

Work Package 4: Simulation Ecosystem

Project kick-off meeting Barcelona,
19-20 January 2023

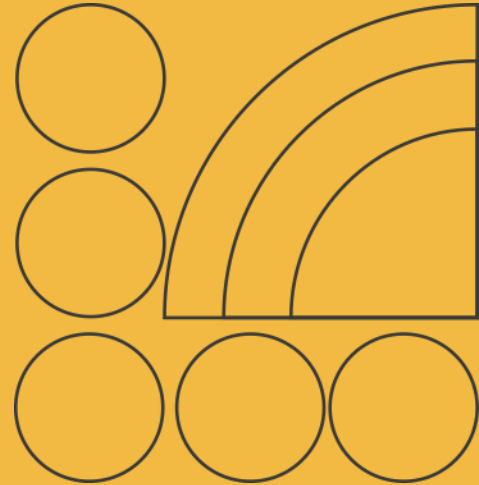
Christoph Niethammer
HLRS



ChEASE



WP4-Overview





WP4 Objectives

1

Address challenges at simulation workflow level that go beyond the main application codes.

2

Decrease overall time to solution by adopting a holistic view on the necessary workflow

3

Exploit new methods of simulation setup, deployment, and data analysis

4

Improve general usability for codes and workflows in the SE Community



WP4 Tasks

Task	Task name	Start	End	Lead
T4.1	Workflow Implementation and Optimization	1	42	HLRS
T4.2	Lightweight Performance Analysis and optimization services	25	48	HLRS
T4.3	Development, Deployment and Quality Assessment	1	48	CIN/ HLRS

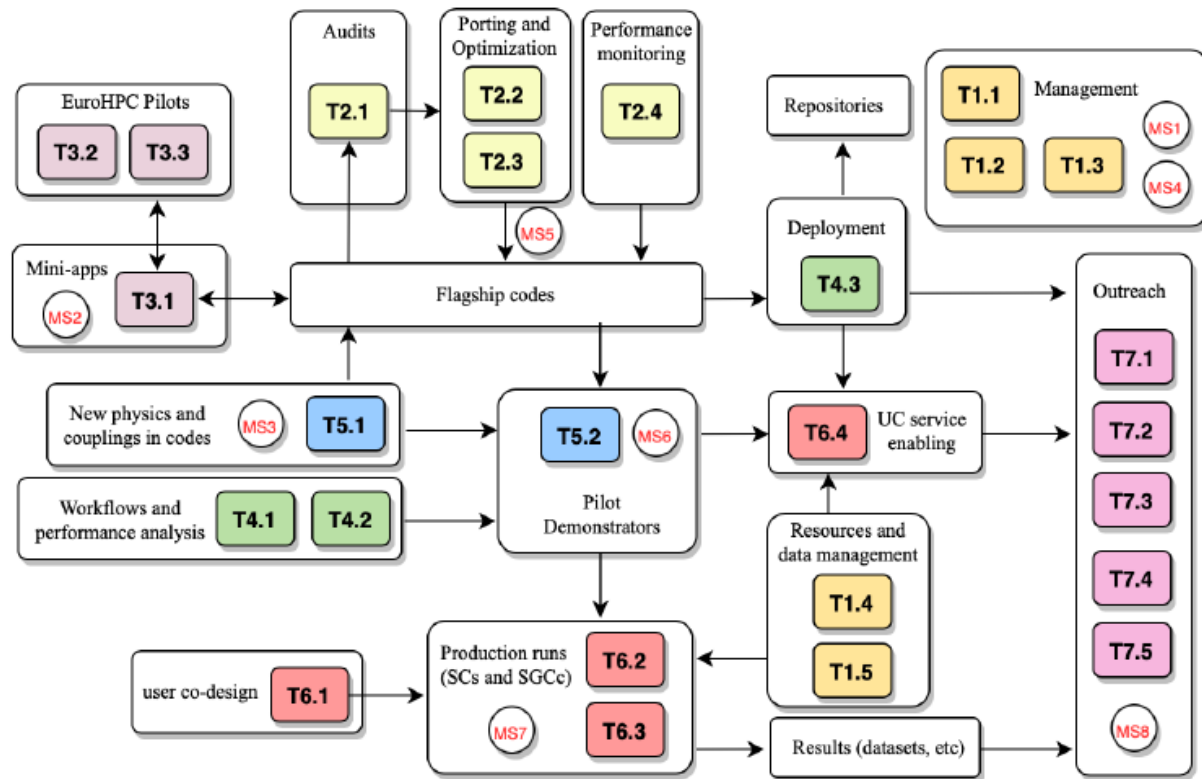
Effort (PMs)														
CSCI	BSC	INGV	INO	HLRS	CIN	TUM	LMU	UMA	NGI	CNRS	BA	CSC	IRB	UM
6	6	6	-	42	12	-	6	6	2	6	-	-	2	-

