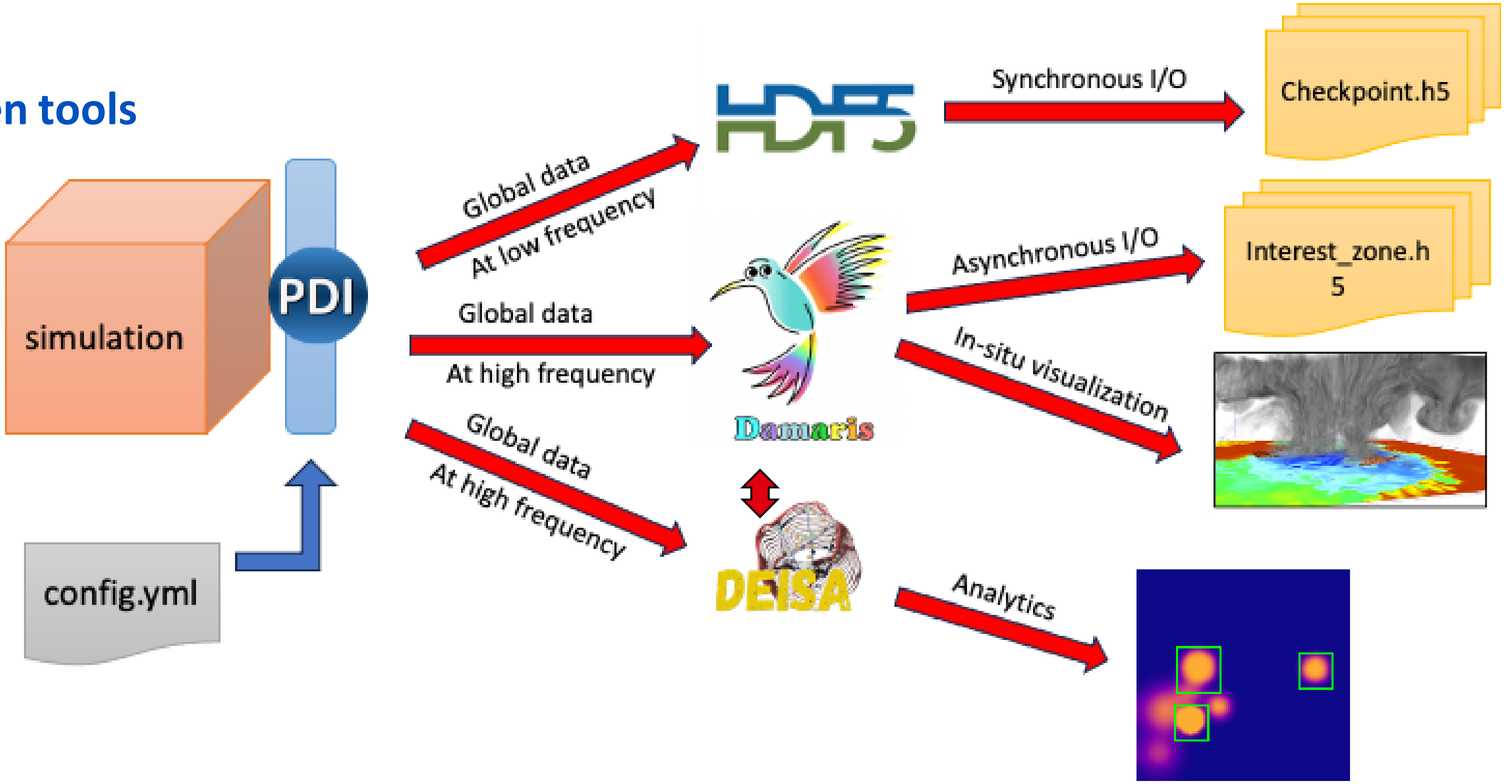
WP2 – Exascale In situ Processing

Design and implement the software building blocks required to support in-situ execution of data processing at Exascale, and to integrate these building blocks in the libraries deliverables of Exa-DoST.

Interaction between tools



WP2 - Current Participants

Partner	Type of position	Name of participant
Inria Rennes	Researchers	Gabriel Antoniu, Silvina Caino-Lores
	Engineers	Etienne Ndamlabin
	PhD student	Arthur Jaquard
Inria Grenoble	Researchers	Bruno Raffin
	Engineers	Andres Bermeo-Marinelli, Hugo Strappazzon
CEA - Maison de la simulation	Researchers	**Yushan Wang**, Julien Bigot, Benoît Martin
	Engineers	Jacques Morice, Julian Auriac, 1 open position
	PhD student	Ivan Lucas, 1 open position
CEA DAM	Researchers	**Laurent Colombet**, Christophe Denoual

