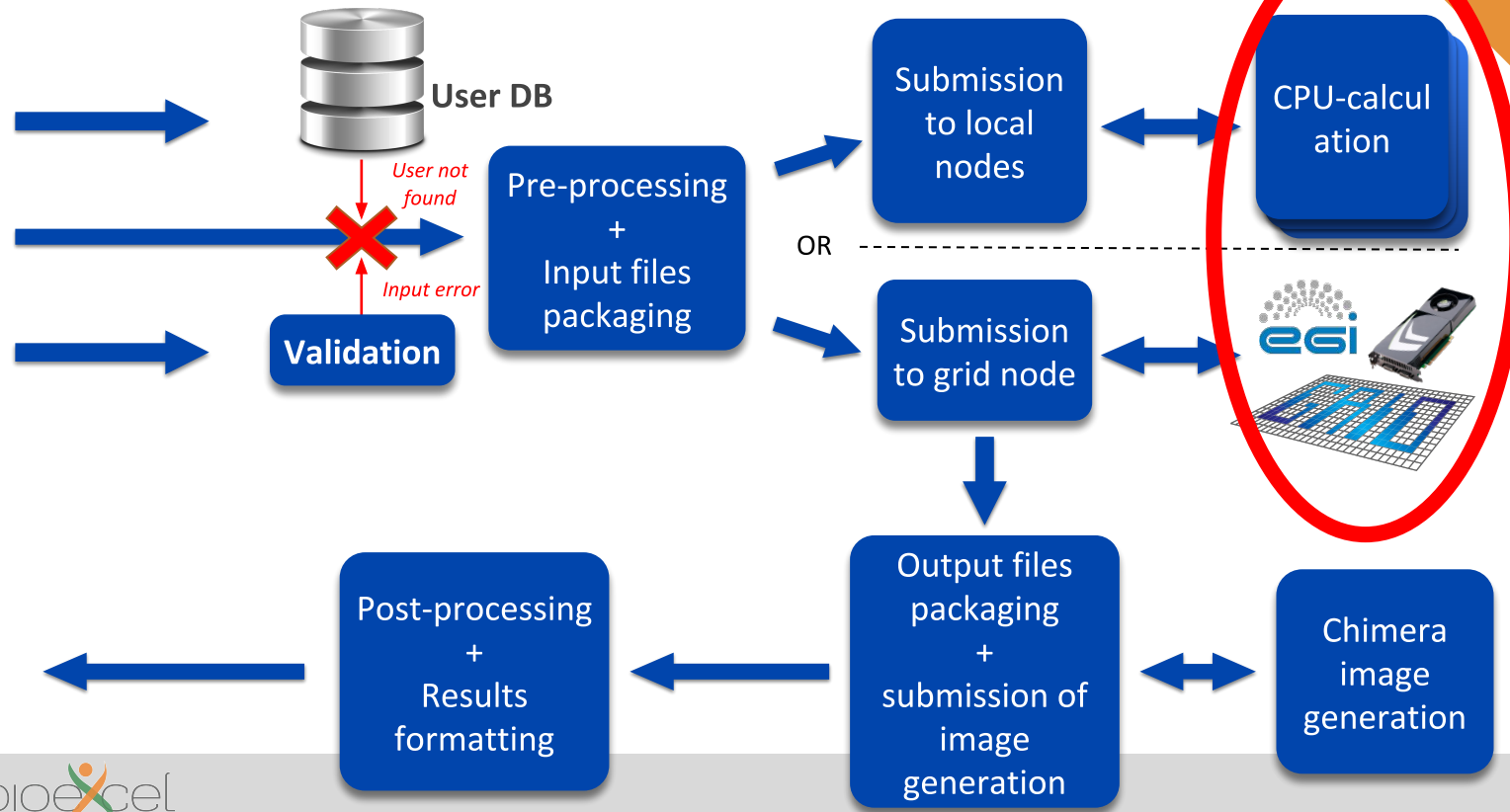
G.C.P. van Zundert and A.M.J.J. Bonvin (2015)  
PowerFit: Quantifying and visualizing accessible interaction space of distance-restrained biomolecular complexes...  
*AIMS Biophysics* 2, 73-87.

