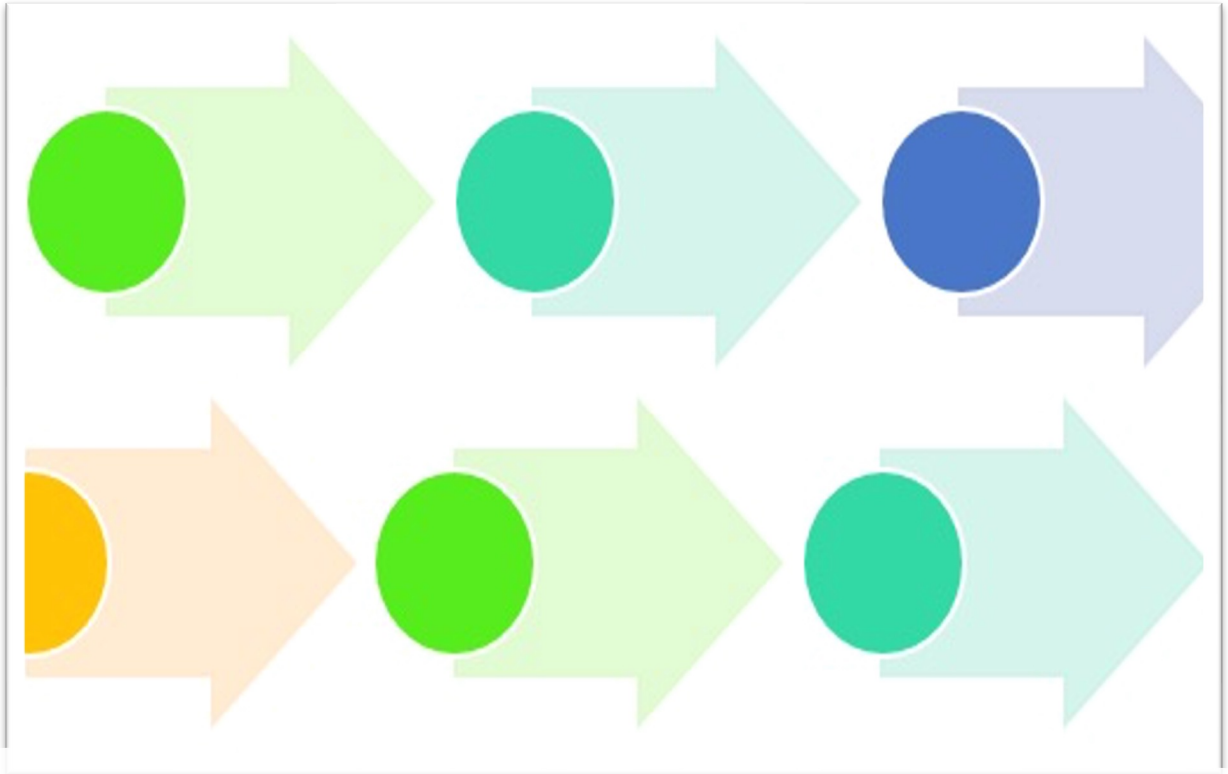
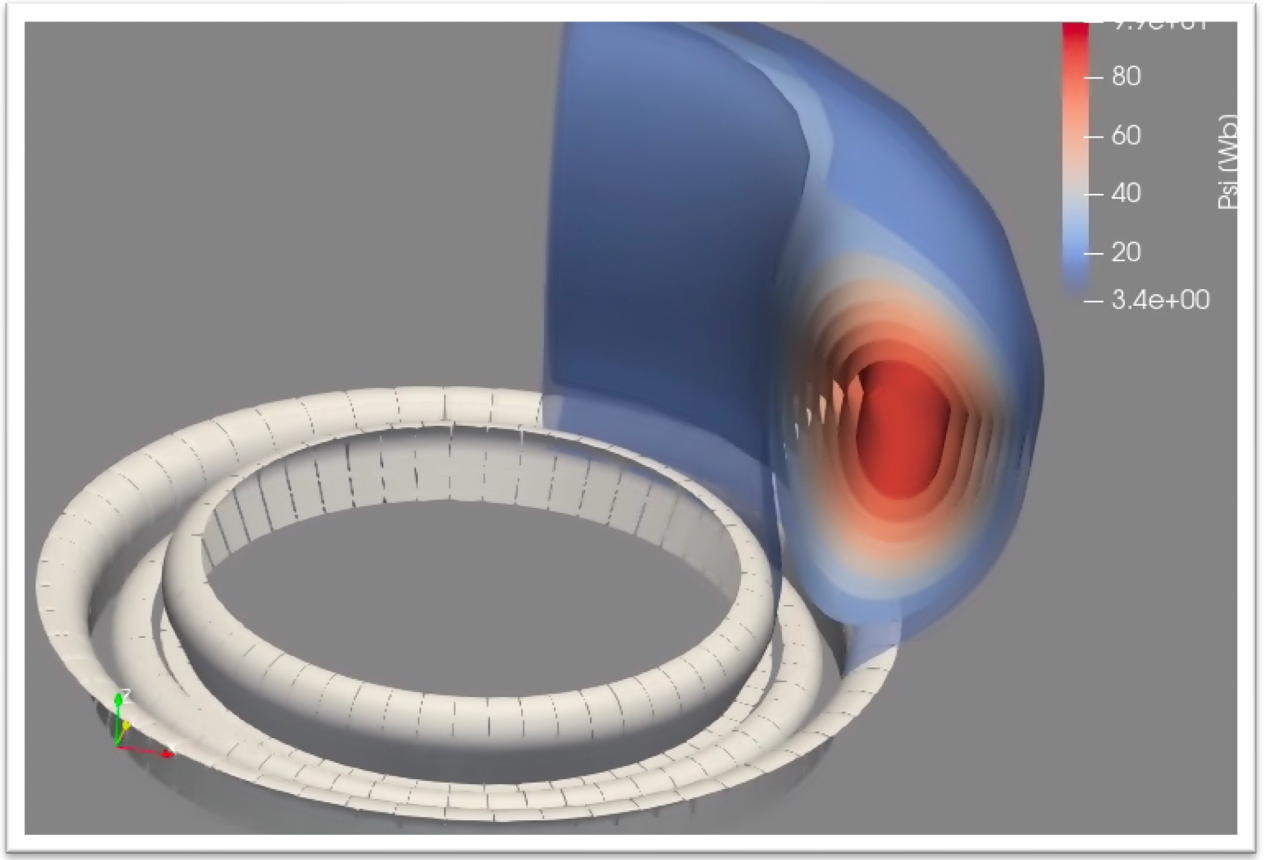




In-pulse analysis at ITER:

workflow managers and 3D visualization tools

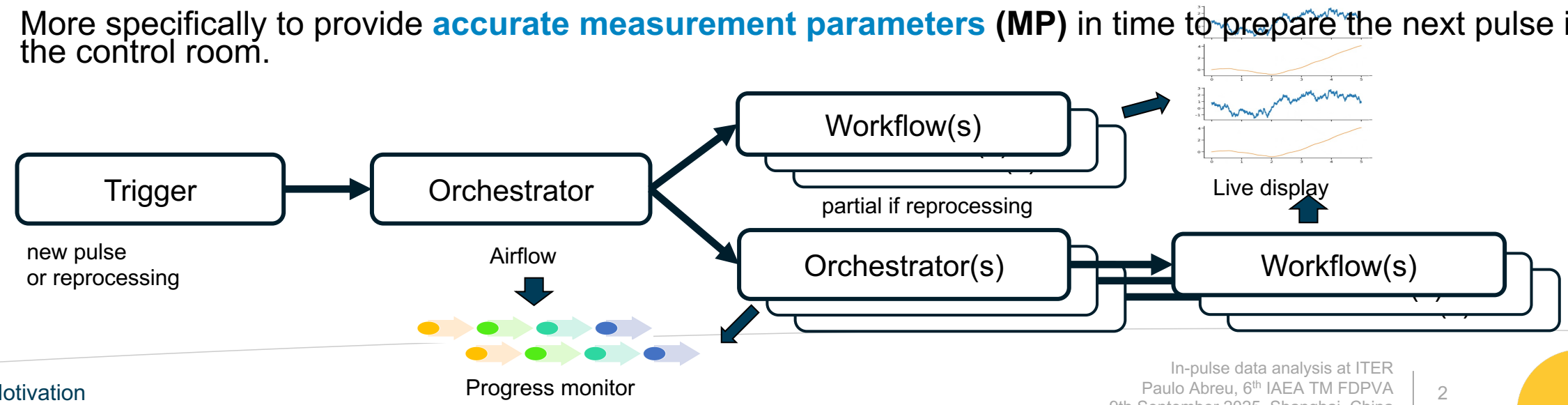
P Abreu (ITER Organization)
S. Rajesh (IO intership, Kalasalingam Academy of Research and Education, India)
S. Blockhuizen, M. Sebregts, D. V. Vougt (Ignition Computing, Netherlands)

iter
china eu india japan korea russia usa

Disclaimer: The views and opinions expressed herein do not necessarily reflect those of the ITER Organization