8 permanent stuff
5 engineers
2 PhD

WP2 - Achievements

Software production

- Damaris
  - Improvements to Damaris to make it easier to install and use : Version v1.12.x
  - An interface with PDI is available
  - Asynchronous in-situ exchange capability for analytics and I/O
- PDI
  - New plugins are available for enabling connections with a wider range of tools : Release 1.9.x
  - Add more advanced data types and incorporate the “GPU-aware” notion into the PDI
- Deisa
  - Deisa has been redesigned to accommodate planned new developments and remove inappropriate dependencies on Dask

Scientific Dissemination

- 2 internship report
- 2 posters (ISC, Teratec)
- Multiple talks (Riken, ISC25)
- 2 training sessions
- Advanced Tutorial in HPCAsia

External Collaborations

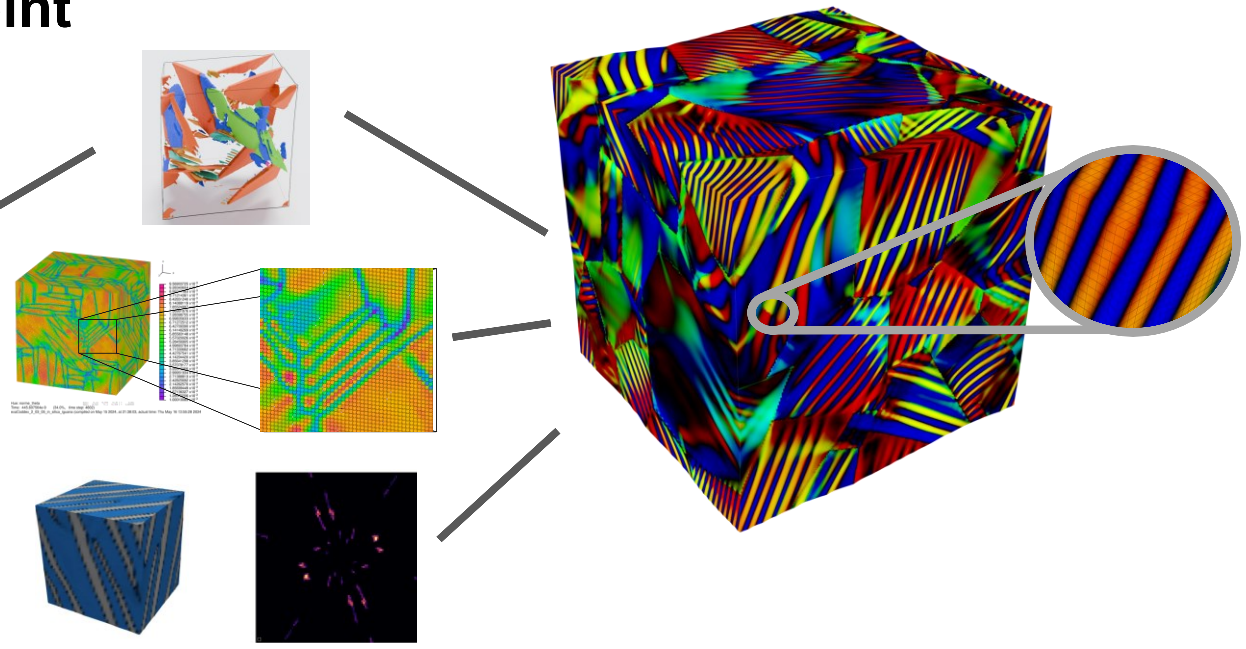
- Initial contacts for collaboration with Riken, Japan
- PDI is well adopted by TRUST (CEA)

WP2: Highlights

Coddex

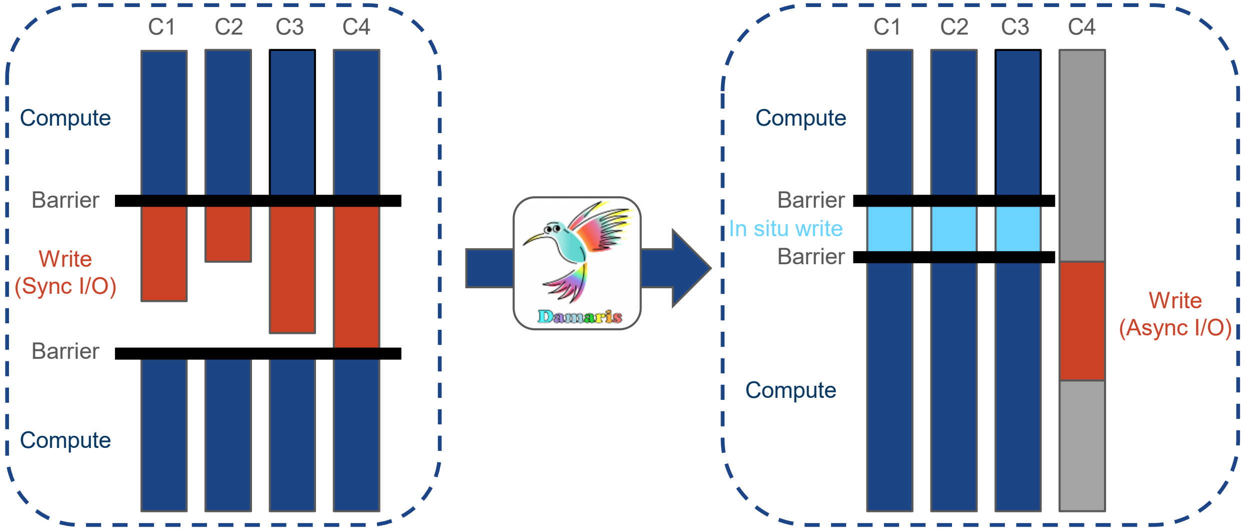
Code for Discontinuous Deformation Evolution in Xstals

