Generalitat de Catalunya
Departament de Recerca i Universitats

GOBIERNO DE ESPAÑA
MINISTERIO DE CIENCIA E INNOVACIÓN

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

UNIÓN EUROPEA
Fondo Europeo de Desarrollo Regional

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Simplifying the life-cycle management of complex application workflows

Rosa M Badia

12/6/2024

AEiC 2024 workshop, Barcelona

EuroHPC systems

		Status	Country	Peak performance	Architecture
Pre-Exascale	LUMI	Operational	Finland	539.13 petaflops	64-core AMD EPYC™ CPUs + AMD Instinct™ GPU
	Leonardo	Operational	Italy	315.74 petaflops	Intel Ice-Lake [illegible]phire Rap[illegible]
	MareNostrum 5	Operational	Spain	295.81 [illegible]	[illegible]IDIA [illegible]el
Petascale	Meluxina	Operational	[illegible]	[illegible]	[illegible] NVIDIA Ampere A100
	Vega	Operational	[illegible]	[illegible] petaflops	AMD Epyc 7H12 + Nvidia A100
	Karolina	Operational	[illegible]epublic	12.91 petaflops	AMD + Nvidia A100
	Discoverer	Operational	Bulgaria	5.94 petaflops	AMD EPYC
	Deucalion	Operational	Portugal	5.01 petaflops	A64FX, AMD EPYC, Nvidia Ampere

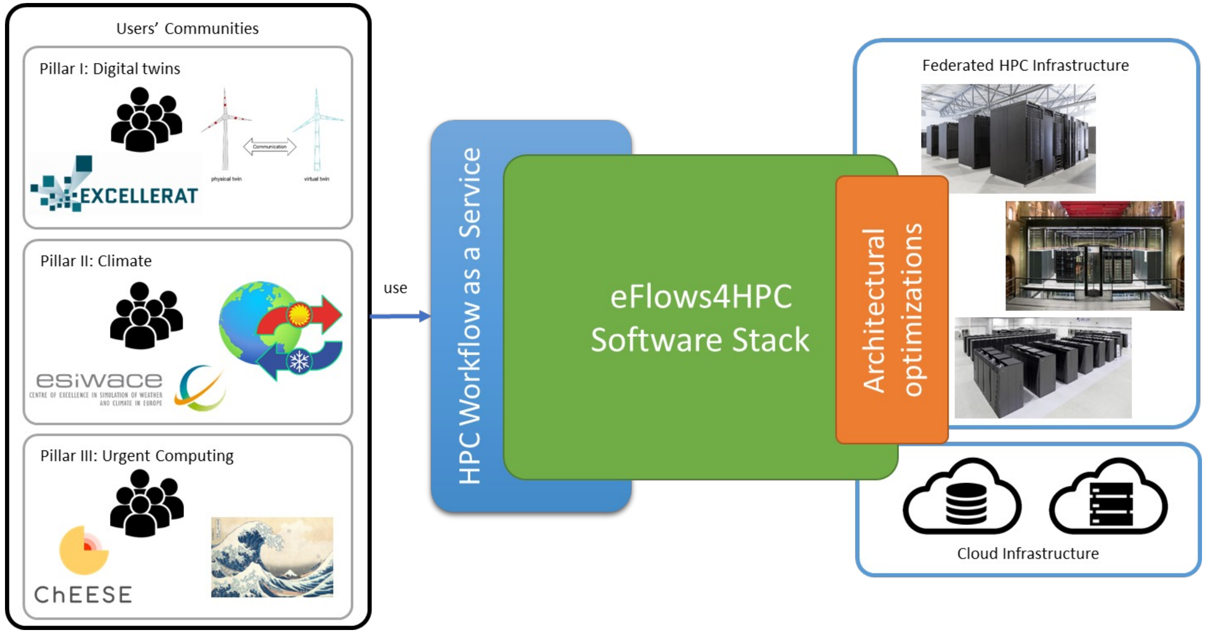
JUPITER, First European Exascale Supercomputer announced to be installed in Jülich

Centro Nacional de Supercomputación

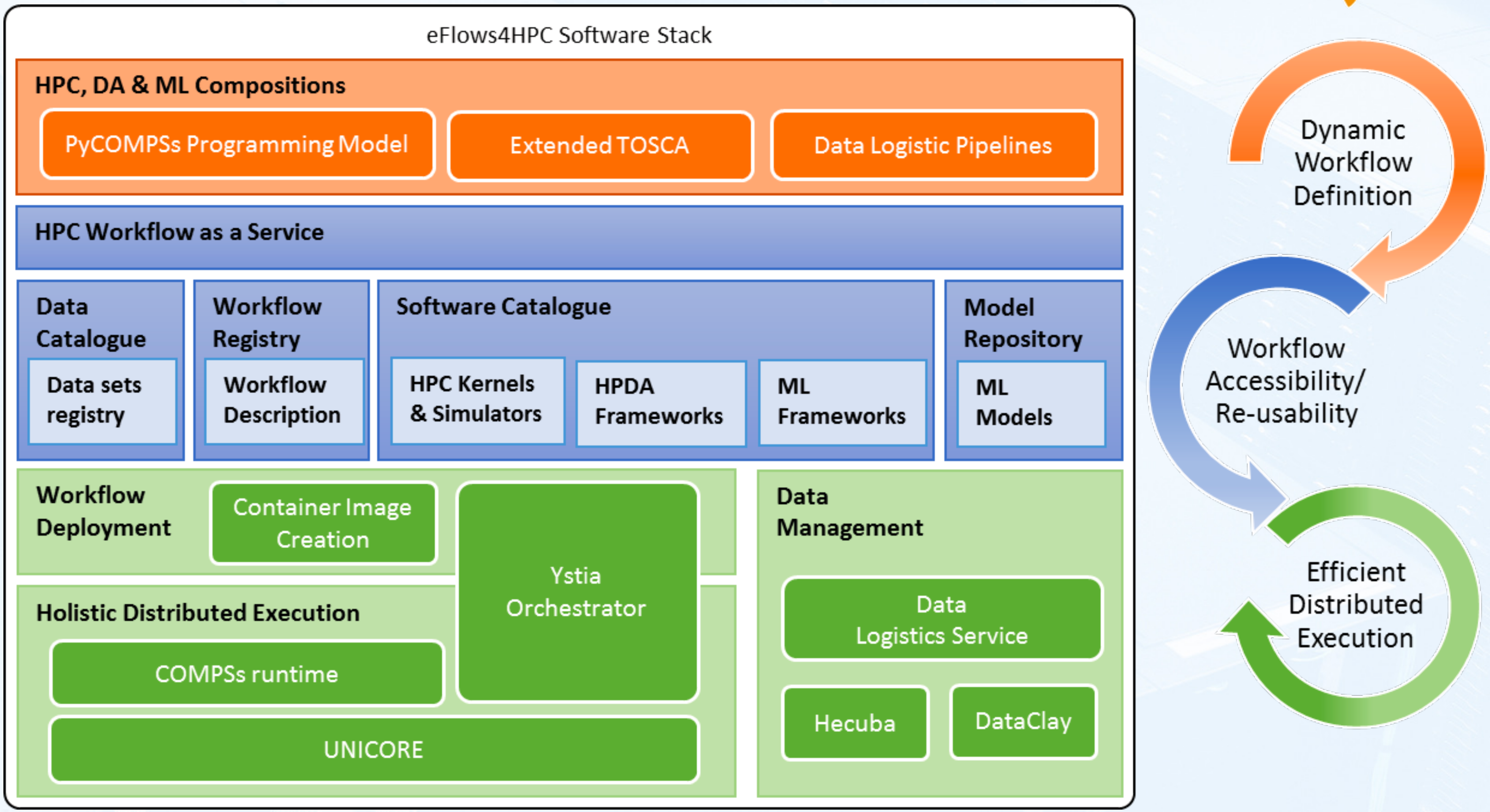
https://eurohpc-ju.europa.eu/about/our-supercomputers_en

