The many ways of using IMAS stack
- or how to apply Docker based IMAS

example of running HELENA code inside Docker

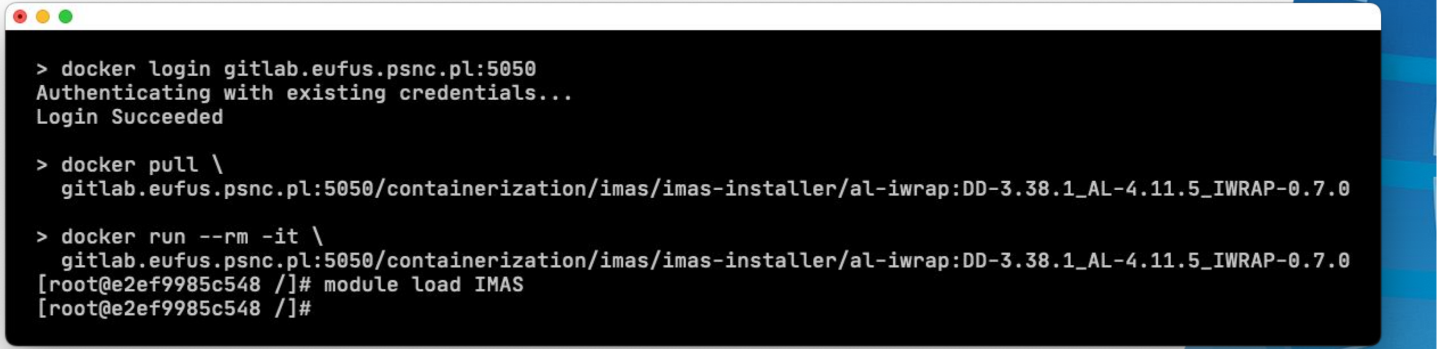
ACH-04

Aleksander Wiszniowski, Bartłomiej Pogodziński

Running codes inside Docker

Docker images with IMAS inside

- Thanks to Docker you can deploy IMAS anywhere
- Docker images are available via Container Registry
- Tested with
  - Docker
  - Singularity
  - uDocker



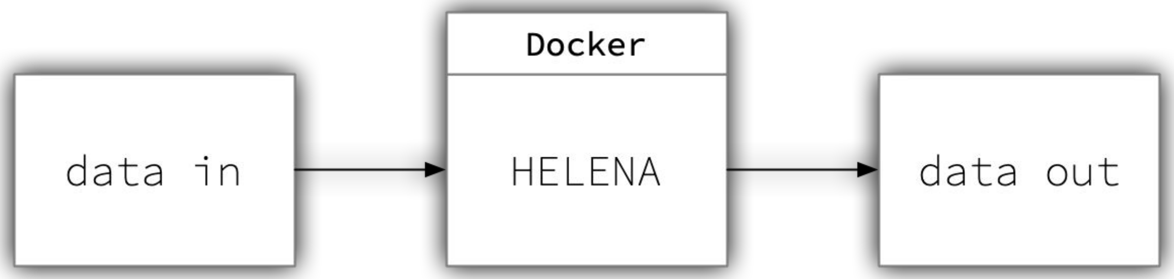
```
> docker login gitlab.eufus.psnc.pl:5050
Authenticating with existing credentials...
Login Succeeded

> docker pull \
  gitlab.eufus.psnc.pl:5050/containerization/imas/imas-installer/al-iwrap:DD-3.38.1_AL-4.11.5_IWRAP-0.7.0

> docker run --rm -it \
  gitlab.eufus.psnc.pl:5050/containerization/imas/imas-installer/al-iwrap:DD-3.38.1_AL-4.11.5_IWRAP-0.7.0
[root@e2ef9985c548 /]# module load IMAS
[root@e2ef9985c548 /]#
```