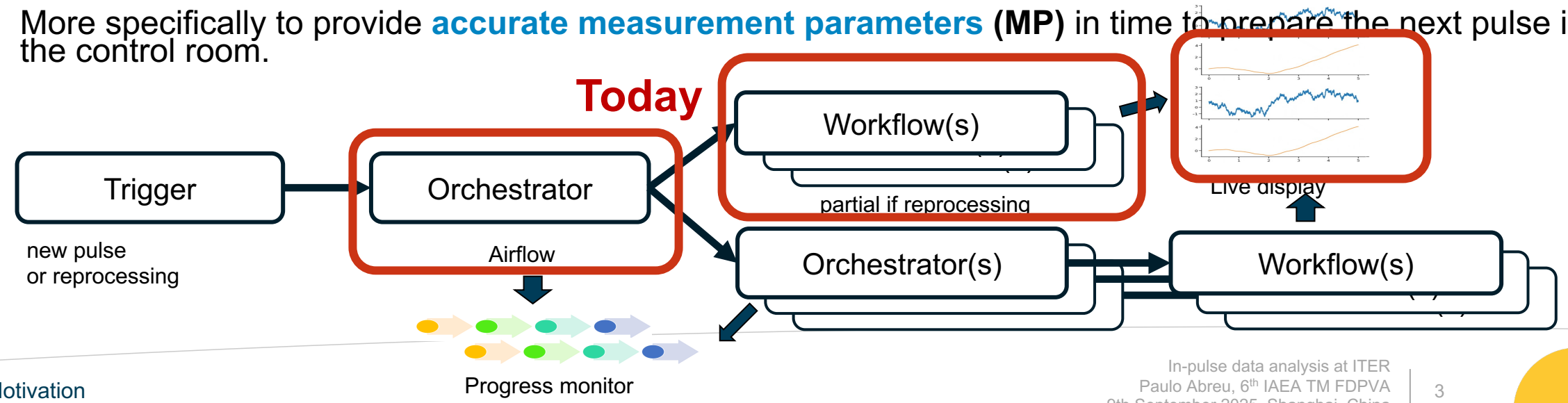
What is in-pulse data processing?

- A set of interdependent programs/actors that calculate **plasma parameters** from measurements.
- They use as **input** both the **raw data** from the diagnostics and **processed data** by other programs (hence the interdependence).
- The complete workflow(s) can be represented by a directed acyclic graph (**DAG**).
- The wider goal is to provide a consistent **interpretation of the plasma state** (with uncertainties) from the measurements **during** each pulse to guide the current experimental session.
- More specifically to provide **accurate measurement parameters** **(MP)** in time to prepare the next pulse in the control room.



What is in-pulse data processing?

- A set of interdependent programs/actors that calculate **plasma parameters** from measurements.
- They use as **input** both the **raw data** from the diagnostics and **processed data** by other programs (hence the interdependence).
- The complete workflow(s) can be represented by a directed acyclic graph (**DAG**).
- The wider goal is to provide a consistent **interpretation of the plasma state** (with uncertainties) from the measurements **during** each pulse to guide the current experimental session.
- More specifically to provide **accurate measurement parameters** **(MP)** in time to prepare the next pulse in the control room.



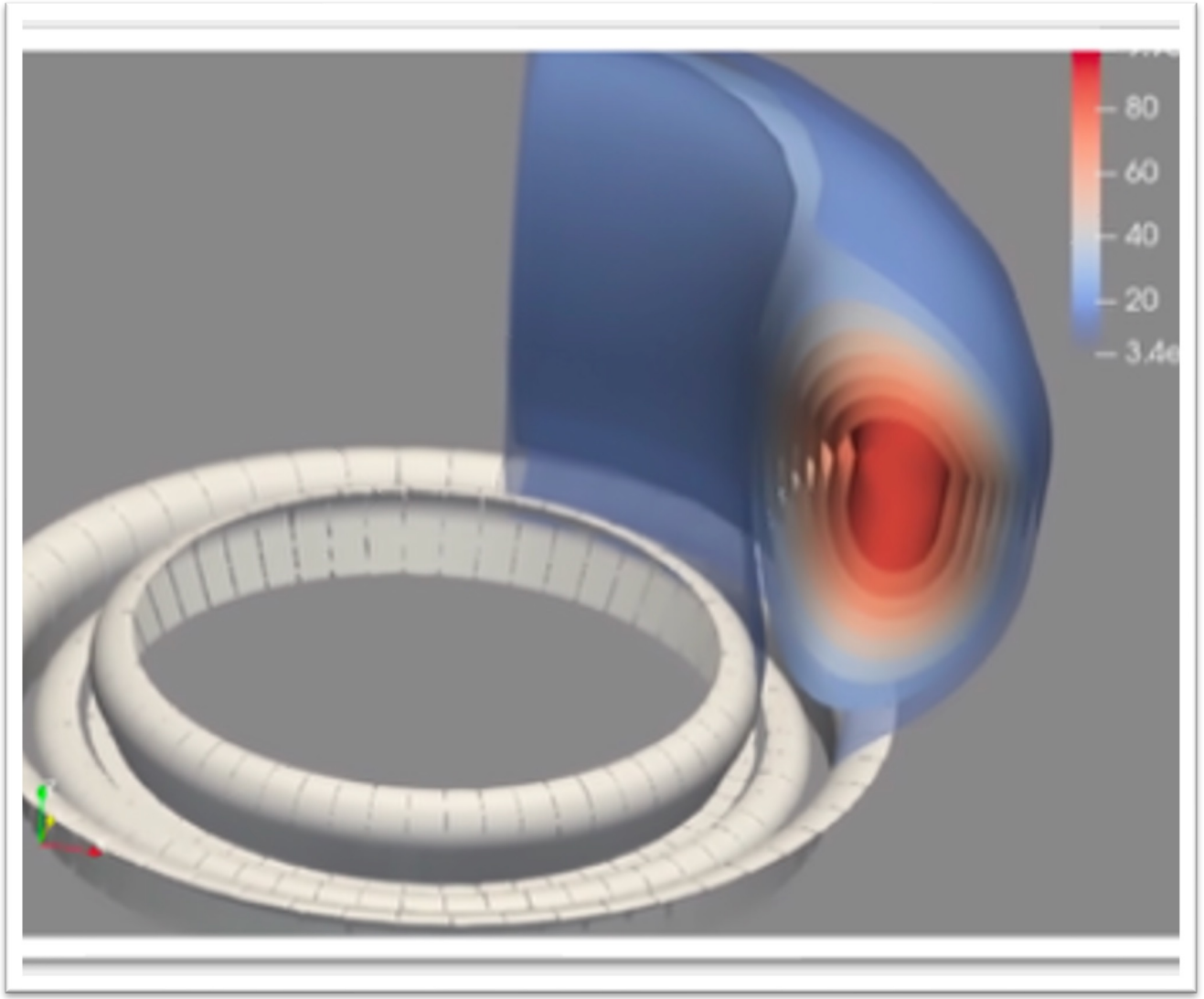
Overview

A. Motivation

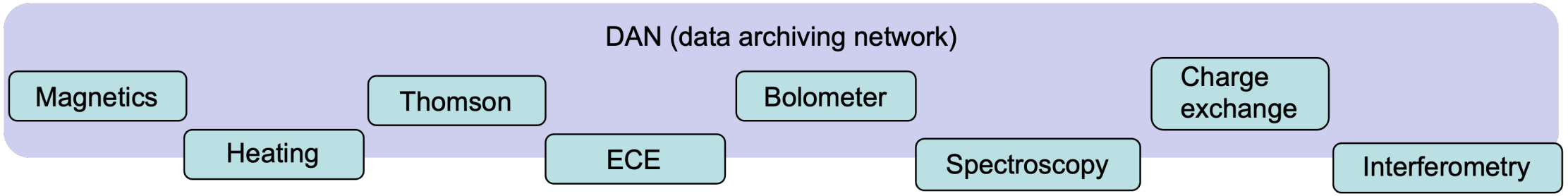
B. Workflow manager
   A. Airflow
   B. First application: equilibrium reconstruction
   C. Workflow framework

C. 3D visualization

D. Conclusions & future work



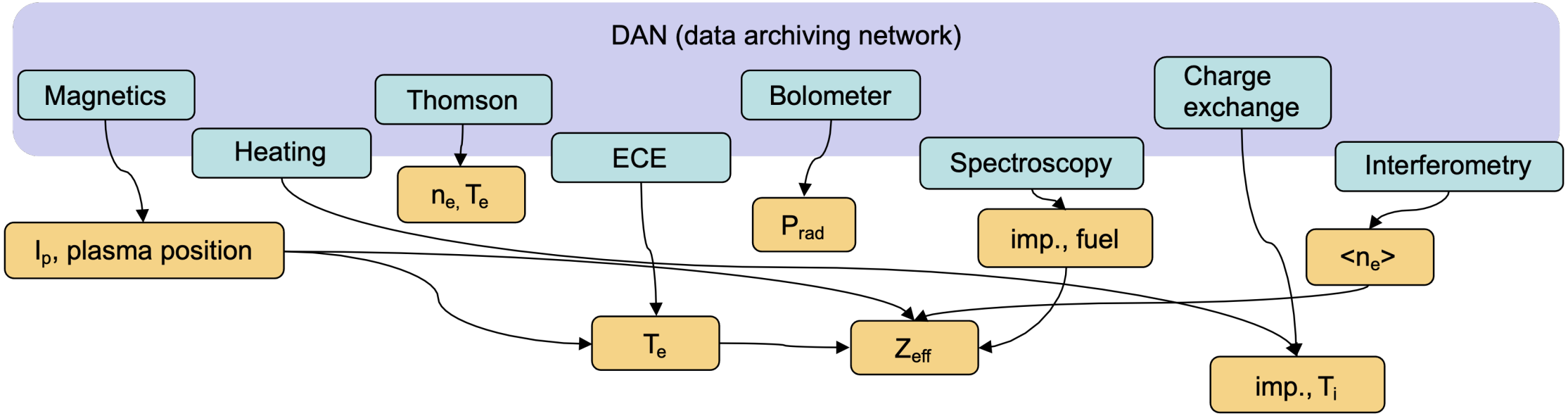
In-pulse analysis workflows: from diagnostics to physics