eFlows4HPC in a nutshell

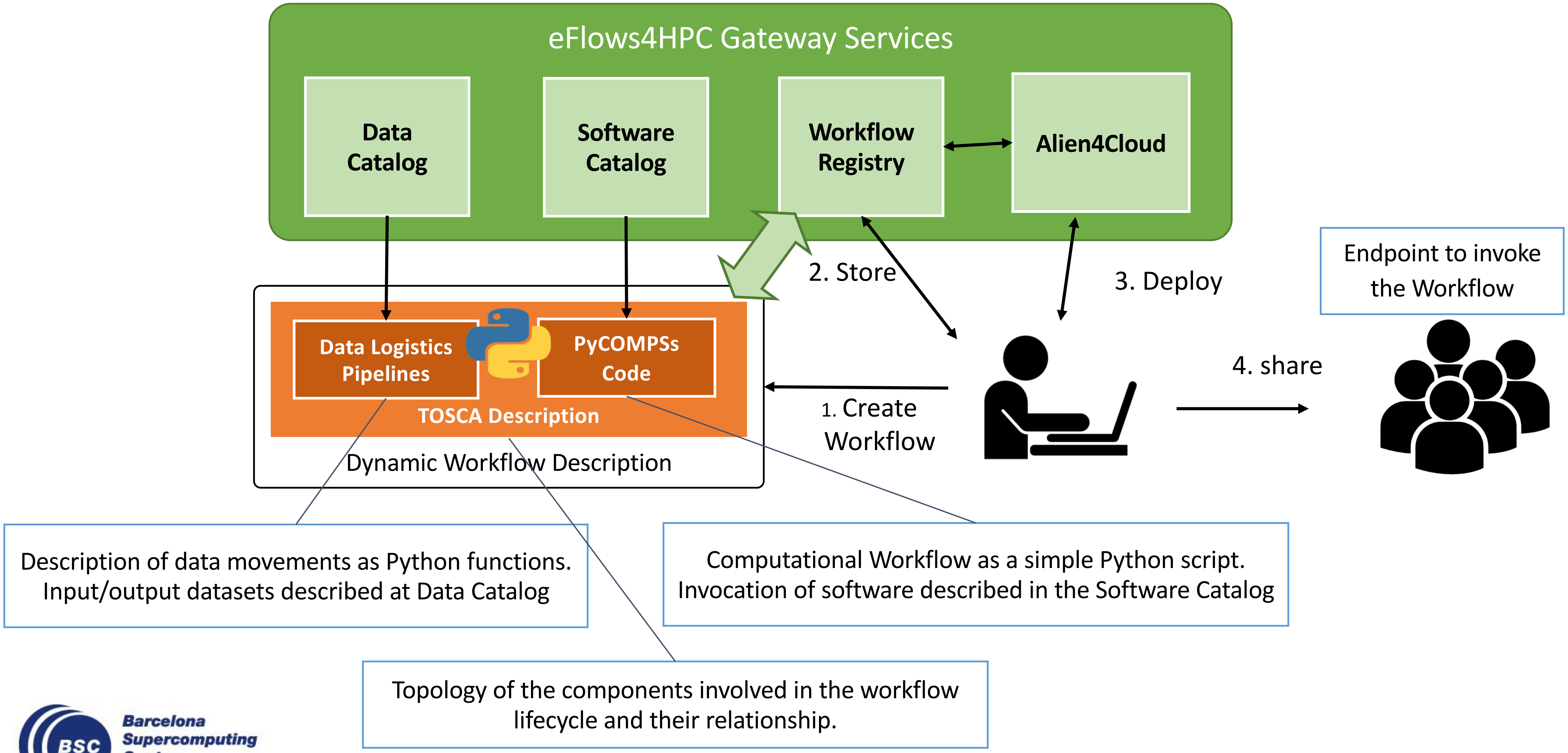
- Software tools stack that makes easier the development and management of complex workflows:
  - Combine different aspects
    - HPC, AI, data analytics
  - Reactive and dynamic workflows
    - Autonomous workflow steering
  - Full lifecycle management
    - Not just execution
    - Data logistics and Deployment
- HPC Workflows as a Service:
  - Mechanisms to make easier the use and reuse of HPC by wider communities
- Architectural Optimizations:
  - Selected HPC – AI Kernels Optimized for GPUs, FPGA, EPI
- Validation Pillar's
  - End-user workflows linked to CoEs