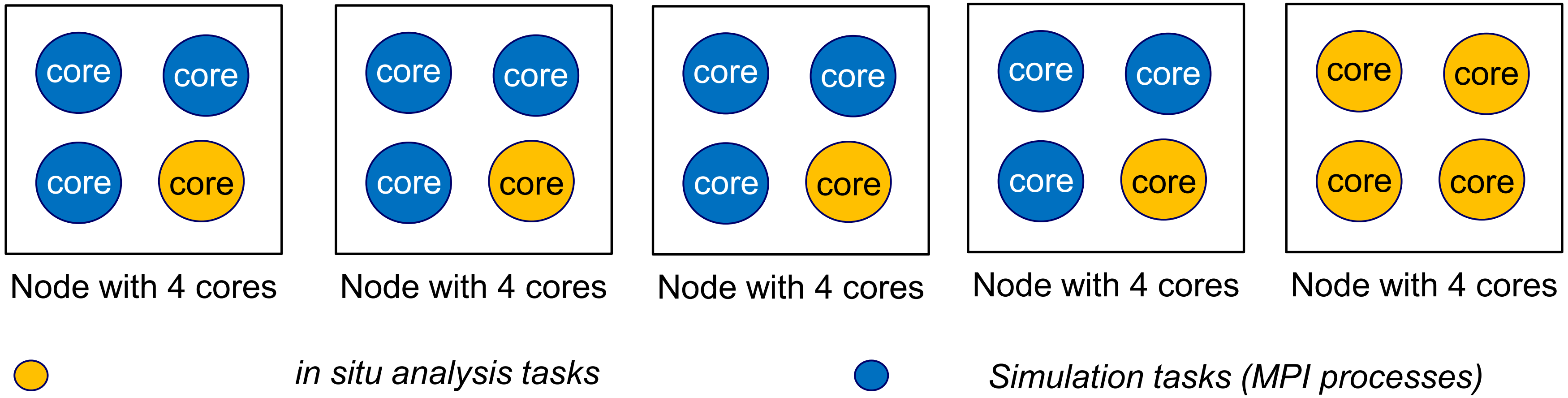
- **MPI-based simulation (+ OpenMPI)**
- **Hundreds of variables per data point**
- **Multiple data-analysis pipelines**
  - Direct output of some variables for later analysis (e.g. Paraview)
  - Projection output of a given variable as Postscript image
  - Compute and render X-ray diffraction within the crystal