Labels: Diagnostics (~50)

Based on D. Dodt https://doi.org/10.1016/j.fusengdes.2012.11.015

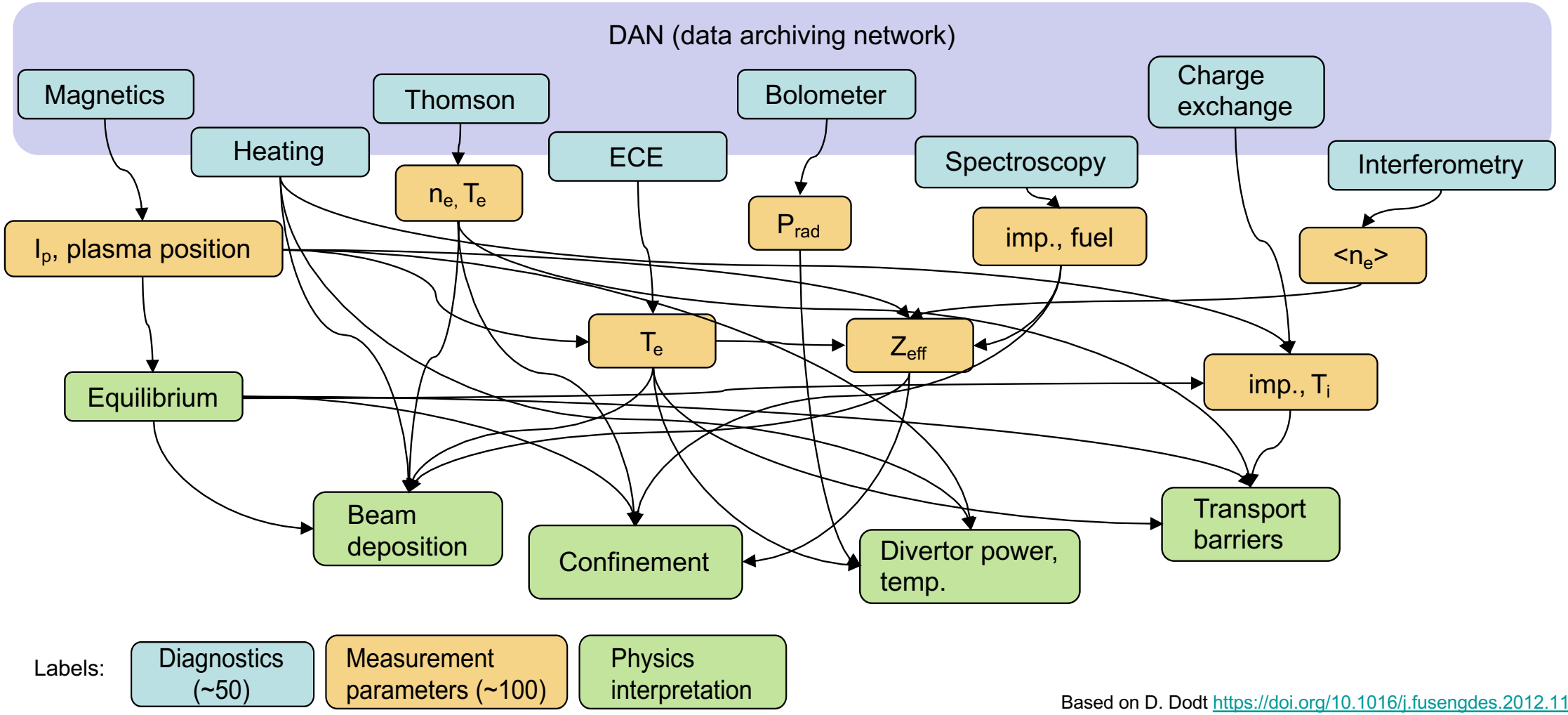
In-pulse analysis workflows: from diagnostics to physics



Labels: Diagnostics (~50) Measurement parameters (~100)

Based on D. Dodt https://doi.org/10.1016/j.fusengdes.2012.11.015

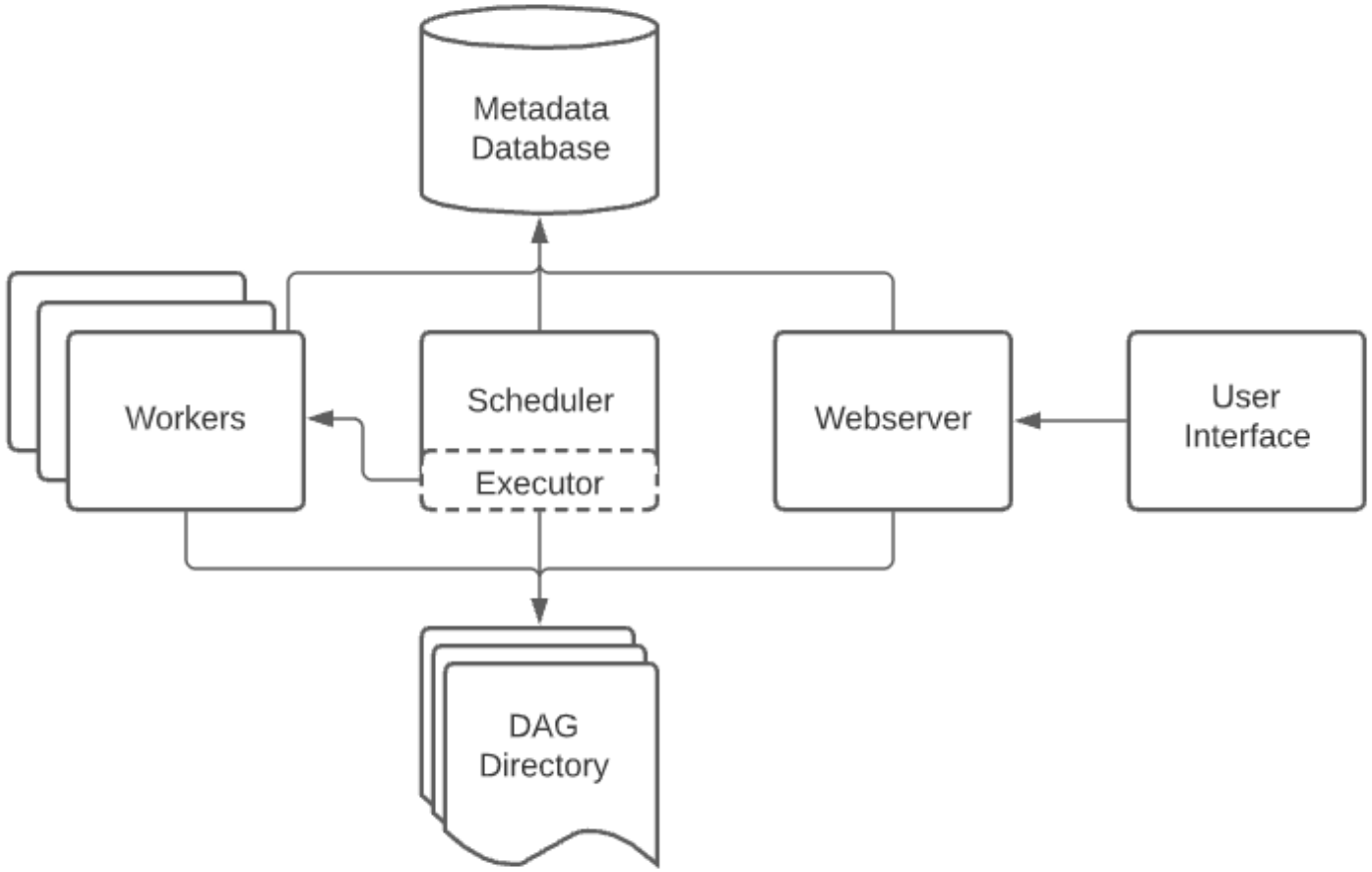
In-pulse analysis workflows: from diagnostics to physics



Based on D. Dodt https://doi.org/10.1016/j.fusengdes.2012.11.015

Apache Airflow as the workflow manager

- Open source
- Python
- DAG
- Web and CL interfaces
- Scalable (built-in task parallelism)
- Operators can have introspection into actors:
  - Fault detection
  - Restart
  - Replace