Barcelona Supercomputing Center
Centro Nacional de Supercomputación

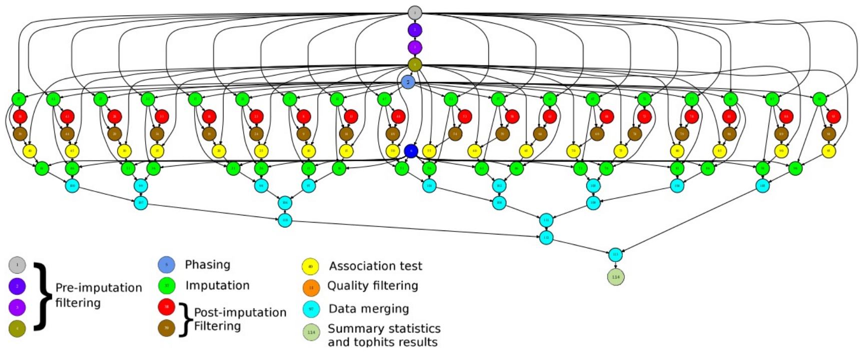

eFlows4HPC Software Stack
HPC, DA & ML Compositions
PyCOMPSs Programming Model
Extended TOSCA
Data Logistic Pipelines
HPC Workflow as a Service
Data Catalogue
Data sets registry
Workflow Registry
Workflow Description
Software Catalogue
HPC Kernels & Simulators
HPDA Frameworks
ML Frameworks
Model Repository
ML Models
Workflow Deployment
Container Image Creation
Ystia Orchestrator
Holistic Distributed Execution
COMPSs runtime
UNICORE
Data Management
Data Logistics Service
Hecuba
DataClay
Dynamic Workflow Definition
Workflow Accessibility/ Re-usability
Efficient Distributed Execution

Workflow development overview



Programming with PyCOMPSs/COMPSs

- Sequential programming, parallel execution
- General purpose programming language + annotations/hints
  - To identify tasks and directionality of data
- Builds a task graph at runtime that express potential concurrency
- Tasks can be complex, parallel, even MPI
- Offers a shared memory illusion to applications in a distributed system
  - The application can address larger data storage space: support for Big Data apps
- Agnostic of computing platform
- Provenance recording
- Syntax extended to better integrate AI and HPDA



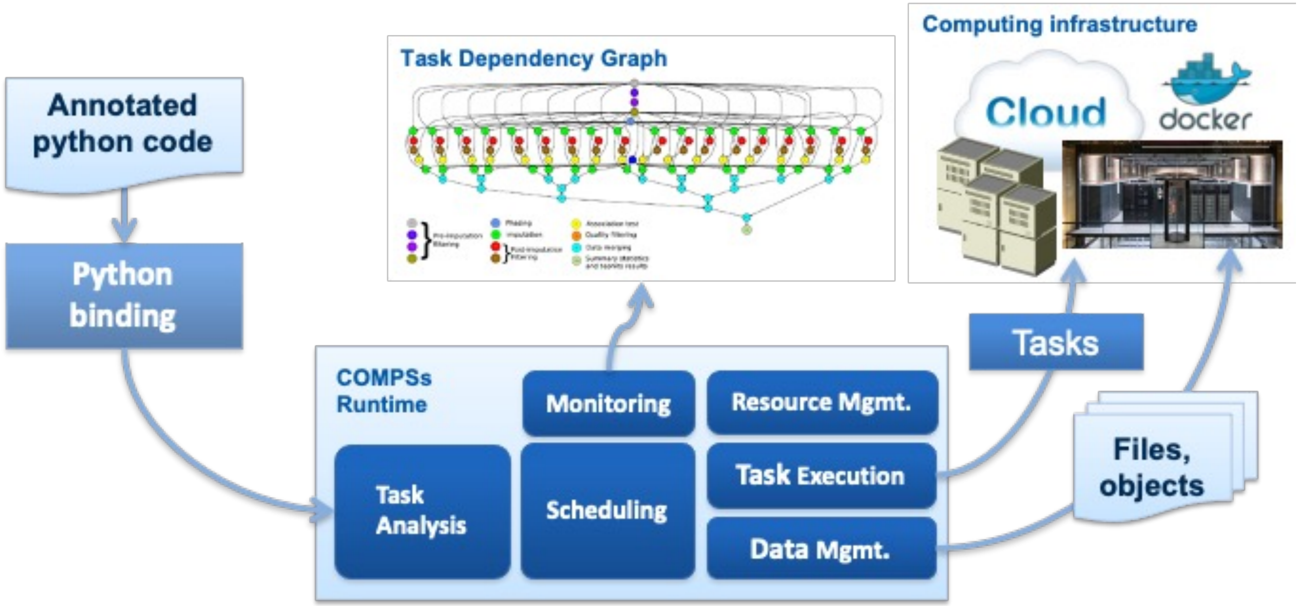

```
@task(c=INOUT)
def multiply(a, b, c):
        c += a*b
```

```
initialize_variables()
startMulTime = time.time()
for i in range(MSIZE):
    for j in range(MSIZE):
        for k in range(MSIZE):
            multiply (A[i][k], B[k][j], C[i][j])
compss_barrier()
mulTime = time.time() - startMulTime
```

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

PyCOMPSs features and runtime

- PyCOMPSs/COMPSs applications executed in distributed mode following the master-worker paradigm
  - Description of computational infrastructure in an XML file
- Sequential execution starts in master node and tasks are offloaded to worker nodes
- All data scheduling decisions and data transfers are performed by the runtime
- All data scheduling decisions and data transfers are performed by the runtime
- Support for elasticity



Heterogeneous Tasks

- A task can be more than a sequential function
  - A task in PyCOMPSs can be sequential, multicore or multi-node
  - External binary invocation: wrapper function generated automatically (@binary)
  - Supports for alternative programming models: MPI (@mpi)
- Can be combined with other decorators
  - @constraint: To indicate amount of memory, number of processors or GPUs per binary or MPI process
  - @container: When software is distributed as a container

```
@binary(binary="app.bin" args="—in {{f_in}} -out {{f_out}})
@task(f_in=FILE_IN, f_out=FILE_OUT)
def app_task(f_in, f_out):
    pass
```

```
@container(engine='SINGULARITY', image="/path/to/app.sif")
@binary(binary="app.bin" args="—in {{f_in}} -out {{f_out}})
@task(f_in=FILE_IN, f_out=FILE_OUT)
def app_task(f_in, f_out):
    pass
```

```
@constraint(processors=[{'processorType':'CPU','computingUnits':'1'},
                        {'processorType':'GPU', 'computingUnits':'1'}])
@task(returns=1)
def func(a, b, c):
    ...
    return result
```

```
@constraint (computingUnits= "8")
@mpi (runner="mpirun", processes= "16", ...)
@task (returns=int, stdOutFile=FILE_OUT_STDOUT, ...)
def nems(stdOutFile, stdErrFile):
    pass
```

Supercomputing Center
Centro Nacional de Supercomputación

Failure management

- Interface than enables the programmer to give hints about failure management

```
@task(file_path=FILE_INOUT, on_failure='CANCEL_SUCCESSORS',
time_out='$task_timeout')
def task(file_path):
    ...
    if cond :
        raise Exception()
```