WP2: Highlights

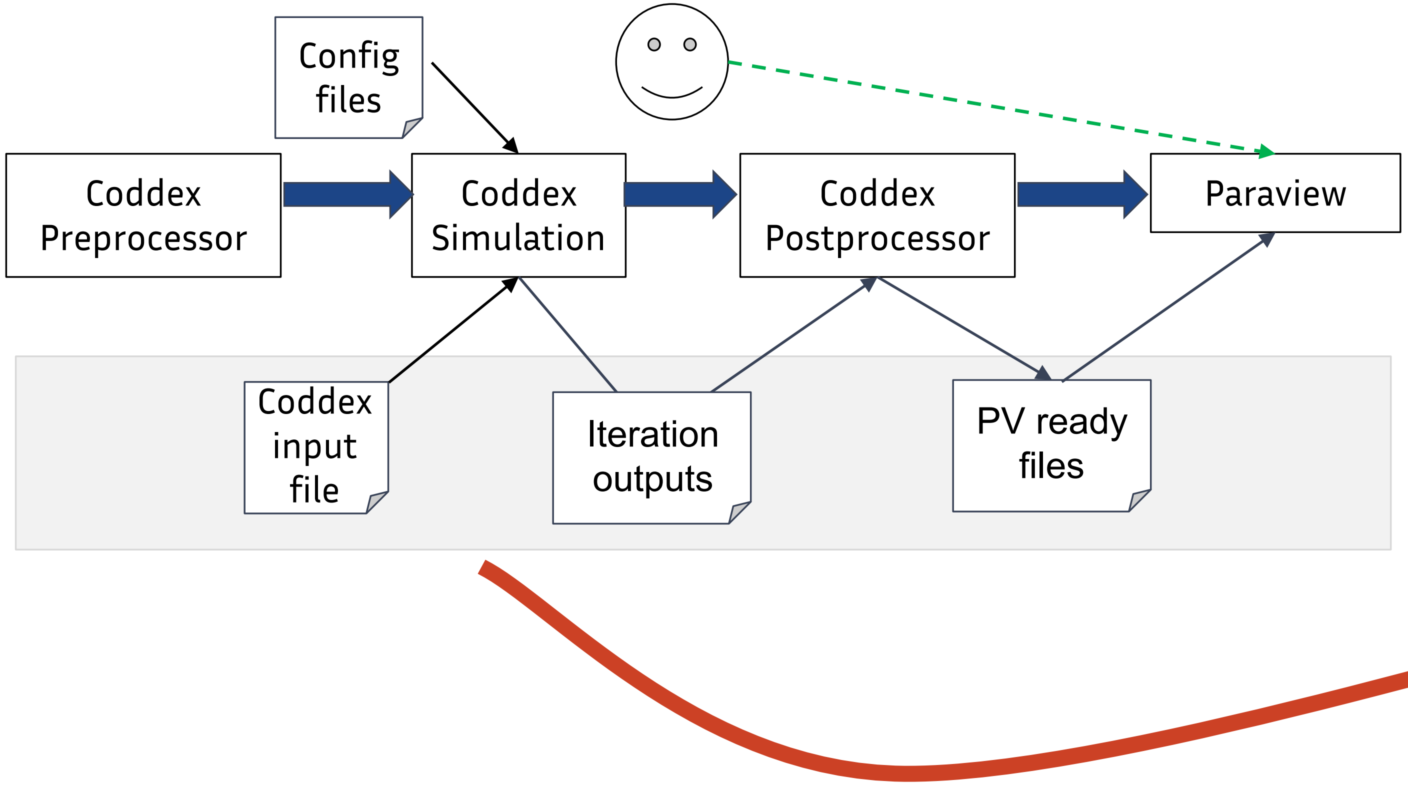
Damaris

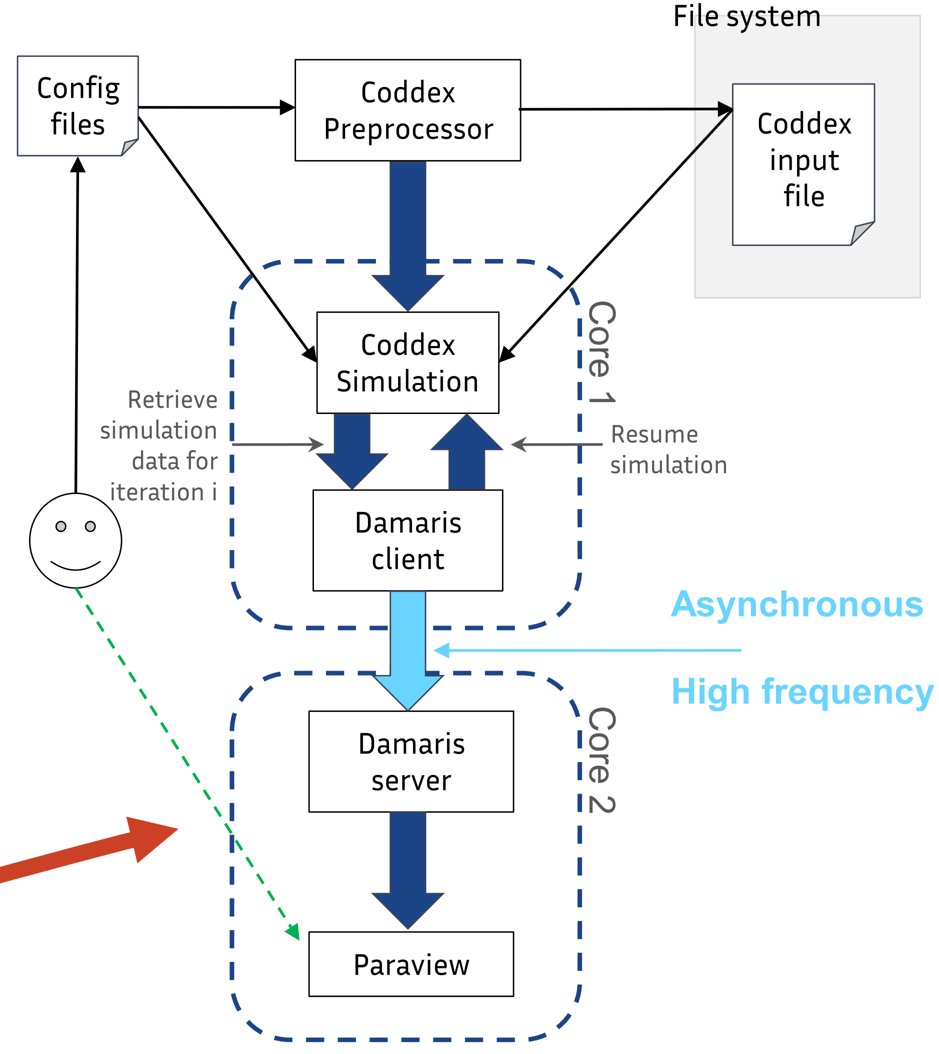




WP2: Highlights

Integrating Damaris in a Coddex visualization workflow





WP2: Highlights

Integrating Damaris in a Coddex visualization workflow - Evaluation

- Tin hysteresis simulation (of size $n^3 = \{1, 8, 27\}$ for $n = \{1, 2, 3\}$)
- Output data every 10 iterations
  - **I/O scenario:** write raw variable values to disk
  - **In situ scenario:** create PV visualization image in situ, then write image to disk
- #output variables = {5, 50}

Deployment settings