Open "milou.science.uu.nl/cgi/entz/services/POWERFIT/powerfit/" in a new tab

bioexcel  
INDIGO - DataCloud  
MoBrain  
esi

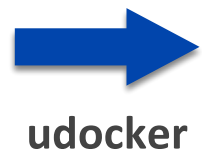
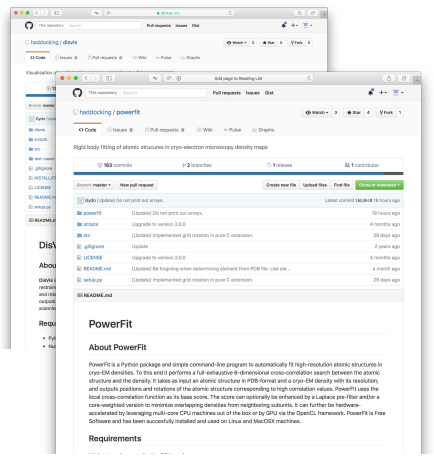
<https://milou.science.uu.nl/services/POWERFIT>

# Architecture





# From web form to GPGPU resources



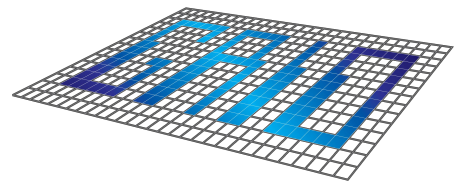
udocker



indigodatacloudapps/disvis



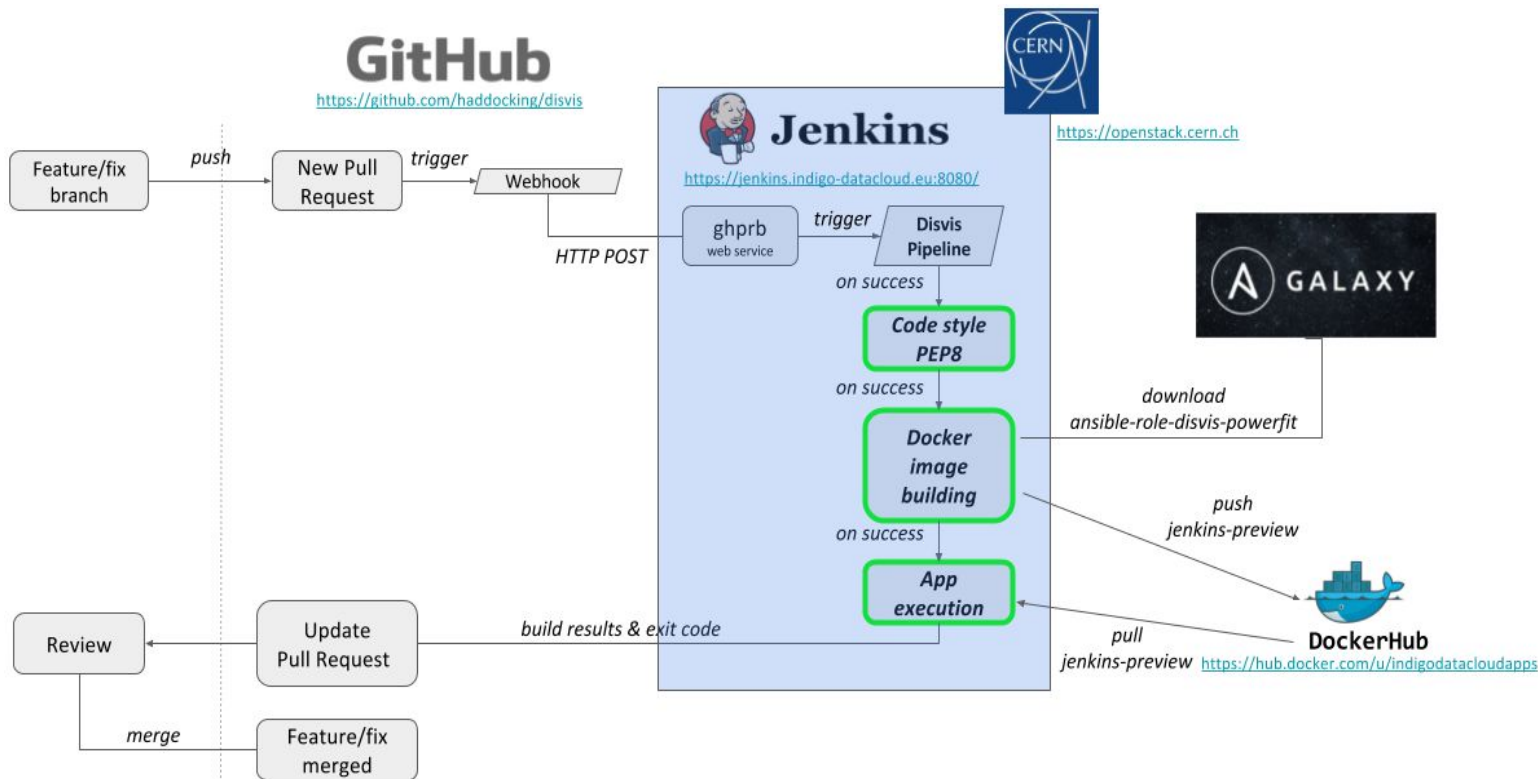
indigodatacloudapps/powerfit



Our continuous delivery pipeline



# DevOps workflow for DisVis/PowerFit



# Step-by-step

## Setup

- **Github repository** setup to trigger “**webhooks**” when certain event occurs (PR, comment on PR, push, etc.)
- **Jenkins server** with (1) GitHubPullRequestBuilder (**GHPRB**) plugin allows to handle webhooks received from Github and triggers the pipeline, (2) pipeline as set of command-line instructions