- Options: RETRY, CANCEL_SUCCESSORS, FAIL, IGNORE
- Implications on file management:
  - i.e, on IGNORE, output files are generated empty
- **Possibility of ignoring part of the execution of the workflow, for example if a task fails in an unstable device**
- **Opens the possibility of dynamic workflow behaviour depending on the actual outcome of the tasks**

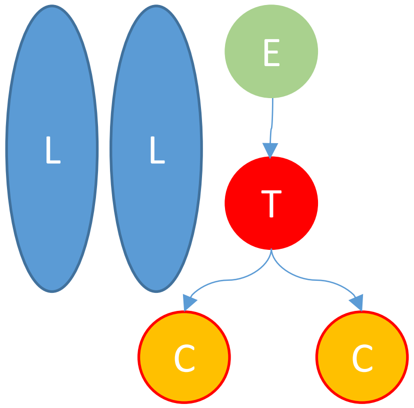
Timeouts and exceptions

- Timeouts can be defined for a task

```
@task(file_path=FILE_IN, time_out=200)
def time_out_task (file_path):
    ...
```

- Tasks can raise exceptions

```
@task(file_path=FILE_INOUT)
def comp_task(file_path):
    ...
    raise COMPSsException("Exception
raised")
```



- Combined with groups of tasks enables to cancel the group of tasks on the occurrence of an exception

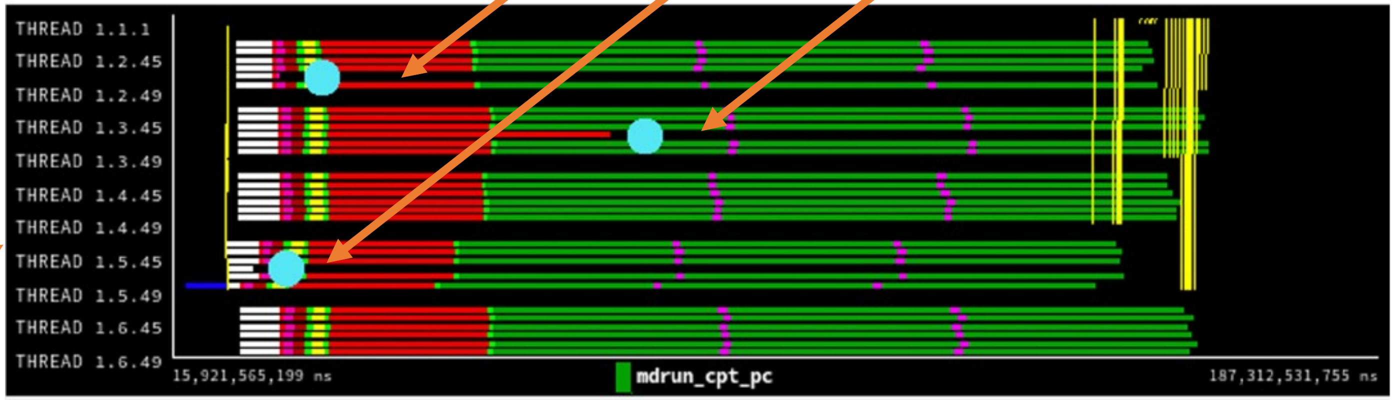
```
def test_cancellation(file_name):
    try:
        with TaskGroup('failedGroup'):
             long_task(file_name)
             long_task(file_name)
             executed_task(file_name)
             comp_task(file_name)
             cancelledTask(FILE_NAME)
             cancelledTask(FILE_NAME)

    except COMPSsException:
        print("COMPSsException caught")
        write_two(file_name)
```

Validation Example

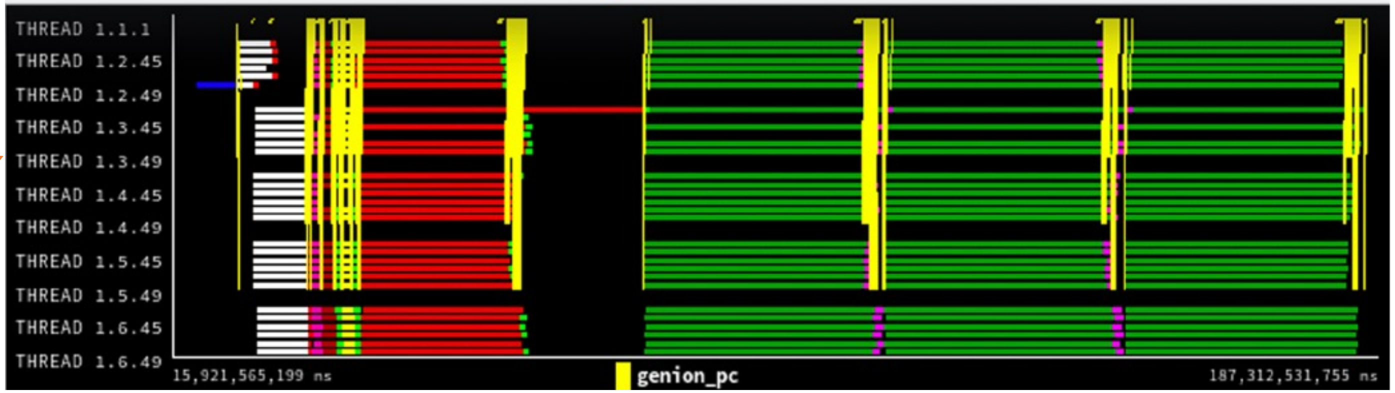
- Protein Mutants workflow from BioBB workflows
- Three types of failures
  - incorrect data,
  - incorrect SW configuration,
  - longer execution time
- On_failure = CANCEL_SUCCESSORS