WP4 Deliverables

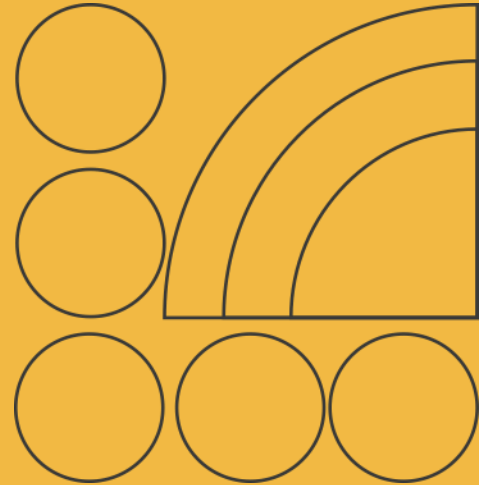
Task	Deliverable name	Lead	Type	Due
D4.1	Initial report on workflow setups and workflow auditing	HLRS	R	M18
D4.2	Plan for development and deployment tool usage for workflows and software	CIN	R	M20
D4.3	Intermediate report on workflow setups and optimization	HLRS	R	M36
D4.4	Final report on workflow implementation and optimization	HLRS	R	M42
D4.5	Final report on development and deployment activities	HLRS	R	M44

MS	Milestone name	Description	Lead	Due
MS10	Application deployment	Codes available to users of EuroHPC via automated deployment, documentation, testing and quality checks	HLRS	M12

WP4 Interactions in ChEESE-2P



WP4-Tasks



T4.1: Workflow Implementation and Optimization

Lead: HLRS

Partners: CSIC, BSC, INGV, LMU, NGI, CNRS, IRB

Duration: M1-M42

Implement and optimize simulation workflows as a whole:

- Reduce user interaction
- Allow fast deployment of simulations and handling of large ensemble workloads

Optimize non-obvious bottlenecks beside the main HPC simulation in workflows:

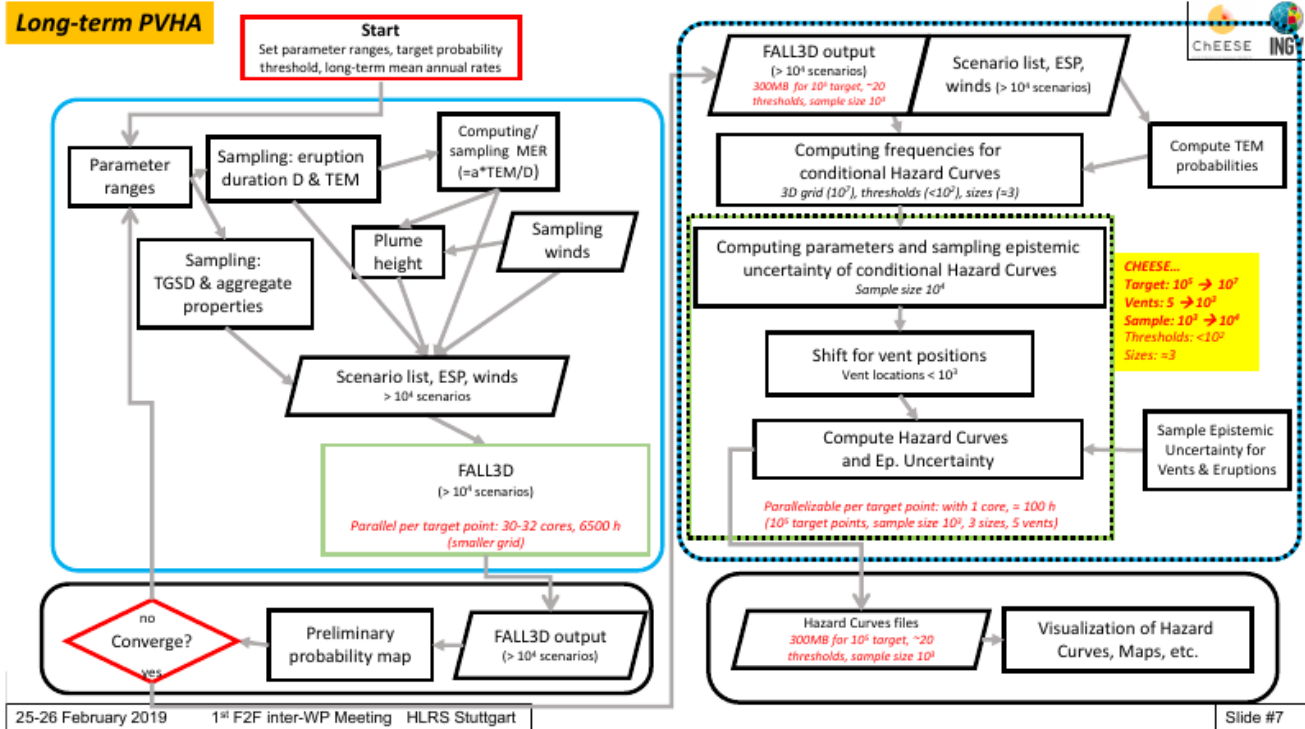
- Important as not only the main codes must scale
- Workflow audits to find bottlenecks
- Optimize time-consuming components and inter-component coupling