## Continuous Delivery pipeline (1/2)

1. **Fetch** the branch to test (source of the Pull Request)
2. Run **pep8** (pycodestyle, soon to be flake8) for code style checking of the whole project
3. **Build** docker images (GPU driver & application) with **Ansible** roles
4. Send to **indigodatacloudapps** ([hud.docker.com](http://hud.docker.com)) with proper **tag** for testing (**not** production yet)

# Step-by-step

## Continuous Delivery pipeline (2/2)

5. Execute app on a running node with **GPU capability**
6. **Validate** results with expected values (**integration test**)

Length of the pipeline: **~7minutes**

## Deployment

- Feedback sent to Github PR that triggered the pipeline (using GitHub API)
- If needed/wanted, the PR is merged
- The indigodataclouds docker image gets its tag updated for production

# Jenkins server

Pipeline #5

```

ghprbActualCommitAuthorEmail: onviz@fca.unican.es
ghprbTriggerAuthorLogin: amj@bonvin
ghprbGitRepository: haddocking/powerfit
ghprbTargetBranch: master
ghprbPullId: 6
ghprbCredentialsId: 455dc27f22949c0aad9ad5fe6669622
ghprbAuthorRepoGitUrl: https://github.com/haddocking/powerfit.git
ghprbTriggerAuthor: Alexandre Bonvin
ghprbPullAuthorEmail:
ghprbPullLink: https://github.com/haddocking/powerfit/pull/6
sha1: origin/pr/6/merge
ghprbPullAuthorLoginMention: @indigobot
ghprbPullDescription: GitHub pull request #6 of commit fb6d280d3deb7c7746251631248a3dc81fb368, no merge conflicts.
ghprbActualCommit: fb6d280d3deb7c7746251631248a3dc81fb368
ghprbPullAuthorLogin: indigobot
ghprbSourceBranch: indigo
ghprbActualCommitAuthor: Pablo Onviz
ghprbTriggerAuthorLoginMention: @amj@bonvin
ghprbPullLongDescription:
ghprbTriggerAuthorEmail:
GIT_BRANCH: indigo
ghprbPullTitle: Test Integration (PLEASE DO NOT MERGE)
ghprbCommentBody: retested:n/n
                    
```

#5 apps-powerfit-codestyle-pipe

Aug 10, 2017 10:28:29 AM

9.3 sec

#6 indigodatacloudapps/powerfitjenkins-preview app-powerfit-dockerhub-pipe


Aug 10, 2017 10:28:45 AM

5 min 15 sec

#2 app-powerfit-run

Aug 10, 2017 10:43:47 AM

13 sec



# Failed step in Jenkins server

The screenshot displays the Jenkins web interface for a pipeline named "pipe-apps-disvis". The main heading is "Build Pipeline". Below it, a "History" icon is visible. The pipeline consists of three stages:

- Pipeline #24**: A small grey box on the left.
- #24 app-disvis-codestyle-pipe**: A red box indicating a failed stage. It shows a failure icon, the date "Aug 15, 2017 10:16:27 AM", and "7 steps".
- app-disvis-dockerhub-pipe**: A blue box indicating a successful stage. It shows "N/A" for both status and duration.
- app-disvis-run**: A blue box indicating a successful stage. It shows "N/A" for both status and duration.

Green plus signs are located between the red and blue boxes, and between the two blue boxes, suggesting a continuation of the pipeline flow.



# In a nutshell

This pipeline is highly configurable and we are showcasing a test case that might be more complete:

- Add **unit tests** in step 2 (as of today, only code style is checked with pep8)
- Also test **multi-CPU** results (as of today, only the GPU version is checked because more error-prone and the one used through docker images in production)
- Implements **Continuous Deployment** where docker images are production-ready

Our short-term view:

- Apply this pipeline to more projects (**PDB-tools**, **HADDOCK-tools**, web servers, etc.)
- Improve the **feedback loop** when a step is failing (remove docker images when integration tests are failing, provide pipeline output to developer in GitHub, etc.)



# Acknowledgments



## CSB\_group

**Alexandre Bonvin**  
Jorg Schaarschmidt  
Adrien Melquiond


## INDIGO datacloud

## **Pablo Orviz**



THANK YOU FOR YOUR ATTENTION


# Failed step in GitHub




mtrellet commented 6 days ago Member + 😊 ✎ ✕

```
run jenkins
```

Add more commits by pushing to the **pep8** branch on **haddocking/powerfit**.



**✖ All checks have failed** Hide all checks  
1 failing check

**✖**  **app-powerfit-pipe** — Build finished. Details

**✔ This branch has no conflicts with the base branch**  
Merging can be performed automatically.

**Merge pull request** ▾ You can also [open this in GitHub Desktop](#) or view [command line instructions](#).