Successor tasks are cancelled

COMPSs managing failures

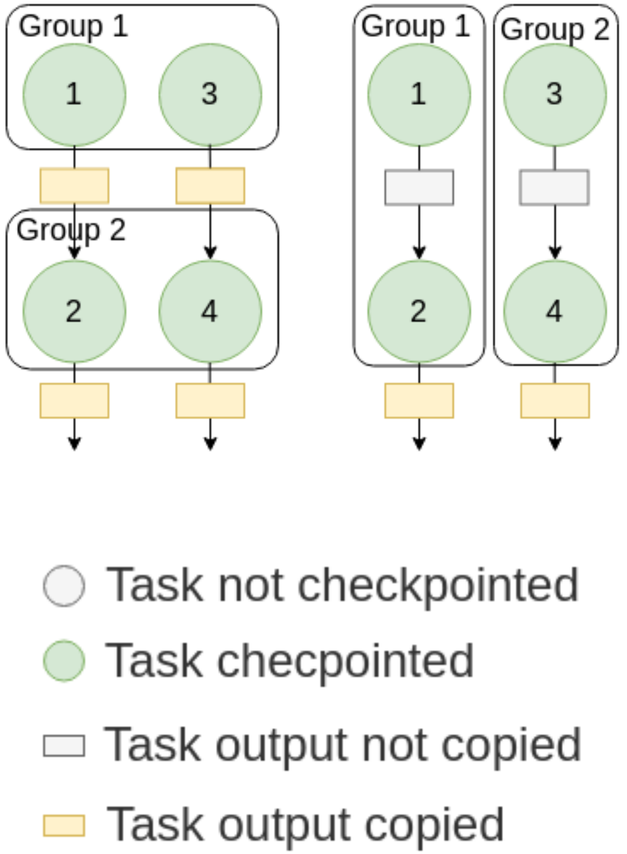


Application managing failures



Checkpointing

- Allows the workflow re-execution avoiding the re-execution of finished tasks
- Asynchronous but with some overhead
  - Save tasks results in a persistent storage
  - Trade-off between performance and time to recover
  - Establishing the right checkpoint granularity is important
- 3 mechanisms for automatic checkpointing
  - **Time:** periodically, COMPSs saves the last version produced for every value
  - **Finished tasks** : after the completion of X tasks, COMPSs saves the last version produced for every value
  - **Instantiation task groups**: Defines groups of tasks, COMPSs saves those data versions that are final results for the group `compss_snapshot()`
- Indicated by the developer with API
- No checkpoint inside the task: Drawback for very large tasks.
  - Possible integration with internal checkpointing



Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Checkpointing Overhead

- Benchmarks:
  - K-means clustering
  - PMXCV19, bio workflow that evaluates changes in the binding affinity between SARS-Cov-2 spike protein and Human ACE2 receptor.
  - Principle Component Analysis (PCA)
- Policies:
  - Instantiated Tasks Groups - ITG (Grouping every 10 instantiated tasks)
  - Finished Tasks – FT (Every 10 finished tasks)
  - Periodic Time - PT (15 seconds interval)

	NC	ITG	FT	PT
K-Means	220.12 s	222.56 s (1%)	229.34 s (4.5%)	228.44 s (4%)
PMXCV19	33 m	33.9 m (2.7%)	34.1 m (3.3%)	33.6 m (1.8%)
PCA	883.13 s	1075.13 s (21.7%)	1026.21 s (16.1%)	1284.07 s (45.4%)

Checkpointing Overhead

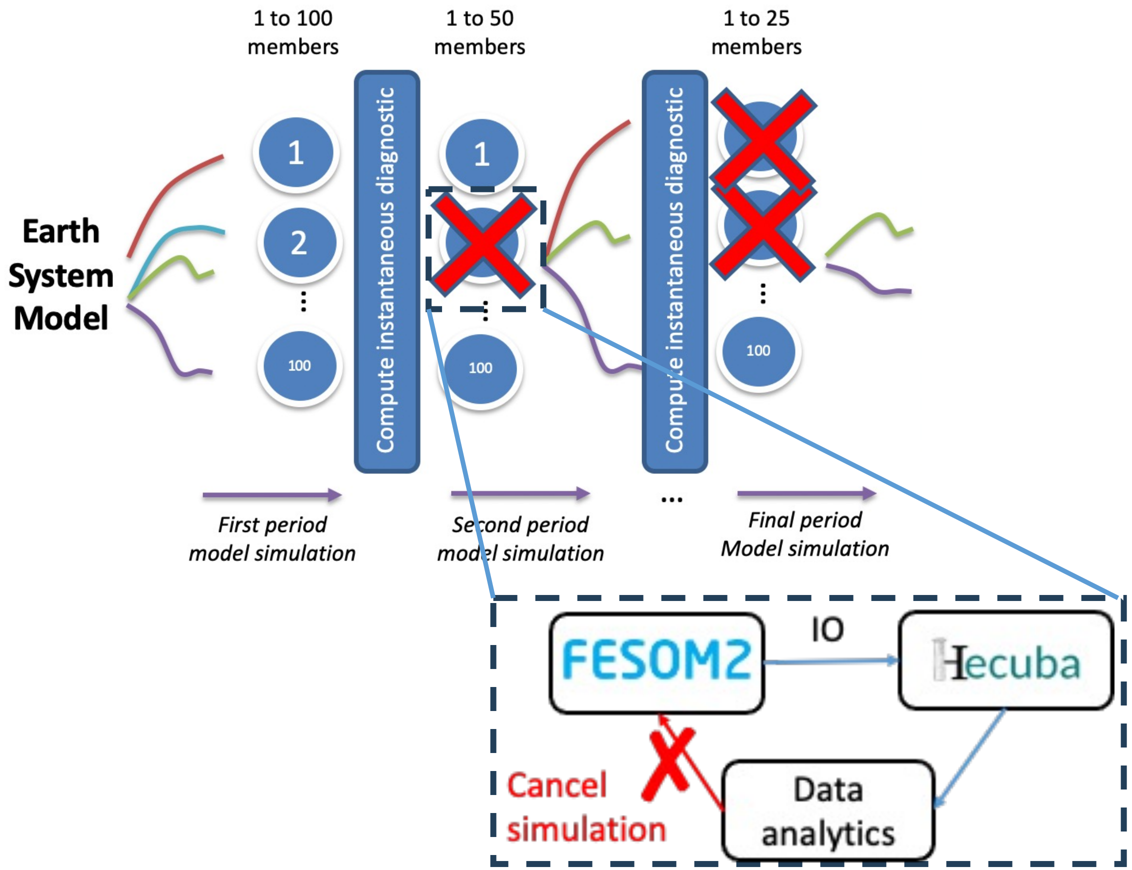
- Importance on the policy and frequency choices depending on the application

- Policies:
  - Instantiated Tasks Groups - ITG (Grouping every 10, 50, 100 instantiated tasks)
  - Periodic Time - PT (15, 30, 60 seconds interval)
  - Finished Tasks – FT (Every 10, 40, 100 finished tasks)

	Fine-grain	Medium-grain	Coarse-grain
Kmeans (ITG)	222.56 s (1.1%)	244.25 s (11%)	264.36 s (20%)
PMXCV19 (PT)	33.6 m (1.8%)	33 m (0%)	33.1 m (0.3%)
PCA (FT)	1026.21 s (16.1%)	1016.20 s (15%)	1103.94 (24.9%)