I/O scenario

Simulation of size n^3

- n^3 MPI processes for Coddex
- 64 cores per Coddex process (for 64 OpenMPI threads)
- CPU -> Coddex*64

In situ scenario

Simulation of size n^3

- n^3 MPI processes for Coddex
- 63 cores per Coddex process (for 63 OpenMPI threads)
- CPU -> Coddex*63 + Damaris*1

Execution time per iteration

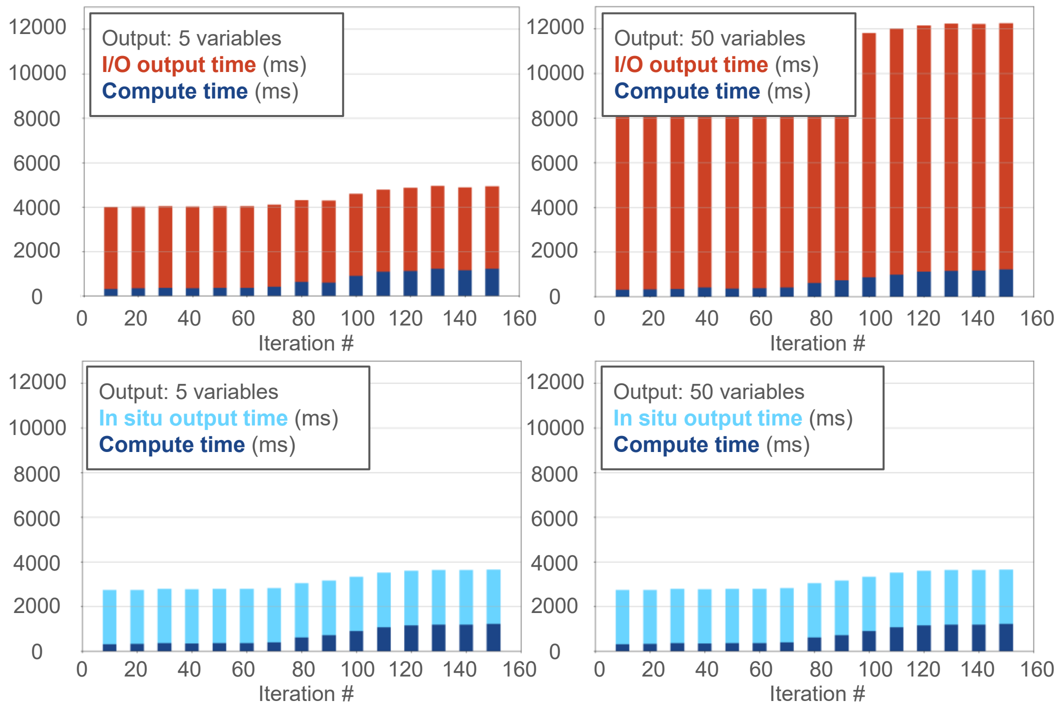


Figure 1: Execution time per iteration (in ms) for the I/O scenario (top) and in situ scenario (bottom), for respectively 5 output variables (left) and 50 output variables (right). 14 nodes ~ 1728 cores ($n^3 = 27$)