→ Especially important for Urgent Computing

- → Main pilot-owner-participation working on their workflows



From Cheese-1P: A workflow overview (PVHA PD)



From Cheese-1P: Basic workflow assessment (PVHA PD)

Basic analysis:

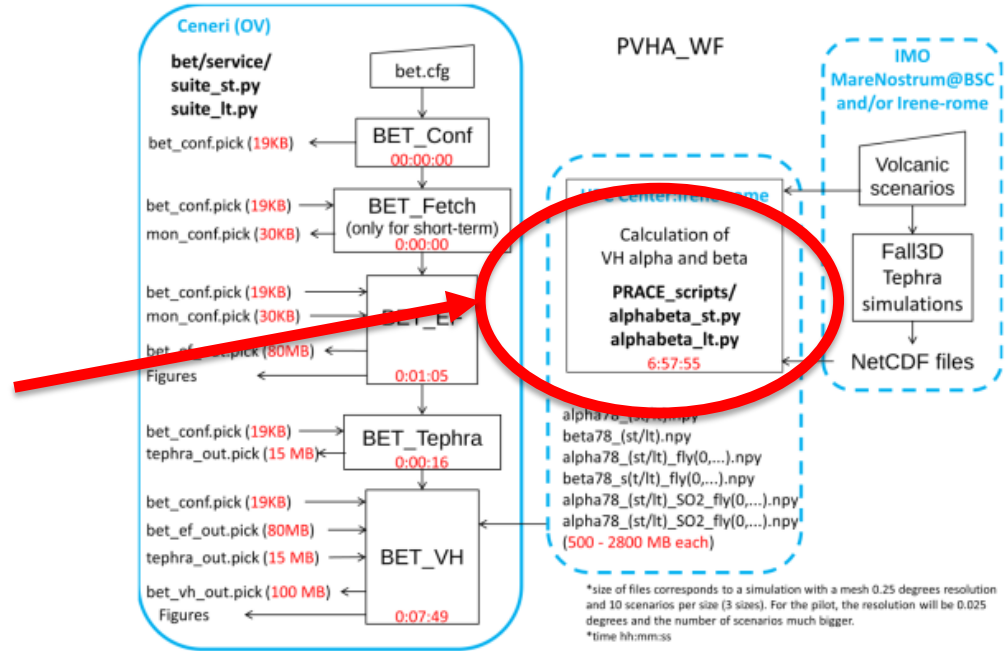
- Check runtimes and data volume of all components in the workflow

Analysis result:

- Non negligible time in non-main code with limited scalability,

Optimization work:

- 500x speedup for component



T4.2: Lightweight Performance Analysis and optimization services

Lead: HLRS

Partners: CSIC, INGV, LMU, UMA, CNRS

Duration: M25-M48

Address challenges for performance analysis and optimization of components in geo-scientific workflows:

- Provide lightweight performance analysis services and tools for workflow components helping with the identification and analysis of performance problems and use for continuous performance monitoring later on
- Help with performance analysis of workflow components that are not based on common HPC programming languages, e.g. python, and do not use the common parallel programming models, i.e., MPI or OpenMP
- Help with performance analysis of components for which POP performance metrics are hard to obtain
- Possible usage of Talp from T2.4



T4.3: Development, Deployment and Quality Assessment

Lead: CIN/HLRS

Partners: CSIC, BSC, INGV, LMU,UMA, CNRS

Duration: M1-M48

1 Development

2 Deployment

3 Quality Assessment





Development

Today's development platforms offer:

- Collaborative working with code
- Continuous Integration (CI) support, e.g., automatic testing and documentation
- User-developer interaction via „issues“ - serving later also as public searchable knowledge base
- Most common platforms: github, gitlab, bitbucket

Status of Cheese codes:

- All available via development platforms – BUT not all use all features





Deployment

Goals:

- Simplify general set up for codes and workflows
- Help with Urgent Computing tasks
- Fault tolerance
- Allow fast testing in development and installation setup

Possible technologies:

- Spack – HPC software package system
 - Containers: Docker, Singularity, udocker
 - Jupiter notebooks / python packages
 - NVIDIA Software collections
 - OpenHPC (?)
- 