DA-driven ensemble member pruning

Earth-System Model (ESM) workflow



Task-groups and **exceptions** used to dynamically prune ensemble members based on data analytics

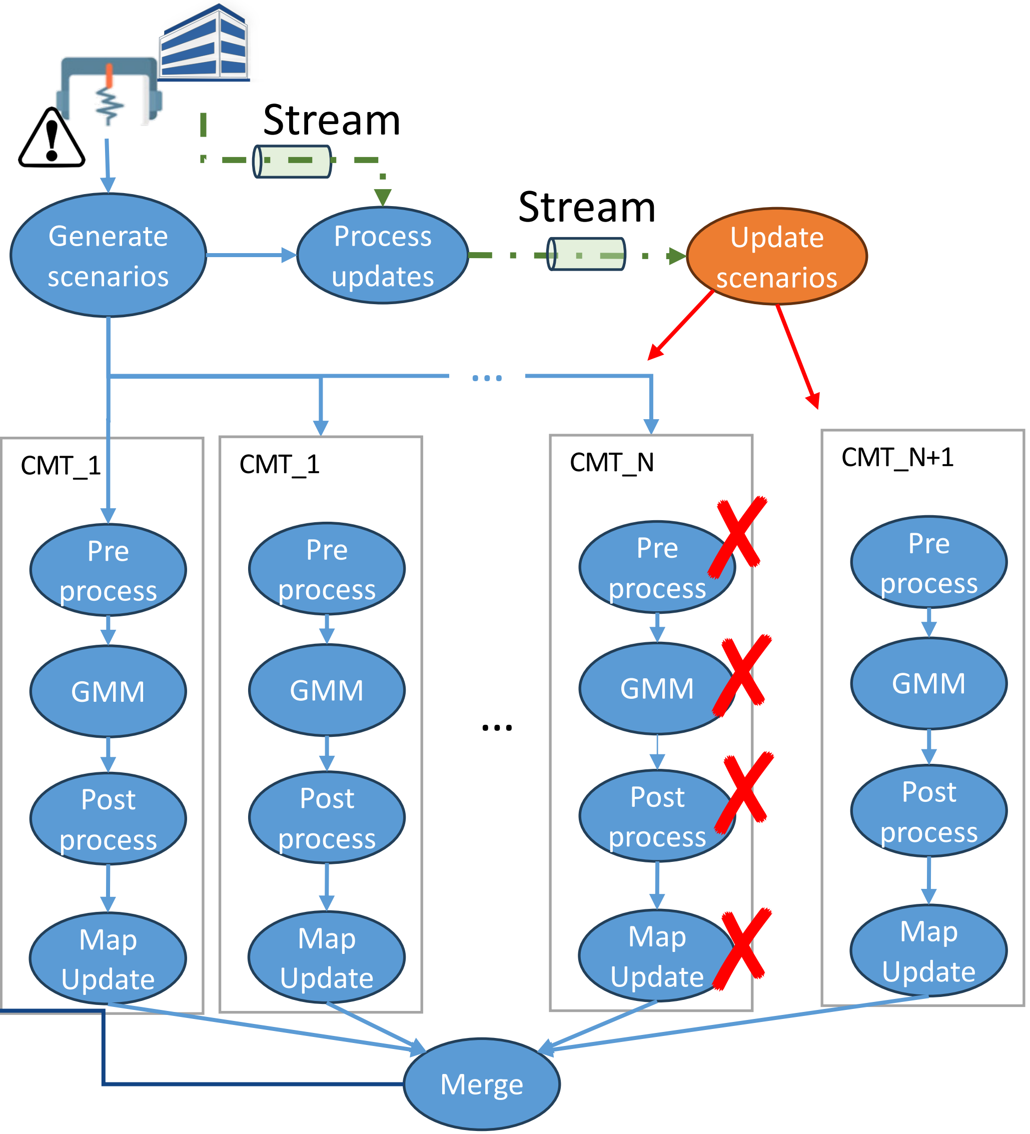
Hecuba support for a lambda architecture, allowing both batch processing and stream processing

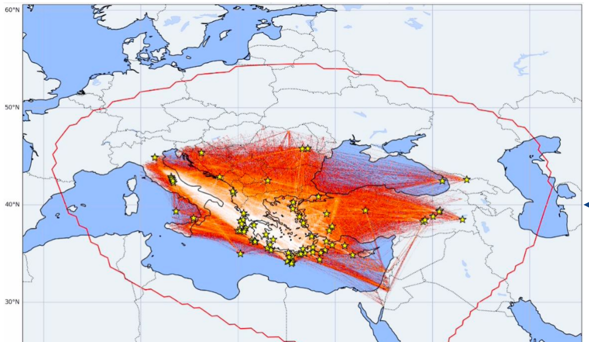
Event-driven cancellation/creation

eFlows4HPC

UCIS4EQ: HPC-based urgent seismic simulation workflow

- Evaluation of scenarios after the occurrence of a seismic event
- Combines multiple web services and HPC simulation (Salvus)
- Workflow Dynamicity:
  - Usage of **data streaming** for communication of events
  - On event occurrence API supports:
    - **Dynamic cancellation** of task groups
    - **Dynamic creation** of new set of tasks