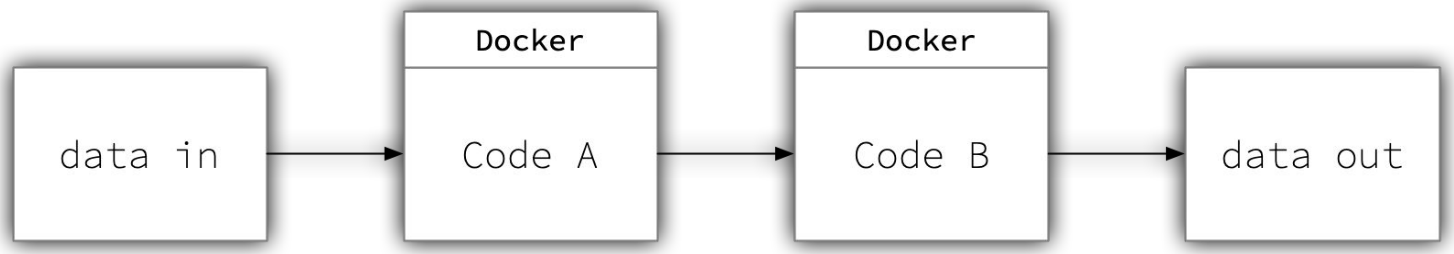
Real life applications

- Helena code inside Docker container
- JINTRAC
- UDA server and **UDA client**
- Simulation Catalogue
- CI/CD plans
  - you can test your code inside **Docker**



Each code as a separate Docker container

- It is possible to couple codes manually - IMAS multi-container demo
- It is possible to couple codes inside Muscle3 - Persistent Actor Framework



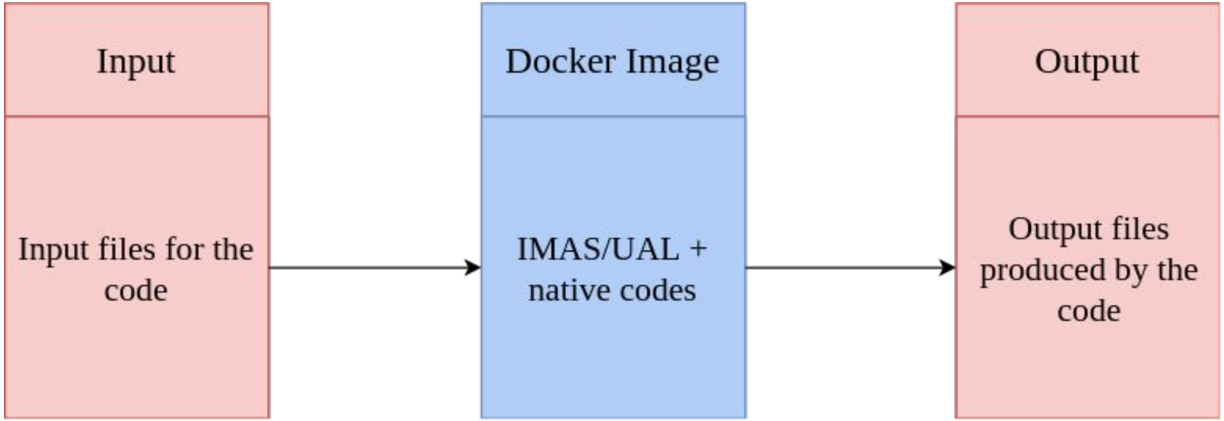
Hello world example

repository: https://gitlab.eufus.psnc.pl/g2awisz/testing_repository

branch: Docker/multi-container-usage

How Docker based IMAS can be applied to your code

- HELENA case



How Docker based IMAS can be applied to your code

- HELENA case -> how to run

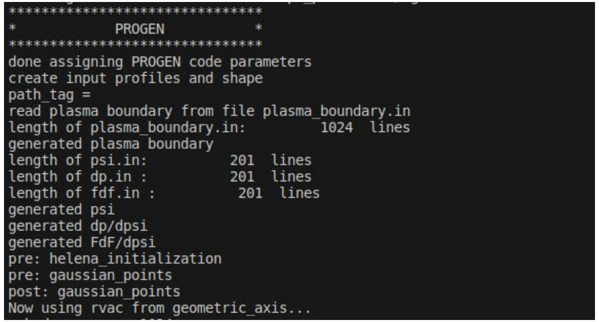
docker run --rm -v path_to_preprocessed_files:/opt/preprocessed_files rockylinux:8.6 bash -c "./some_native_code --include /opt/preprocessed_files "