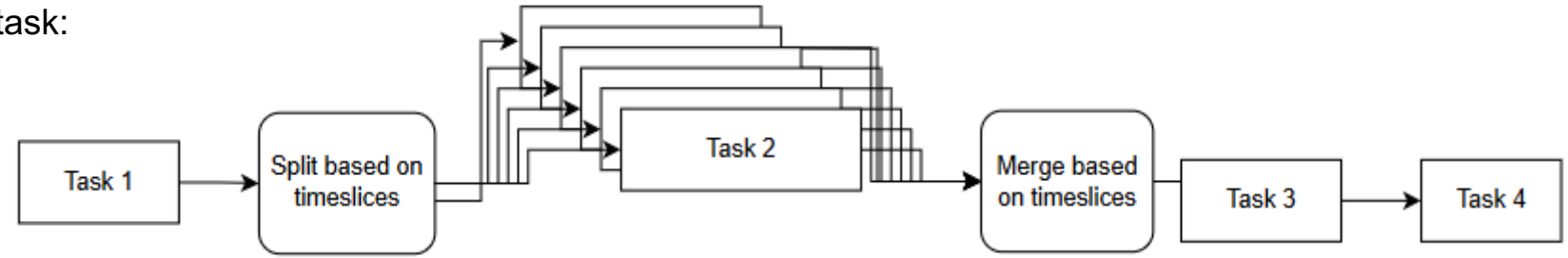
Airflow features at a glance

1. **Retry Mechanism:**
   - Automatically retries failed tasks to enhance reliability.
2. **Configuring Task Scheduling:**
   - Inbuild scheduling with Airflow.
   - External scheduling.
3. **Parallel Execution:**
   - **Parallelism:** Maximum number of tasks running across all DAGs.
   - **Concurrency:** Maximum tasks allowed for a single DAG.
4. **Task Dependencies:**
   - Specify execution order explicitly.

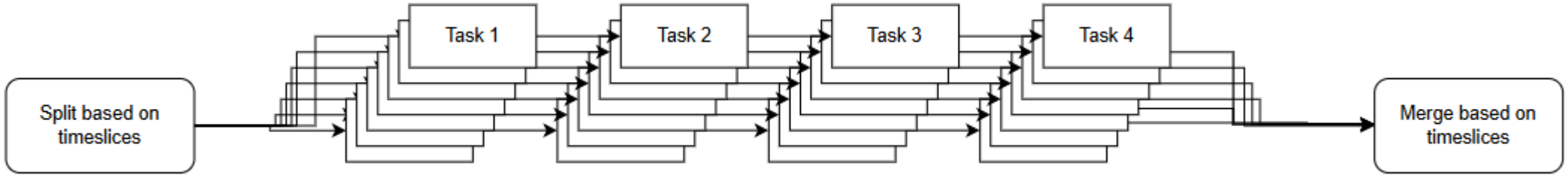
```python
# Define default_args for the DAG
default_args = {
    'owner': 'airflow',                        # Task owner
    'depends_on_past': False,                  # Don't wait for previous runs
    'email_on_failure': False,                 # Disable email alerts on failure
    'email_on_retry': False,                   # Disable email alerts on retry
    'retries': 3,                              # Retry up to 3 times
    'retry_delay': timedelta(minutes=5),       # Wait 5 minutes between retries
}

# Initialize the DAG
with DAG(
    dag_id = 'print_time_dag',                   # Unique DAG ID
    default_args = default_args,                 # Apply default arguments
    description = 'A DAG with key configurations to print the current time',
    schedule_interval='@hourly',               # Runs every hour
    max_active_runs = 10,                        # Limit concurrent runs of this DAG
) as dag:

    # Define the task functions
    def print_time():
        print(f"Current Time: {datetime.now()}")

    def end_message():
        print("Workflow Completed Successfully!")

    # Task 1: Print the current time
    print_time_task = PythonOperator(
        task_id='print_time',                  # Task name
        python_callable=print_time,            # Callable function
    )
    # Task 2: Print an end message
    end_message_task = PythonOperator(
        task_id='end_message',                 # Task name
        python_callable=end_message,           # Callable function
    )
    # Define task sequence flow
    print_time_task >> end_message_task      # Task 1 must finish before Task 2
```

Simple parallelization per time slice

- Most in-pulse actors process single time slices.
- We can process time slcies in batches / ranges with simple parallelism
  - Per actor / task:



  - Or per workflow:

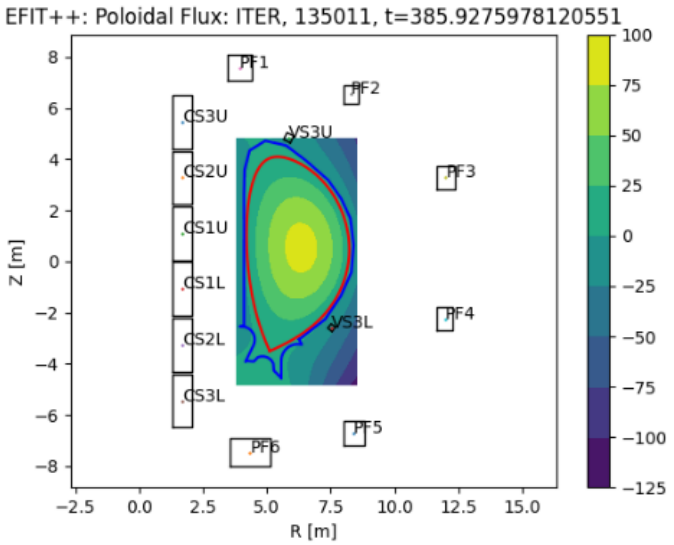


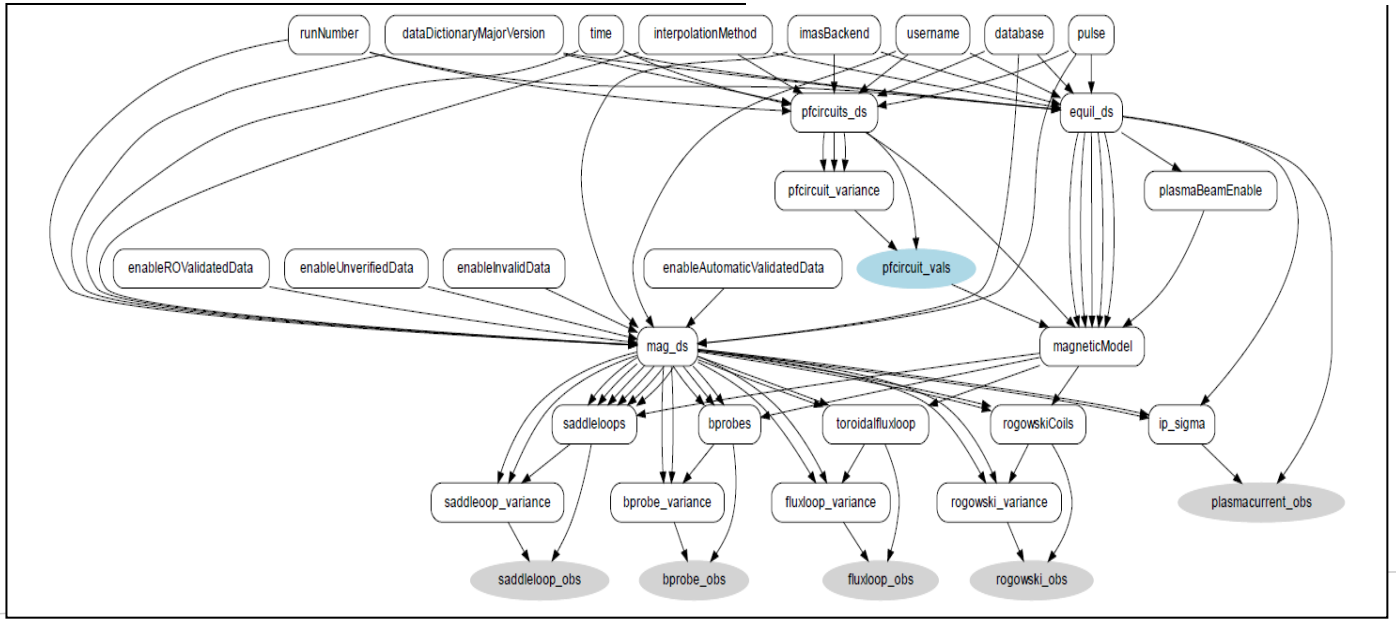
First application: equilibrium reconstruction with synthetic magnetics constraints