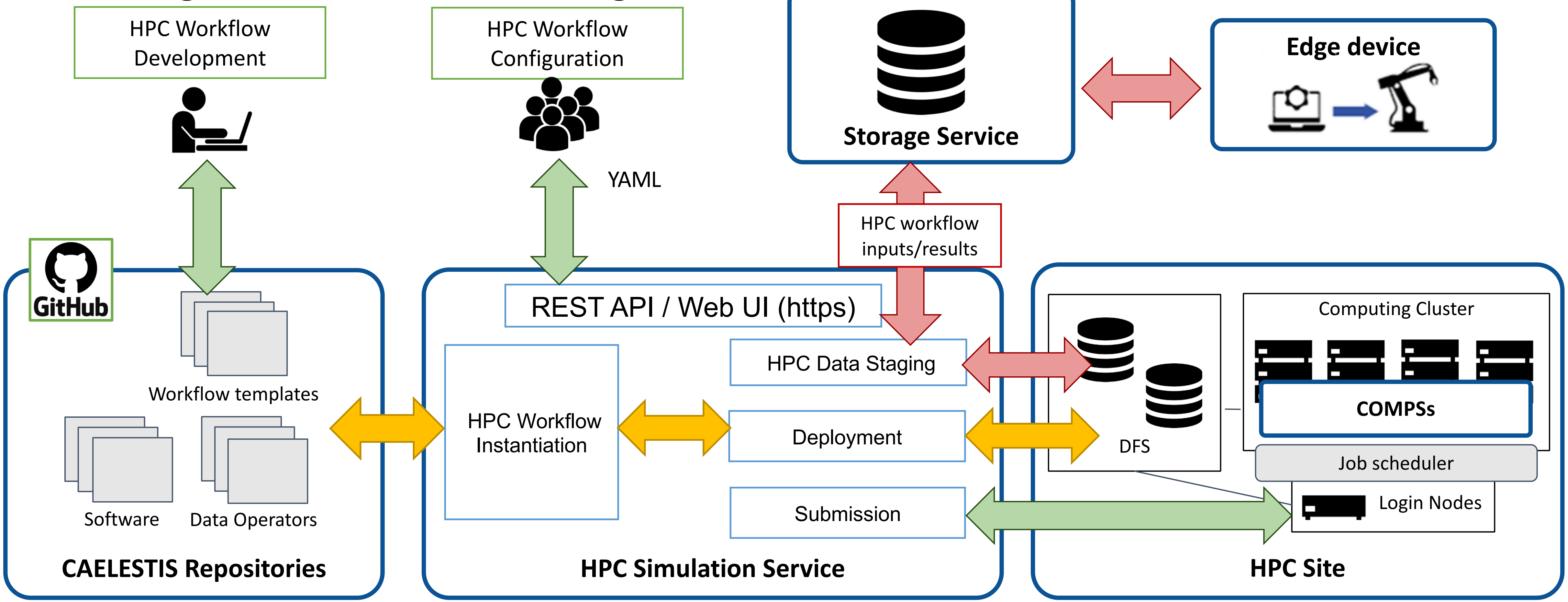




Barcelona Supercomputing Center
Centro Nacional de Supercomputación

CAELESTIS Simulation Ecosystem Architecture

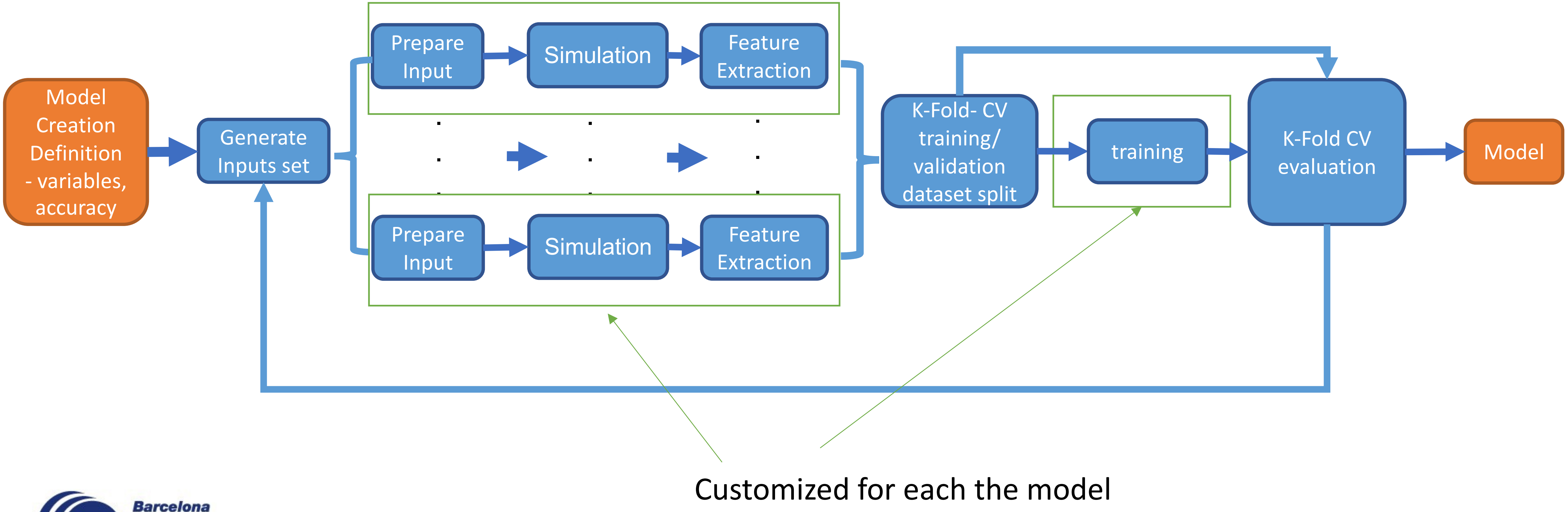
Towards a digital twin for aircraft design