Quality assessment

Code and script quality checks:

- state of the art tools identify potential programming errors, e.g., static code analysis (clang-tools, github/LGTM, Coverity)
- dynamic methods using sanitizers find failures at runtime, e.g., used in CI testing pipelines and help with user bug reports

Test harnesses for common use cases:

- Allow developers to detect quickly logical programming errors, i.e., errors leading to wrong results
- Allow administrators to test and benchmark installations

Documentation tools:

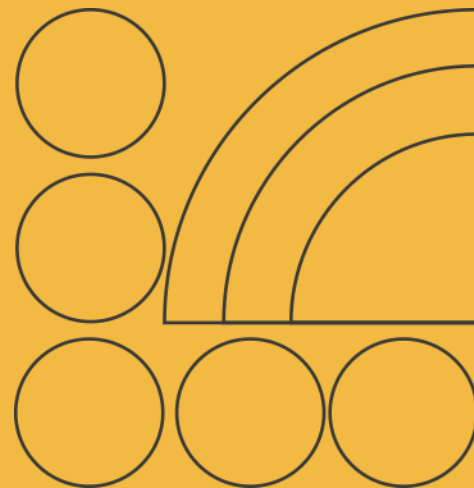
- Generate well formatted documentation and keep it up to date
- e.g. readthedocs, github pages, doxygen



Code Status

No	Code name	development	CI / testing	deployment	Quality assessment
1	SeiSol	github	gitlab	Docker, Spack*	Clang-tools, readthedocs
2	SPECFEM3D	github	travis, github	Spack (SPECFEM3D-globe)	readthedocs
3	ExaHyPE	gitlab	gitlab, jenkins		LaTeX/PDF Guidebook
4	Tandem	github	travis		Clang-tools, readthedocs
5	xSHELLS	bitbucket	gitlab		
6	HySEA	github			
7	FALL3D	gitlab**			**
8	OpenPDAC	?	?	?	?
9	LaMEM	gitlab		Julia package	
10	PTatin3D	bitbucket			Doxygen, LaTeX/PDF manual
11	Elmer/Ice	Sourceforge***		Spack (elmerfem)	Wiki documentation

WP4-action items



How to cope with MS10 „Application deployment“ (M12)

Questions and Comments:

- **What can we achieve** with planed resources? **OR** **How much additional effort (PMs)** would we need dependent on expectations from Castiel?
- Will we get easy+free access to target platforms?
- HPC centers do not apply continuous software deployment operation model so far
- **Who pays for additional CPU hours** required for continuous deployment + testing?
- Do they expect to drop system specific optimizations – e.g. for GPGPUs – that contradicts goals of a call for Exascale!?
- CI on HPC – Security questions normally to be considered that will take time!
- **Do we need to rethink D4.2 „Plan for development and deployment tool usage“ (M20) to reflect MS10 (M12)?**

Suggestions:

- We select a (sufficient) set of codes and target systems
- Focus on easy and already in use solutions from the Flagship codes
- Rely on the common Spack package manager and/or Docker recipes

TODO:

- Ascertain Status of codes in terms of fast deployment with containers, etc.?
- Feedback from/to Castile about expectations?
- Set extra telcos beside WP4 monthly



WP4 action items

Administrative:

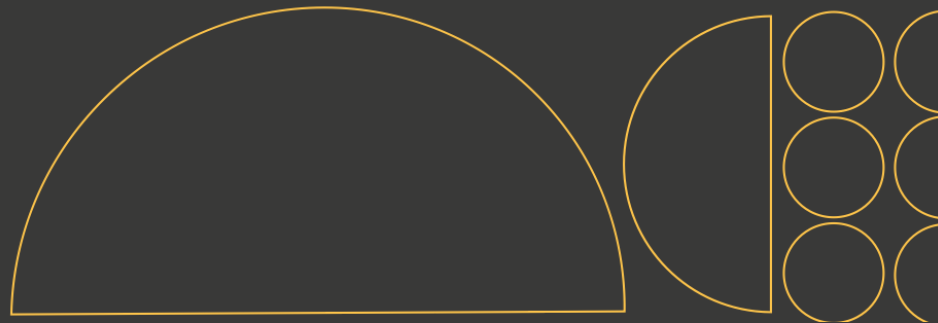
- Set up monthly WP4 TelCo meeting
- Create WP4 page in the ChEESE-2p wiki for all relevant infos

Technical:

- Create overview pictures describing the workflows for all PDs/SCs including all components, target platforms, data handling, ...
- Identify candidate workflows for performance checks
- Complete overview with the state of development, deployment, CI usage, documentation, ... for ChEESE codes
- Identify quality assessment tools for PDs/codes



Thank you!



@cheese-coe



<http://cheese2.eu>



@cheese-coe



@cheese_coe@techhub.social