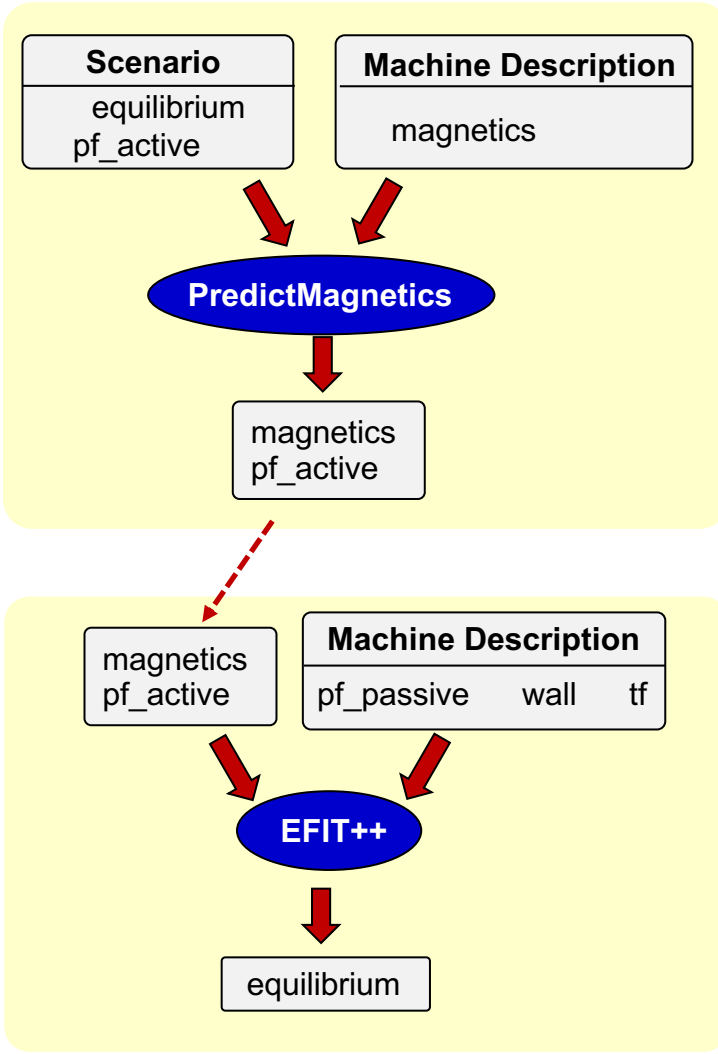
- **Magnetics Actor**
  - **Magnetics_prep** – Reads machine/diagnostic specific data.
  - **Magnetics** – Synthetic diagnostic modelling for magnetics measurements.
- **Equilibrium Actor**
  - **Equilibrium_prep** – Reads machine/diagnostic specific data for the equilibrium reconstruction.
  - **Equilibrium** –Actor used to reconstruct plasma equilibrium.
- **Machine agnostic**: diagnostic/machine specificities are parameters to the actors.
- **Code agnostic**: actors exchange data in IDS format, easily replaceable by other actors with the same inputs and outputs.
- In this particular exercise:
  - predictMagnetics (from the Minerva framework, L. Appel, UKAEA)
  - EFIT++ (L. Appel, UKAEA)

First application: equilibrium reconstruction with synthetic magnetics constraints

- 55.A* magnetic diagnostics
- PredictMagnetics by L. Appel within the Minerva framework
- EFIT++ actor





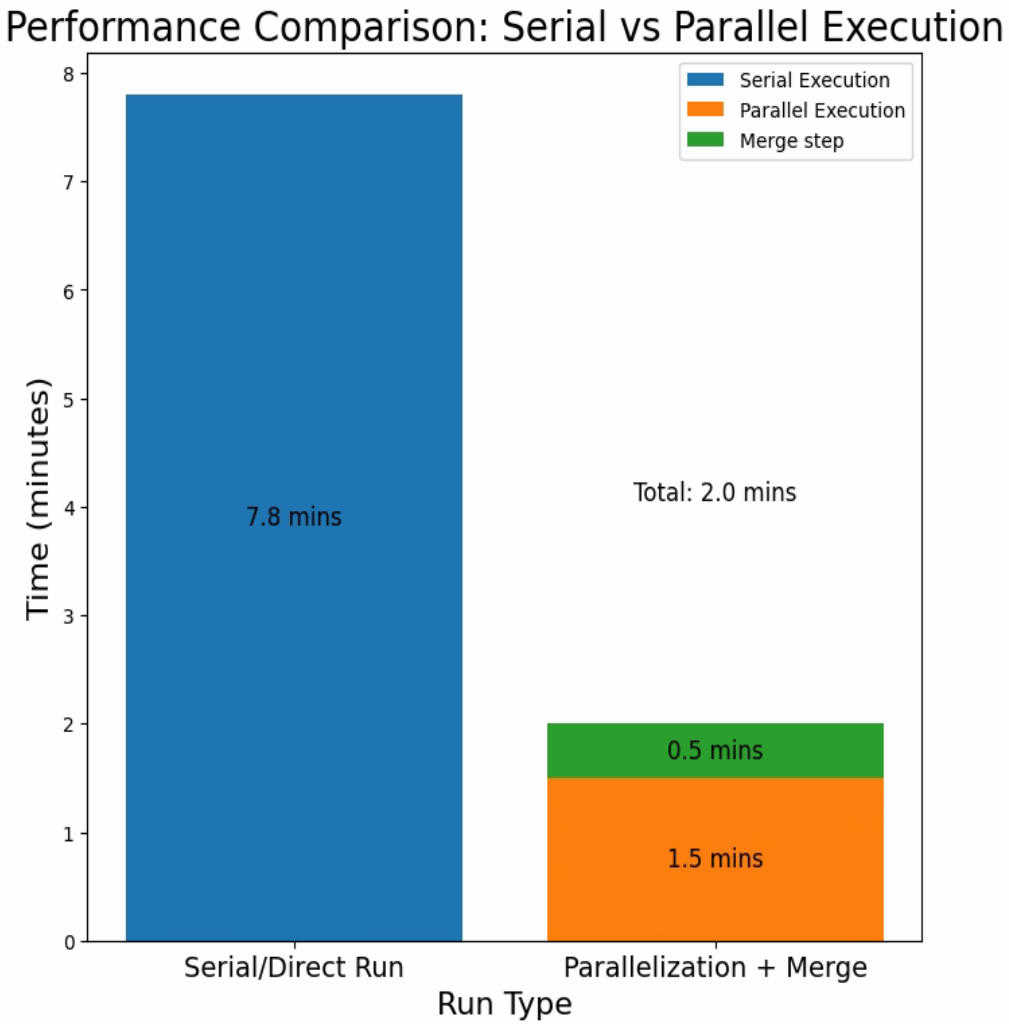


M. Schneider @ ITPA TG Diag 2022

Workflow manager: equilibrium reconstruction

First application: simple parallelism

- **Workflow info:** the **Magnetics Actor** for IMAS **Pulse 105027**, consisting of **614 timeslices.**
- **Serial Execution:**
  - Total execution time: **7.8 minutes**.
  - Tasks were executed sequentially without concurrency.
- **Parallel Execution + Merge:**
  - Average execution time of **5 parallel runs** : 1.5 minutes.
  - Merge operation of the runs : 0.5 minutes.
  - Total execution time reduced to **2.0 minutes** (1.5 + 0.5 minutes).
- **Conclusion:**
  - Even with the **overhead of the merge step,** the average execution time of the parallel runs combined is significantly faster than the serial/direct run.

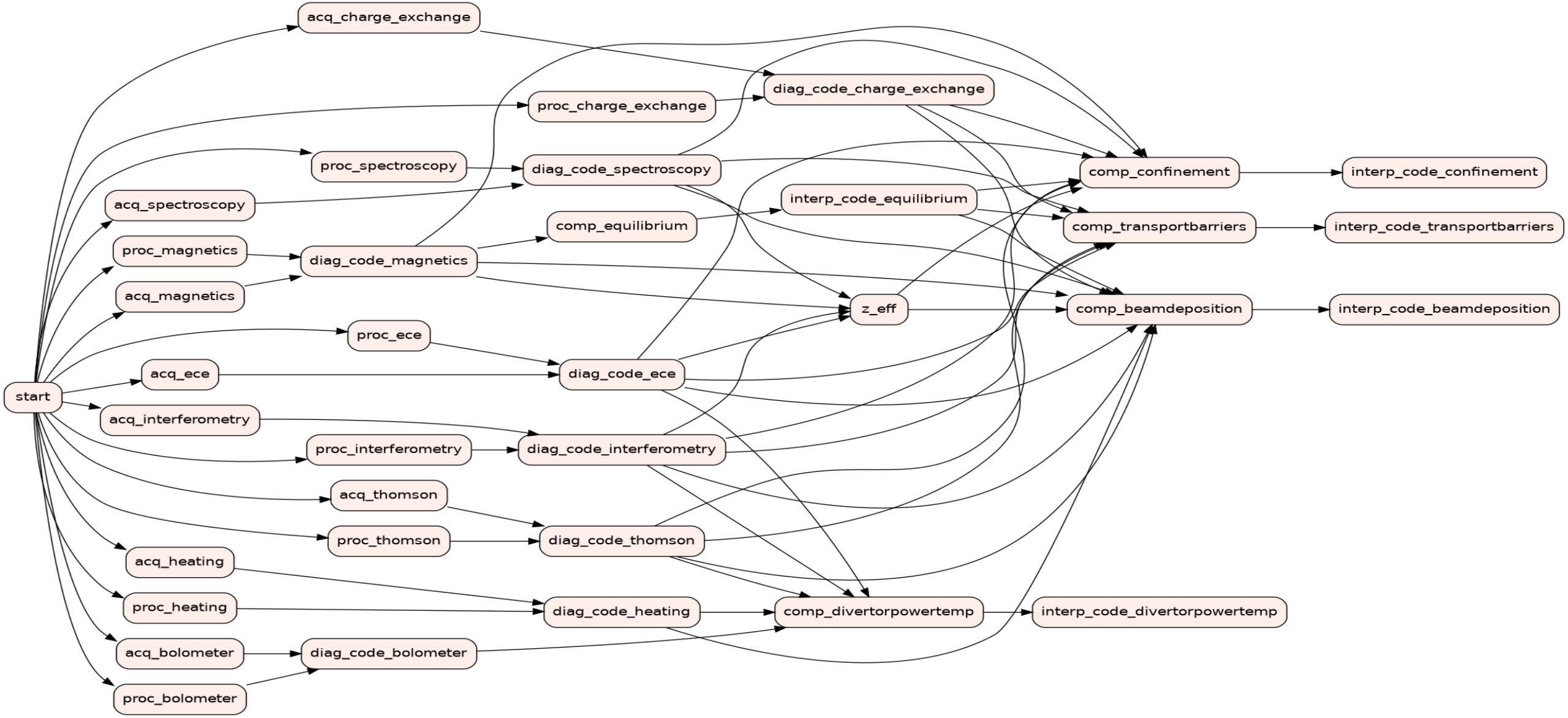


In-pulse Workflow Framework

Developed a **mock in-pulse processing workflow** to simulate a complete plasma analysis workflow during a pulse.