Helena based docker image pre requirements:

1. Pull build helena image from container registry -> recommended
2. Build image on your own -> Not recommended + works only on unix-based machines

How to run Helena based docker image:

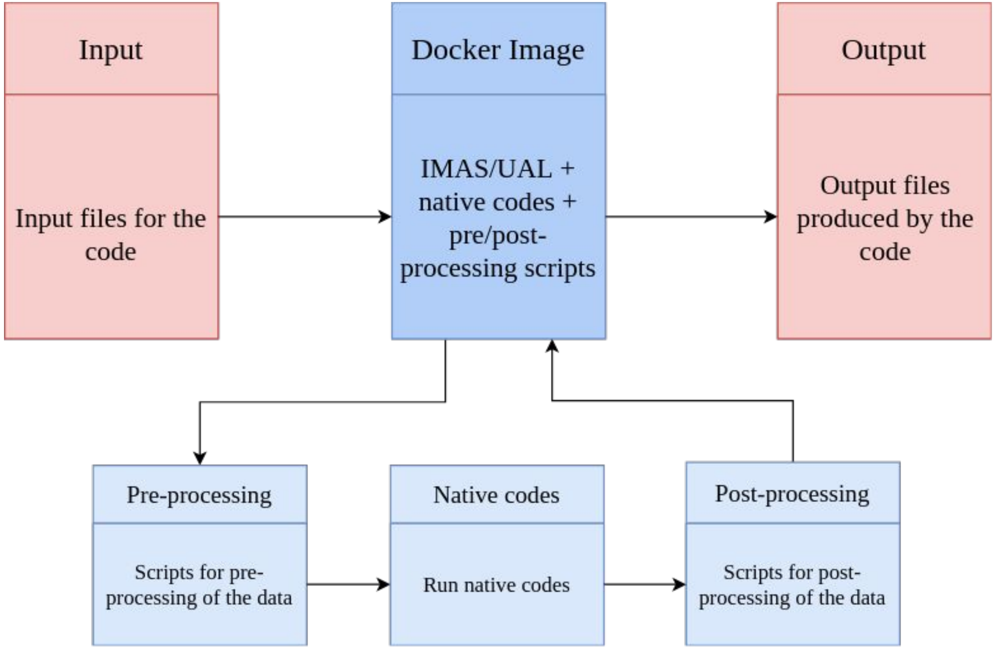
1. Run docker container with helena: **docker run --rm -it helena_image**
2. Set intel vars: **. /opt/intel/compiler/latest/env/vars.sh**
3. Navigate to helena code directory: **cd /opt/helena_itm/my_stuff**
4. Run Helena code: **../bin/HELENA_IDS_3381 666 1 1 root IMASDB 3.37.0**



```
*******************************
*            PROGEN           *
*******************************
done assigning PROGEN code parameters
create input profiles and shape
path_tag =
read plasma boundary from file plasma_boundary.in
length of plasma_boundary.in:         1024  lines
generated plasma boundary
length of psi.in:          201  lines
length of dp.in :          201  lines
length of fdf.in :          201  lines
generated psi
generated dp/dpsi
generated FdF/dpsi
pre: helena_initialization
pre: gaussian_points
post: gaussian_points
Now using rvac from geometric_axis...
```