Average iteration output time

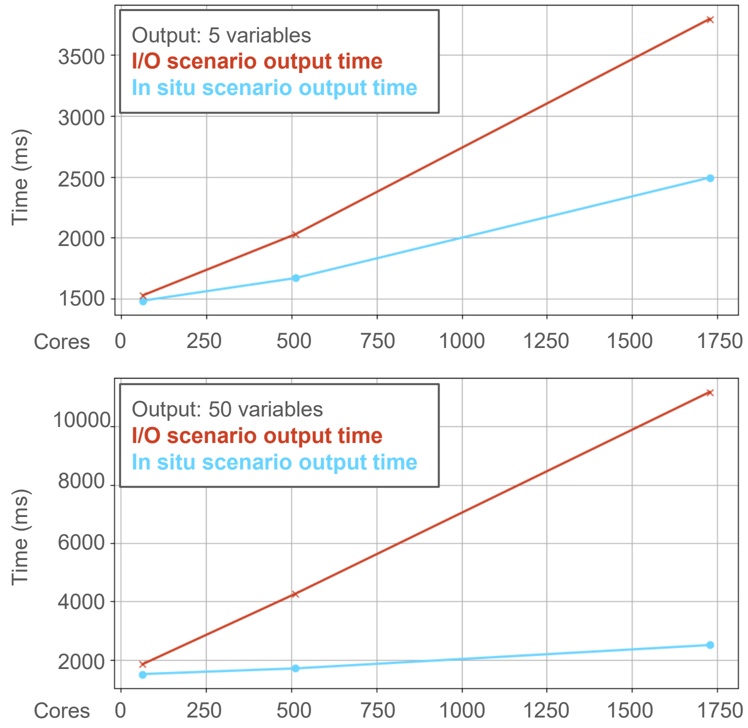


Figure 2: Mean iteration output time (in ms) for I/O scenario and in situ scenario, for respectively 5 output variables (top) and 50 output variables (bottom).

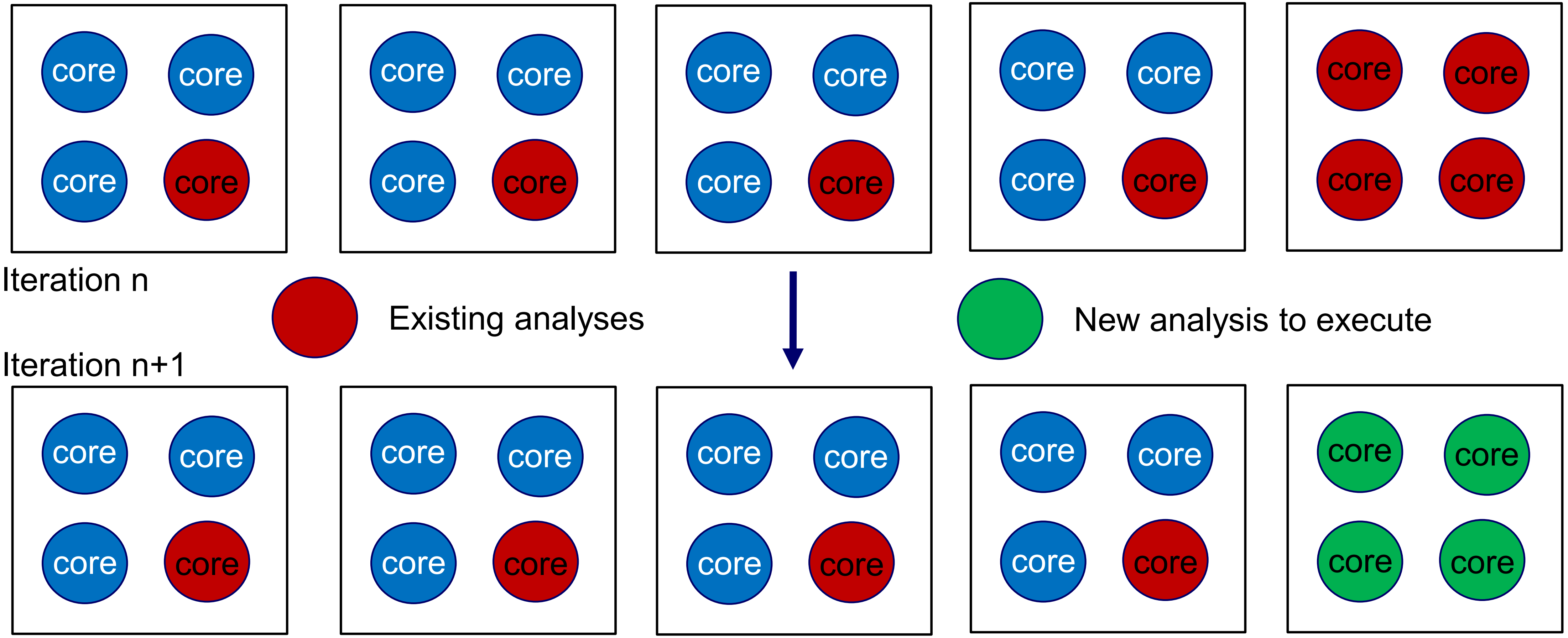
WP2: Highlights

Damaris for dynamic in situ analyses

- Handling analyses whose execution is decided at runtime
- Coddex as a demonstrator
- Ongoing work
    - Reconfiguration of analyses placement at runtime