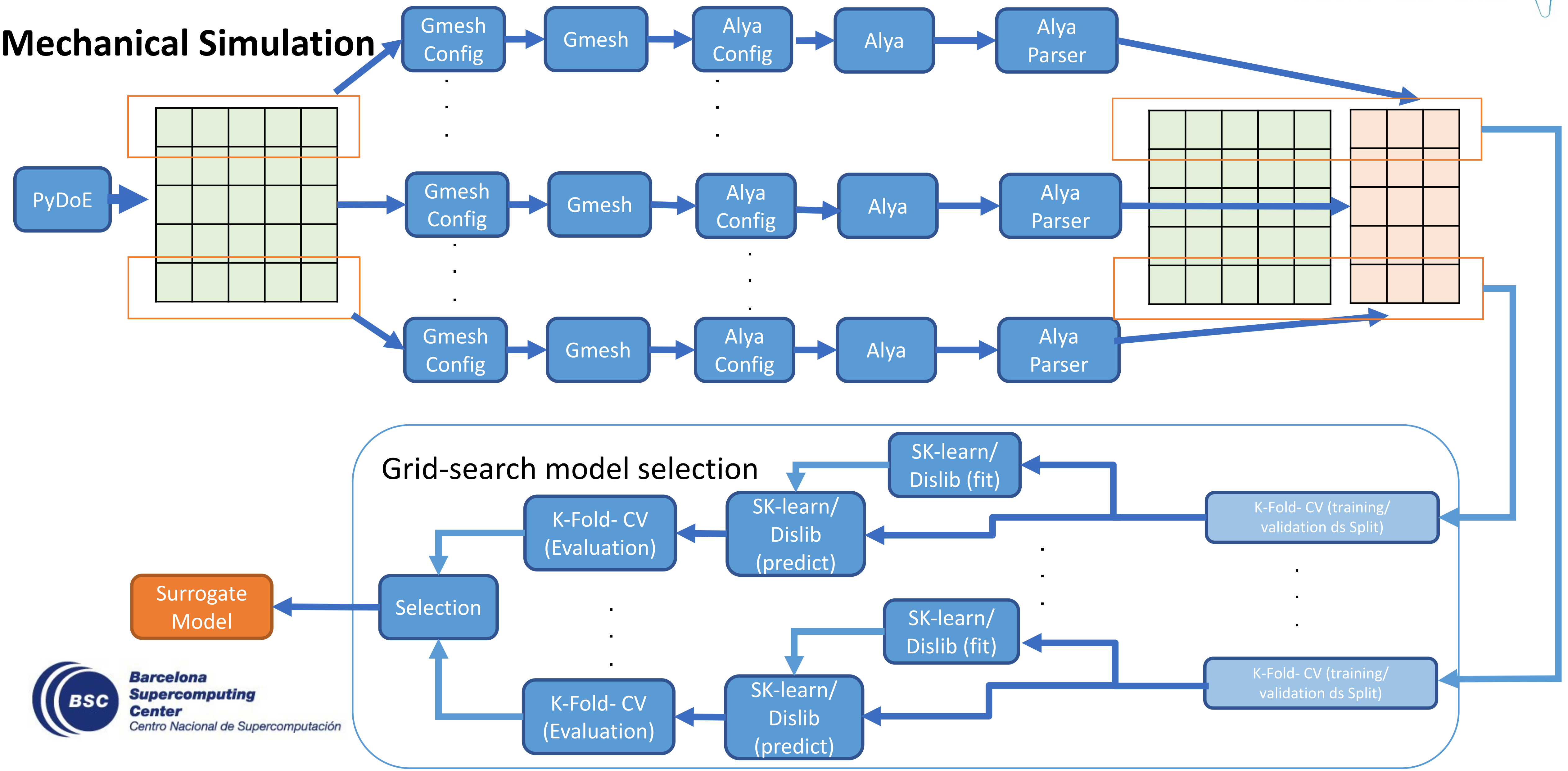
Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Workflow templates

Surrogate Model Creation Workflow



Specific workflow instance

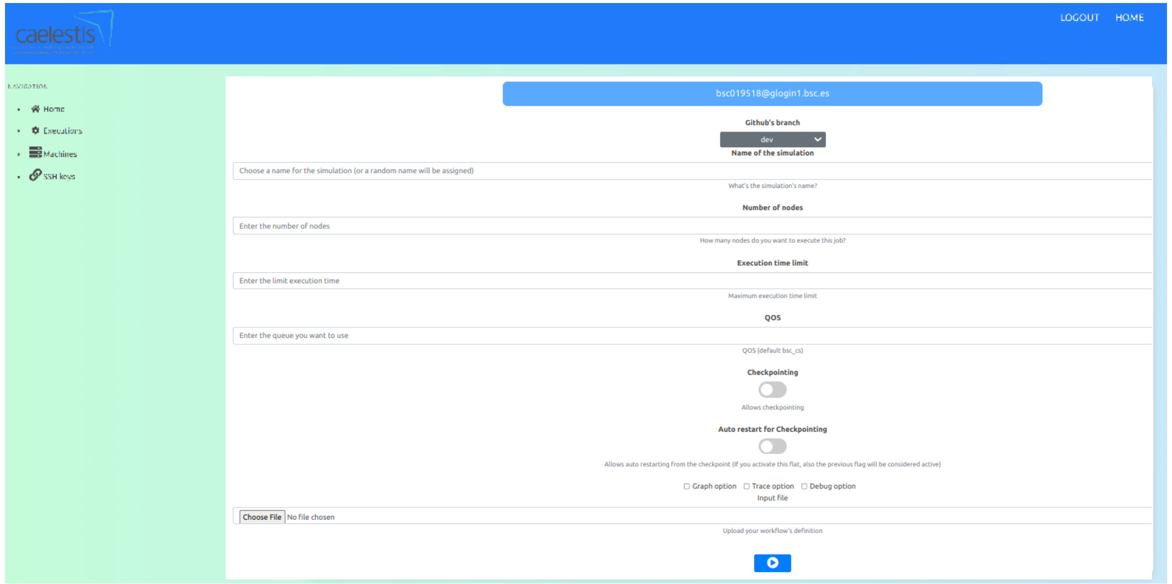


Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Checkpoint management

caelestis

- CAELESTIS service offers two management options of checkpointed jobs:
  - Automatic handling: If job terminates due to time constraint, the service reinitiates the workflow submission, which is rerun from the checkpoint. This process can continue until the job concludes successfully.
  - Manual handling: users have the option to handle checkpoints manually. If a job ends due to a time constraint, it will appear in the execution dashboard under the ”timeout jobs” list. Users can then initiate a new job from the checkpoint.



Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Final thoughts

- Coarse-grained task-based programming models provide suitable environments for developing HPC + AI workflows
  - eFlows4HPC
  - CAELESTIS
  - DT-GEO
- These programming environments should provide means to guarantee resiliance in the workflows' executions
- Fault tolerance and exceptions mechanisms can be used to provide maleability and dynamicity to the workflow applications
- Checkpointing provides resiliance, but also mechanisms to avoid job execution constraints

Further Information

- Project page: http://www.bsc.es/compss
  - Documentation
  - Virtual Appliance for testing & sample applications
  - Tutorials
- Source Code
  - https://github.com/bsc-wdc/compss
- Docker Image
  - https://hub.docker.com/r/compss/compss
- Applications
  - https://github.com/bsc-wdc/apps
  - https://github.com/bsc-wdc/dislib
- Dislib
  - https://dislib.readthedocs.io/en/latest/

Barcelona
Supercomputing
Center
Centro Nacional de Supercomputación

ACKs

DT-GEO

CEEC

Colmena

HP2C-DT

CyclOps

DISCOVER US

CAELESTIS

ICOS

Joint Laboratory for Extreme-Scale Computing

Inria NCSA Argonne BSC JÜLICH RIKEN

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Generalitat de Catalunya
Departament de Recerca i Universitats

MINISTERIO DE CIENCIA E INNOVACIÓN

Financiado por la Unión Europea
NextGenerationEU

Plan de Recuperación, Transformación y Resiliencia

AGENCIA ESTATAL DE INVESTIGACIÓN

MareNostrum 5

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

rosa.m.badia@bsc.es

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Thanks!

rosa.m.badia@bsc.es