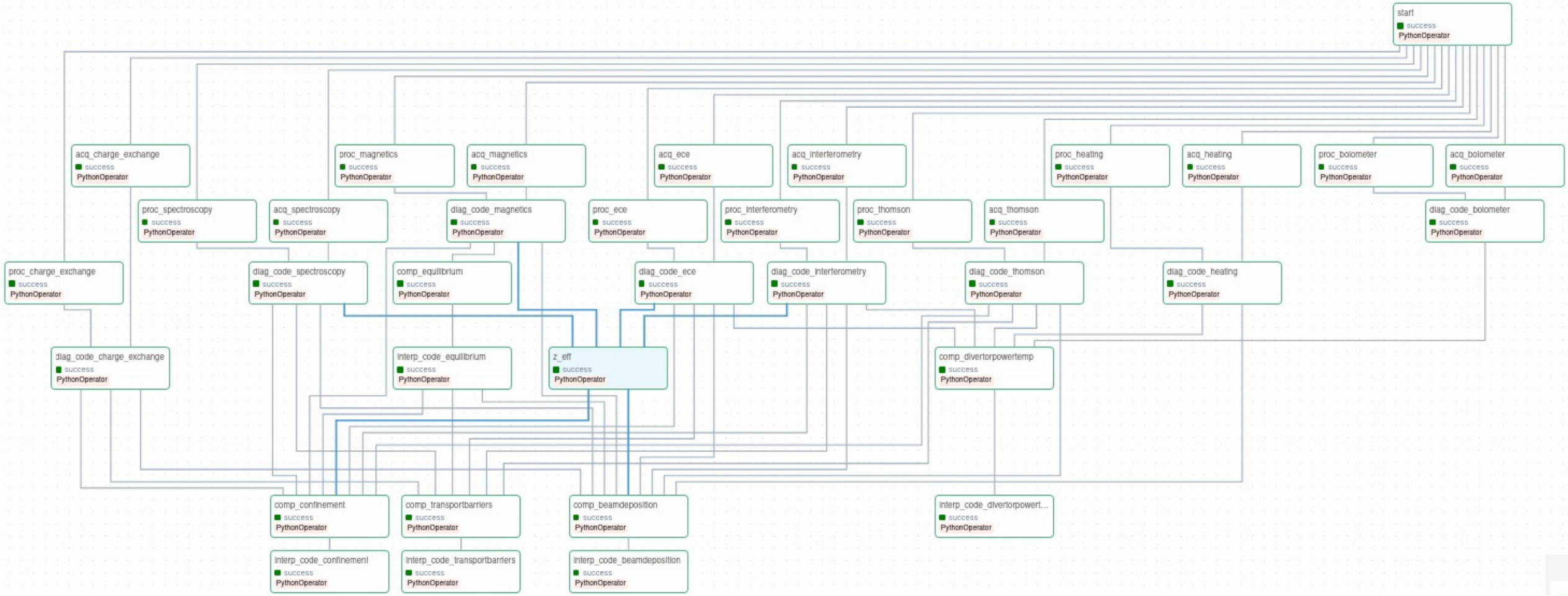
- The workflow serves as a **blueprint** for future integration of real physics-based codes in ITER's real-time processing pipeline.
- All computational steps currently use **placeholder actors**, designed to be replaced with validated physics modules in future iterations.
- This **framework** establishes a baseline architecture for modular, scalable in-pulse workflows spanning multiple diagnostics and control loops.

In-pulse Workflow Simulation



Workflow manager: framework

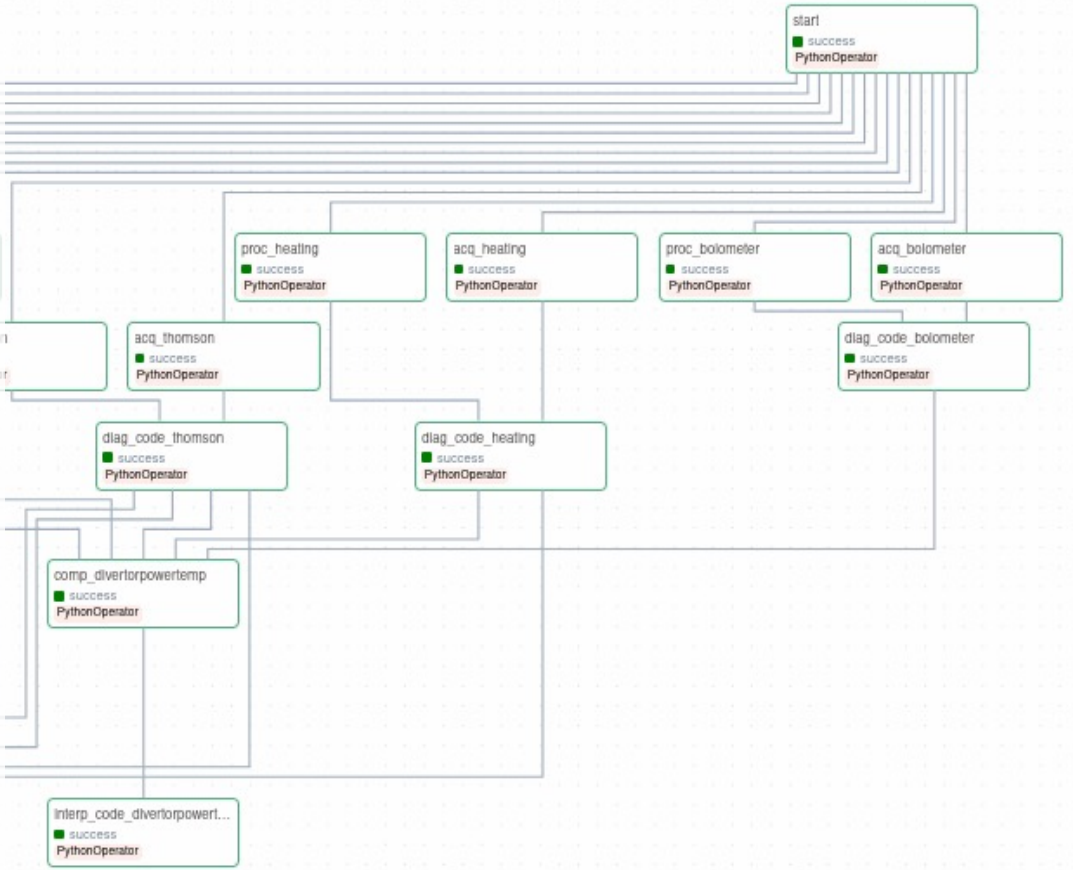
In-pulse Workflow Simulation in Airflow



Workflow manager: framework

In-pulse Workflow Simulation in Airflow

- **Failure detection**: detects failures and supports resolution through **retry logic**, **fallback paths**, and execution monitoring.
- **Failure resolution**: includes alternative placeholder codes that can be triggered dynamically if a primary actor fails enabling **fault tolerance**.
- The architecture supports **horizontal scaling**, where multiple timeslices can be processed concurrently as they arrive.
- It also enables **vertical scaling**, allowing additional dependent tasks (e.g., diagnostics, post-processing) to be added without disrupting existing workflows.
- The orchestration setup is **modular** and **extensible**, allowing increased complexity and integration over time without changing the core design.



Workflow Simulation: in-pulse and reprocessing

1. **In-Pulse Workflow** (Simulated):

   Mimics online timeslice-by-timeslice processing during an active pulse. Tasks triggered sequentially as new timeslices arrive.

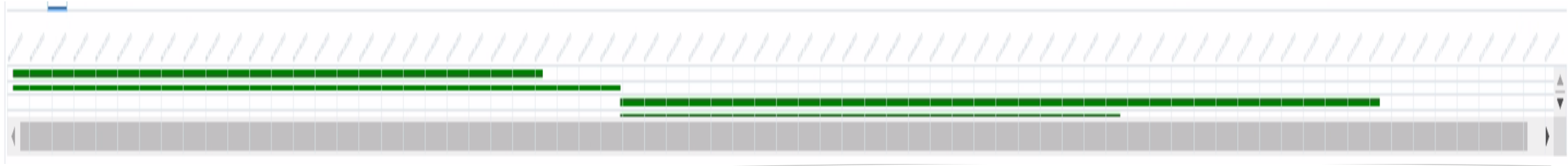
2. **Reprocessing Workflow** (Simulated):

- Executes after pulse completion using full data availability. Represents a traditional, serial processing pipeline.
- Both workflows use identical placeholder actors and input conditions, isolating orchestration behaviour for easier comparison.
- Each actor had randomized processing times, generated deterministically using the component's name as a seed.