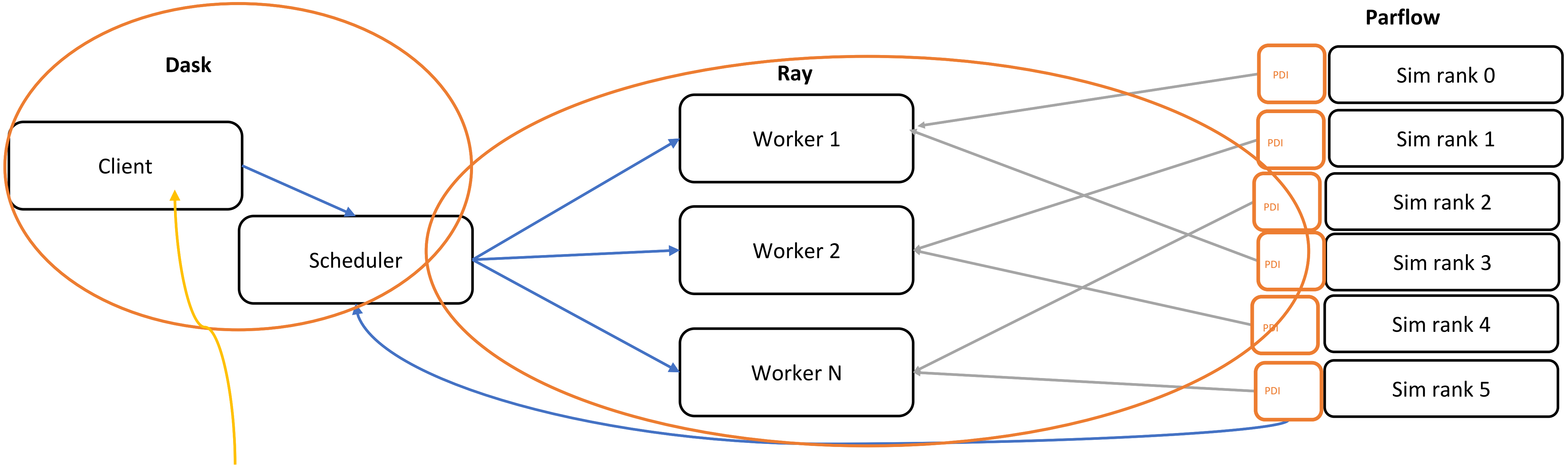
WP2: Highlights

Reconfigure analyses placement at runtime when a new one is triggered (mixed use of dedicated cores and nodes, fixed amount of computing resources):



WP2: Highlights

- Deisa On Ray : In-Situ Data Analytics (coupling between MPI simulations and Python analytics)



Use Dask-on-Ray:

- Why **Dask**: Dask Arrays -> a parallel version of Numpy
- Why **Ray**: performance, flexibility, and support of other tools
- In Situ specific –> Stream of data –> Rolling window

10

WP2: Highlights

- Deisa On Ray : Streaming window approach with dask API (distributed numpy arrays)

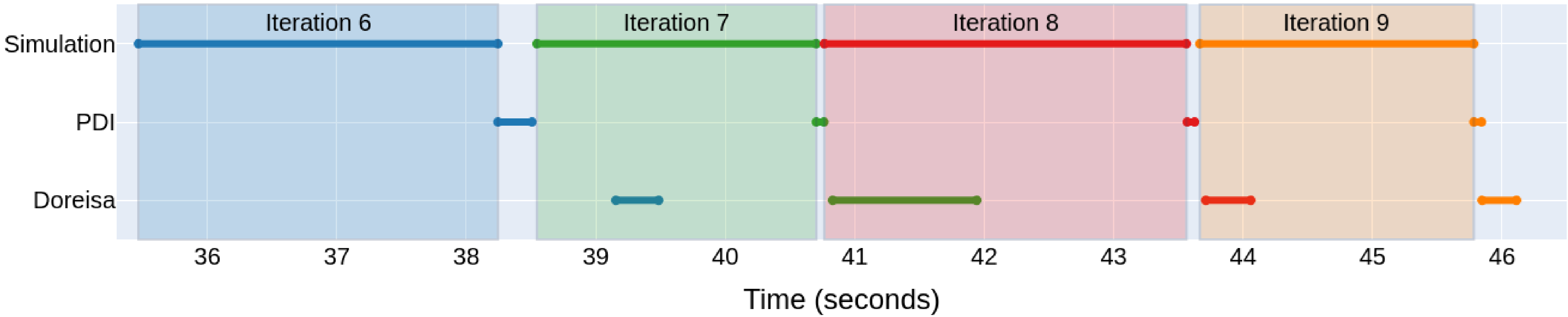
```python
def callback(temperature: list[da.Array], pressure: da.Array, *, timestep: int):
    if len(temperature) == 2:
        diff = temperature[1] - temperature[0]
        print("Mean temperature difference:", diff.mean().compute())

    print("Max pressure:", pressure.max().compute())

run_simulation(callback, [
    ArrayDefinition("temperature", window_size=2),
    ArrayDefinition("pressure", preprocessing_callback=lambda array: 10 * array),
])
```