How Docker based IMAS can be applied to your code

- HELENA case + Python inside Docker



Input

Input files for the code

Docker Image

IMAS/UAL + native codes + pre/post-processing scripts

Output

Output files produced by the code

Pre-processing

Scripts for pre-processing of the data

Native codes

Run native codes

Post-processing

Scripts for post-processing of the data

How Docker based IMAS can be applied to your code

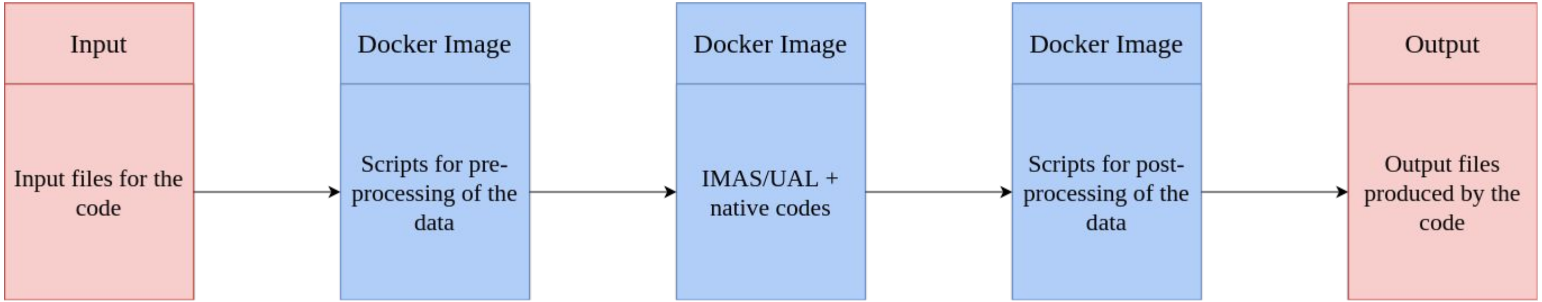
- HELENA case + Python inside Docker -> how to run

```
docker run --rm -v $(pwd):$(pwd) rockylinux:8.6  bash -c "echo =========  Output && \
echo pre-processing script execution > $(pwd)/single_docker.txt && \
echo native code execution >> $(pwd)/single_docker.txt && \
echo post-processing script execution >> $(pwd)/single_docker.txt && \
cat $(pwd)/single_docker.txt"
```

```
========= Output
pre-processing script execution
native code execution
post-processing script execution
```

How Docker based IMAS can be applied to your code

- HELENA case + two additional Docker containers



Input

Input files for the code

Docker Image

Scripts for pre-processing of the data

Docker Image

IMAS/UAL + native codes

Docker Image

Scripts for post-processing of the data

Output

Output files produced by the code

How Docker based IMAS can be applied to your code

- HELENA case + two additional Docker containers -> how to run

```
docker run --rm -v $(pwd):$(pwd) rockylinux:8.6  bash -c "echo ========= Output && echo Hello from
docker nr. 1 > $(pwd)/chained_dockers.txt && cat $(pwd)/chained_dockers.txt" && \

docker run --rm -v $(pwd):$(pwd) rockylinux:8.6  bash -c "echo ========= Output && echo Hello from
docker nr. 2 >> $(pwd)/chained_dockers.txt && cat $(pwd)/chained_dockers.txt" && \

docker run --rm -v $(pwd):$(pwd) rockylinux:8.6  bash -c "echo ========= Output && echo Hello from
docker nr. 3 >> $(pwd)/chained_dockers.txt && cat $(pwd)/chained_dockers.txt"
```

```
========= Output
Hello from docker nr. 1

========= Output
Hello from docker nr. 1
Hello from docker nr. 2

========= Output
Hello from docker nr. 1
Hello from docker nr. 2
Hello from docker nr. 3
```

Git repository with example of chaining IMAS based images: repo

How Docker based IMAS can be applied to your code

- Size reduction of Helena Docker Image

In order to run Helena codes inside Docker Container we're required to have Intel OneApi installed. Unfortunately Docker Images containing Intel OneApi inside them weights over 15Gb.

Solution:
In production versions of Docker Images we can install solely Intel OneApi runtimes. It allows us to reduce size of Image from 15Gb to 8Gb (47% space reduction). With these runtimes we're able to run codes compiled with Intel OneApi compilers from previous build stage

Acknowledgment

This work has been carried out within the framework of the EUROfusion Consortium, funded by the European Union via the Euratom Research and Training Programme (Grant Agreement No. 101052200-EUROfusion). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Commission. Neither the European Union nor the European Commission can be held responsible for them.

The project activities are co-financed by Polish Ministry of Science and Higher Education from financial resources of the programme entitled "PMW" in 2023, 2024.

Any questions?

Acknowledgment

This work has been carried out within the framework of the EUROfusion Consortium, funded by the European Union via the Euratom Research and Training Programme (Grant Agreement No. 101052200-EUROfusion). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Commission. Neither the European Union nor the European Commission can be held responsible for them.

The project activities are co-financed by Polish Ministry of Science and Higher Education from financial resources of the programme entitled "PMW" in 2023, 2024.