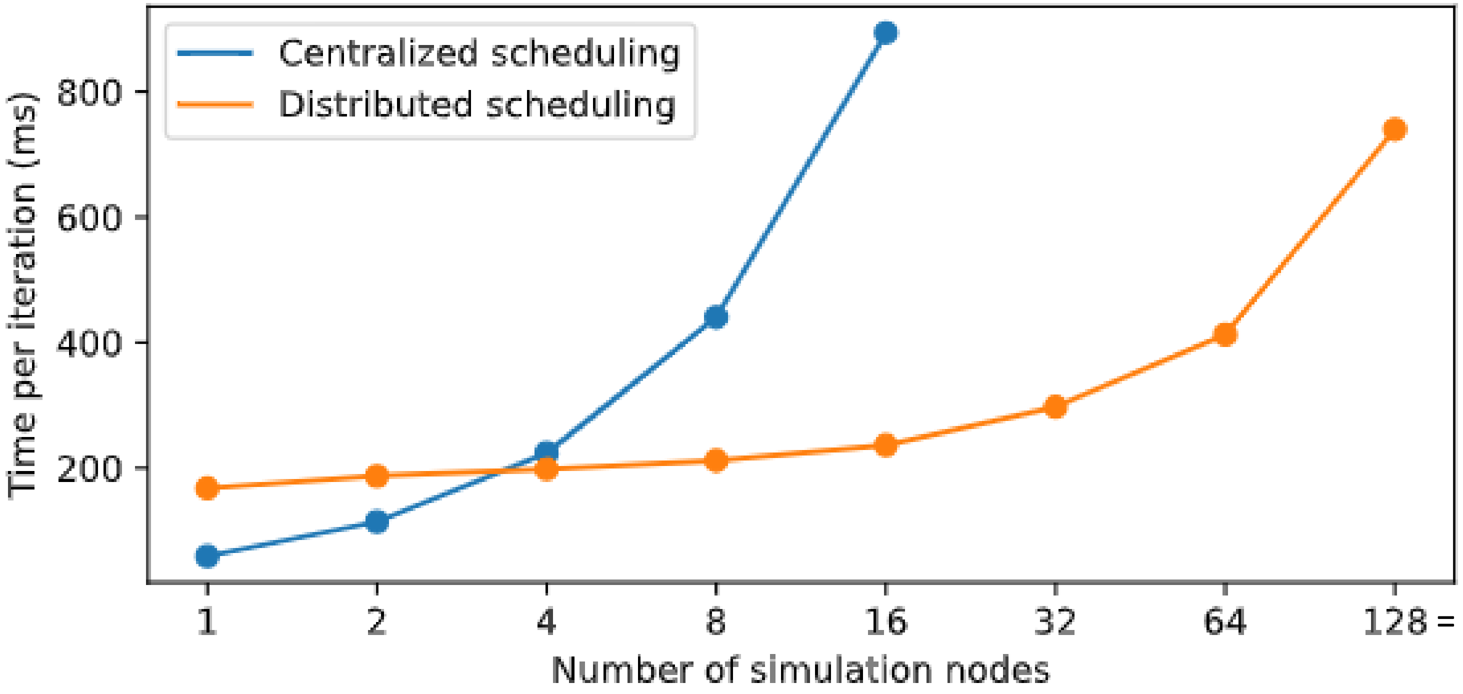
WP2: Highlights

- Deisa On Ray : A typical workflow of sim + pdi + analytics



WP2: Highlights

- Deisa On Ray : Ray Backend with decentralized distributed scheduling



= 5,120 procs.

WP2: Highlights

- Deisa On Ray:
  - Started to support: gysela + parflow (Eocoe 3)
  - Unify API of Deisa on Dask + Deisa on Ray with MdlS (https://github.com/deisa-project)
  - Mini App based on Gray-Scott Simulation (C++/Kokkos and Python)
  - Next steps:
    - Stabilize and improve API with feedback from applications.
    - Benchmark production runs + GPU.
    - Further optimizations: prepare DAG ahead of time, compiled graphs.

WP2: Next steps

- Continue to develop our libraries: issues, bugs, PRs, lots of coffee, review, release
- Work with demonstrators to answer theirs needs as much as possible.
    - Integrate PDI dans Coddex for using Deisa.
    - Test other in-situ scenarios (depending on the discussion with WPs)
    - Take Slack messages very seriously
- Unique API for PDI and Damaris
- PhD Thesis of Ivan: elastic scheduling
- Paper(s?) for PDI and Deisa : white paper, technical report, etc
- Regular training sessions (Damaris)
- Work with other WPs, PCs : AI, packaging, etc
- HR: one PhD open position