Workflow Simulation: in-pulse and reprocessing

In-pulse:

Reprocessing:



Data visualization and exploration

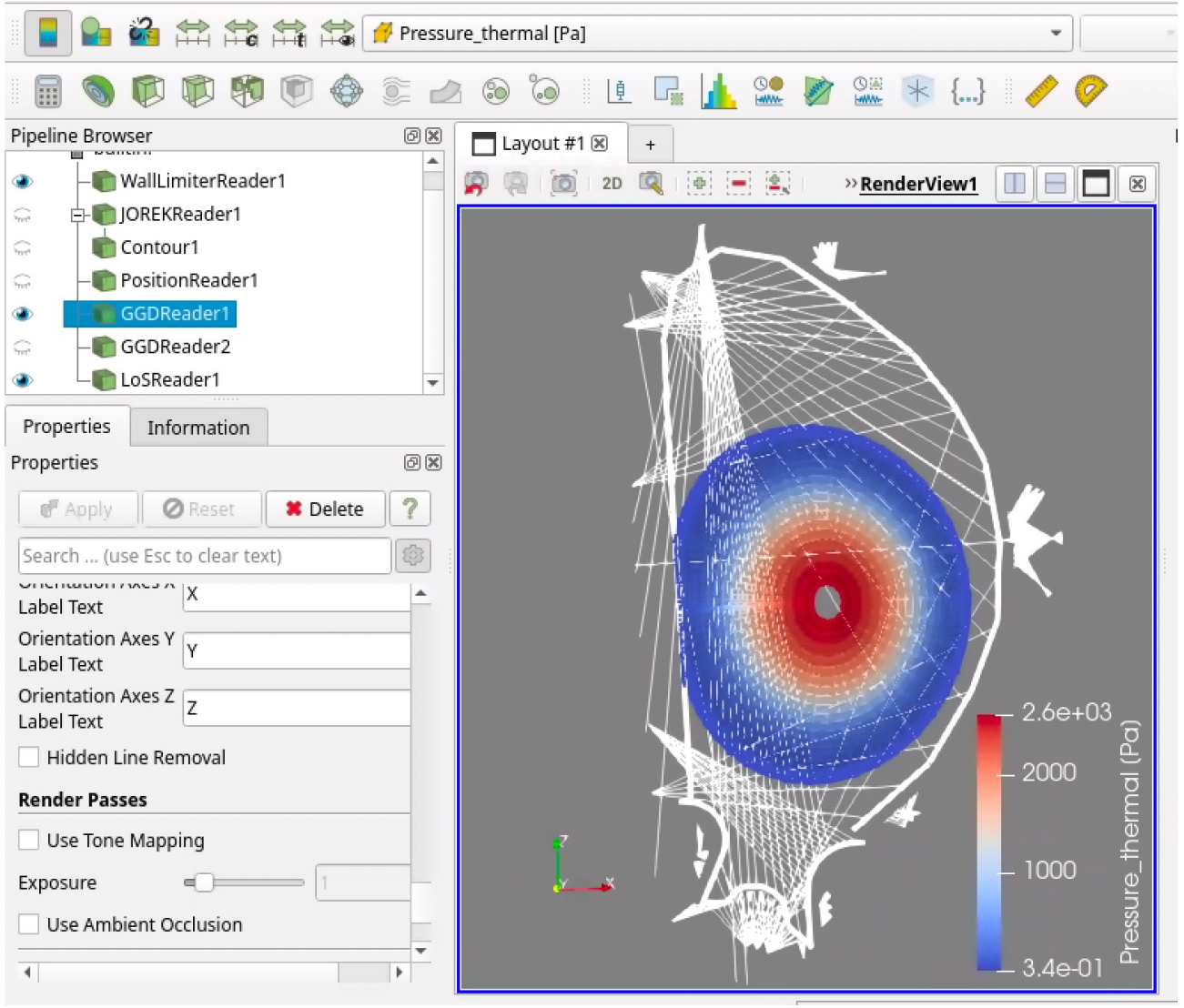
- Data is useless without insight.
- There is an ongoing development of ITER plotting library (L. Abadie):
  - iplot ecosystem and MINT
  - DAN, SDN, IMAS, …
  - 1D and 2D
  - control room display and user desktop
- 3D scientific data exploration tool was missing:

  Kitware Paraview

  +

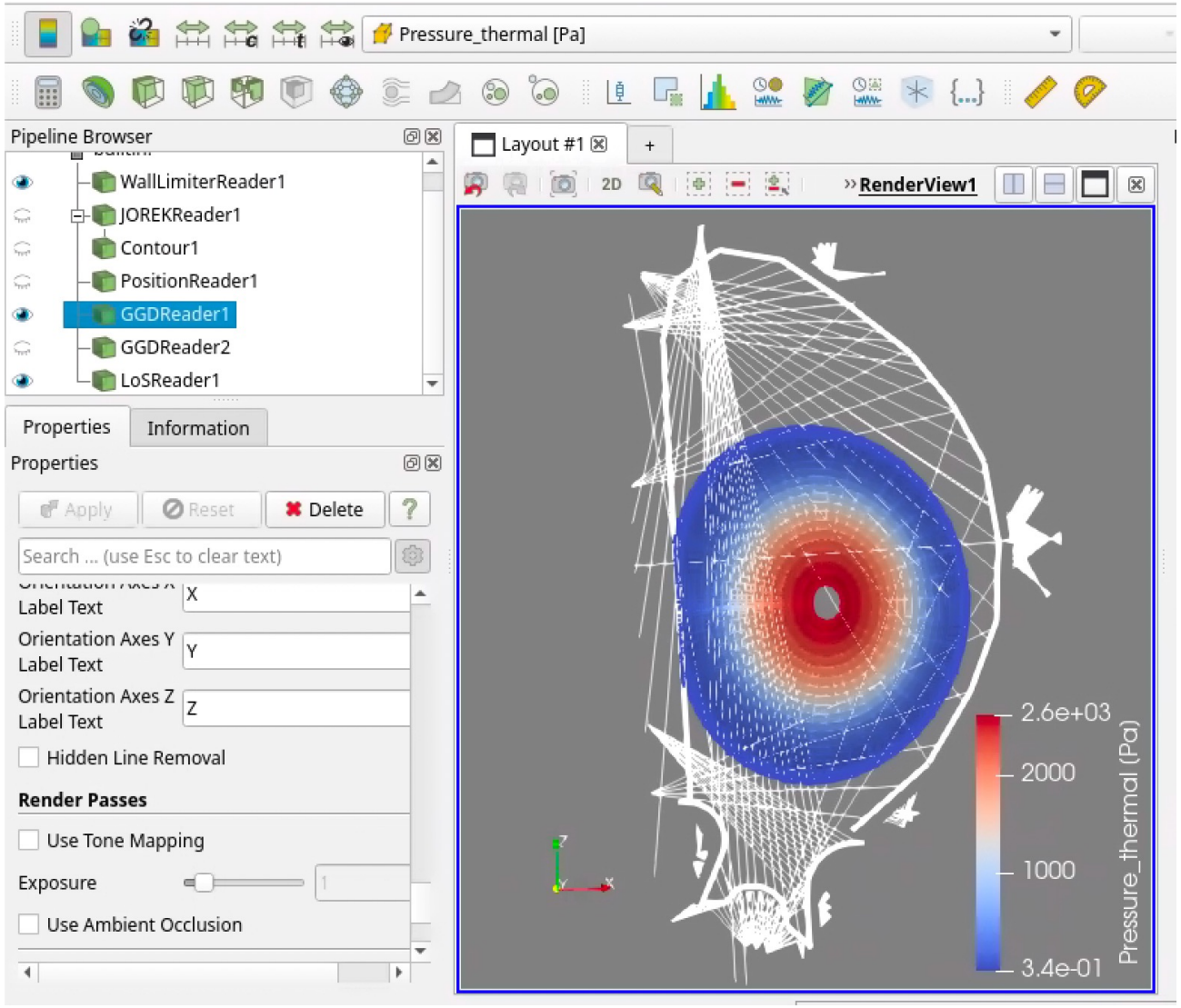
  IMAS plugin

SOLPS + Limiter + ECE LOS



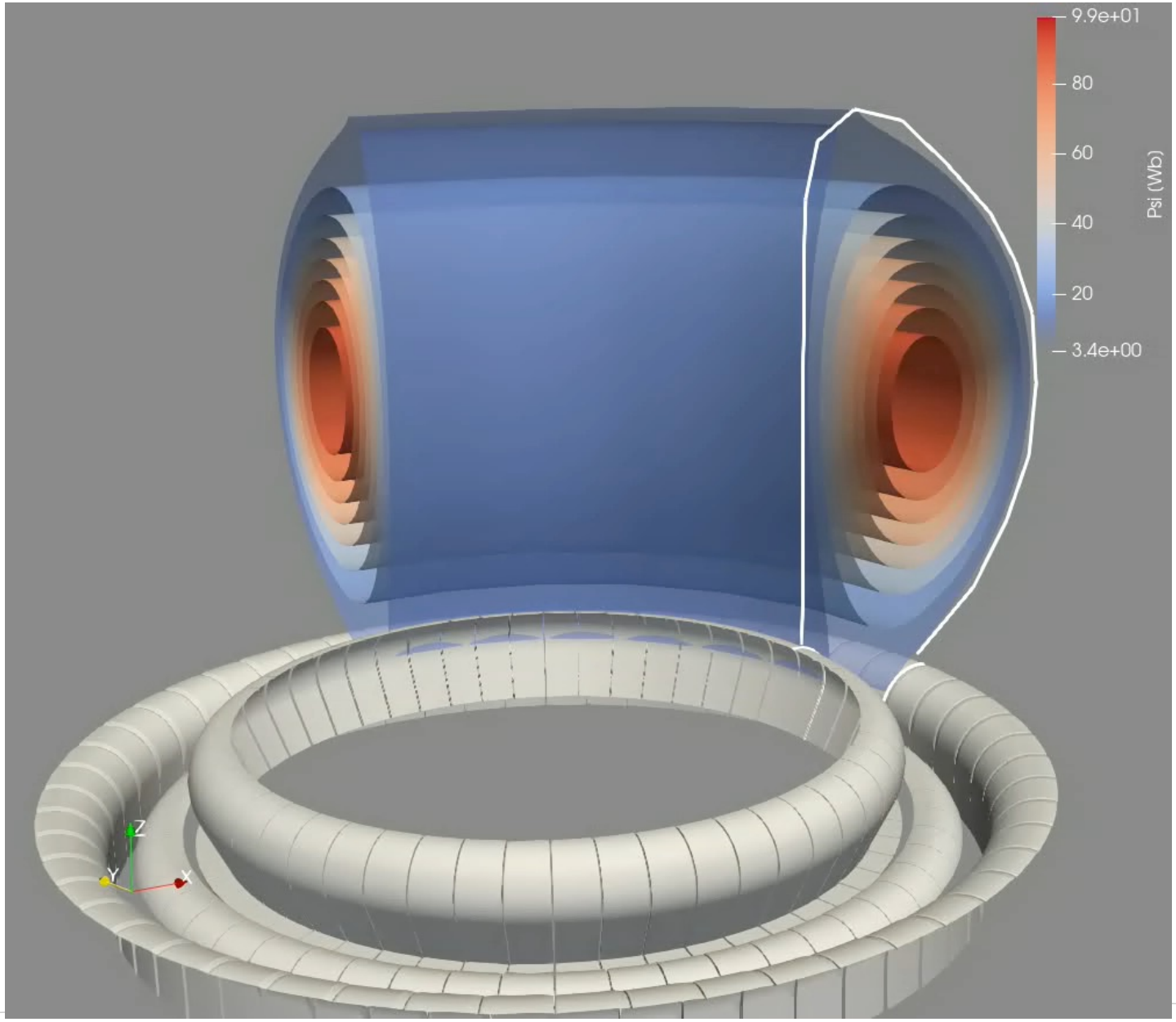
Data visualization and exploration

- See IMAS-Paraview manual and example.
- Open source on github
- GUI and CLI (export to VTK)
- GGD data (2D and 3D)
- JOREK
- 1D profile data:
  - Integrate 1D core with 2D edge
- Machine/diagnostic description data:
  - Divertor
  - Wall
  - LoS
  - Magnetic sensors
  - …



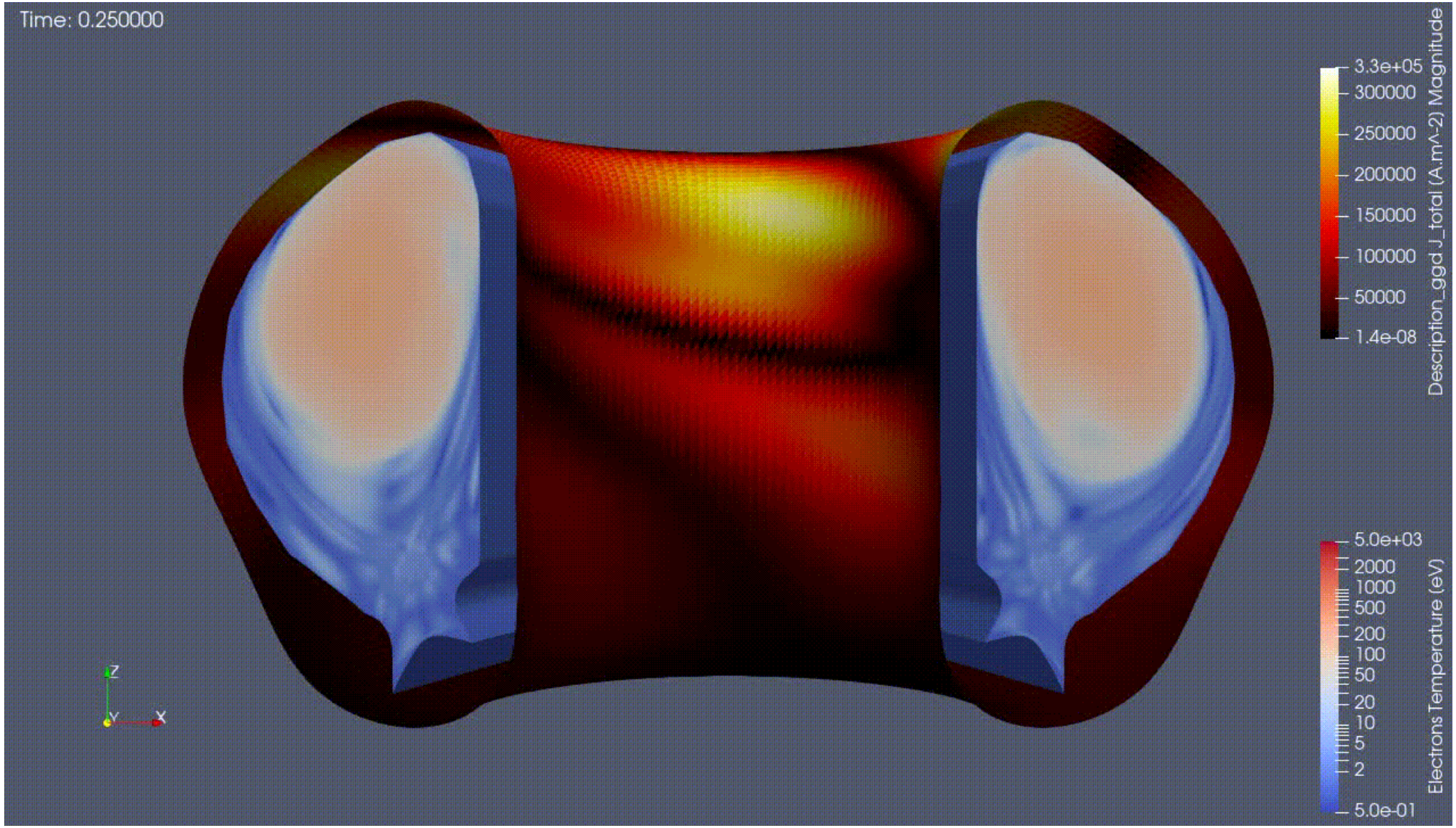
Example:

- JOREK data (psi)
- Divertor
- Limiter



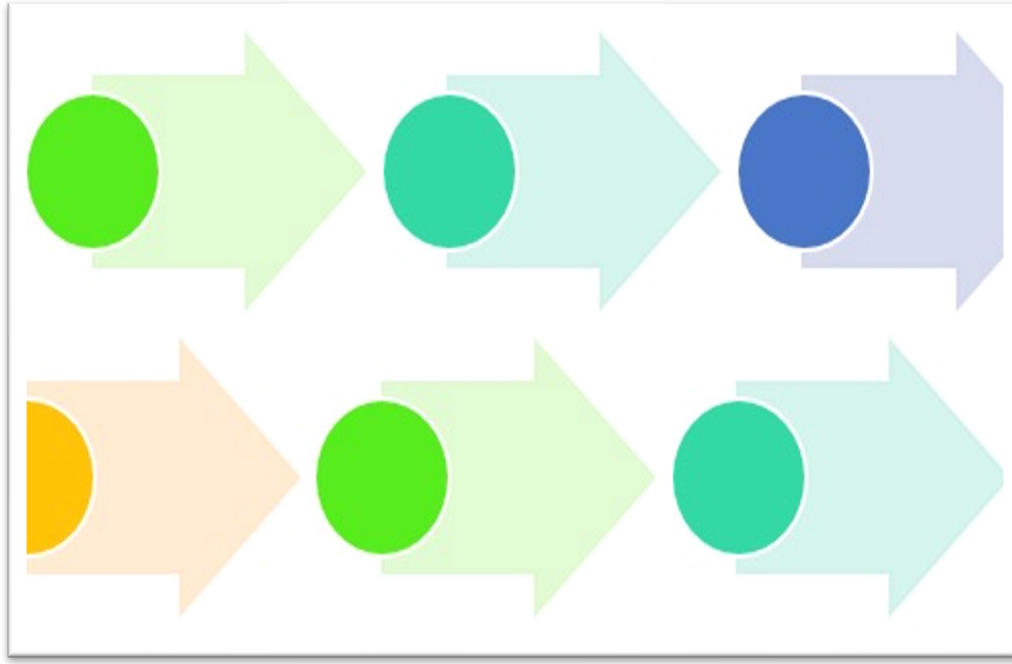
Example:

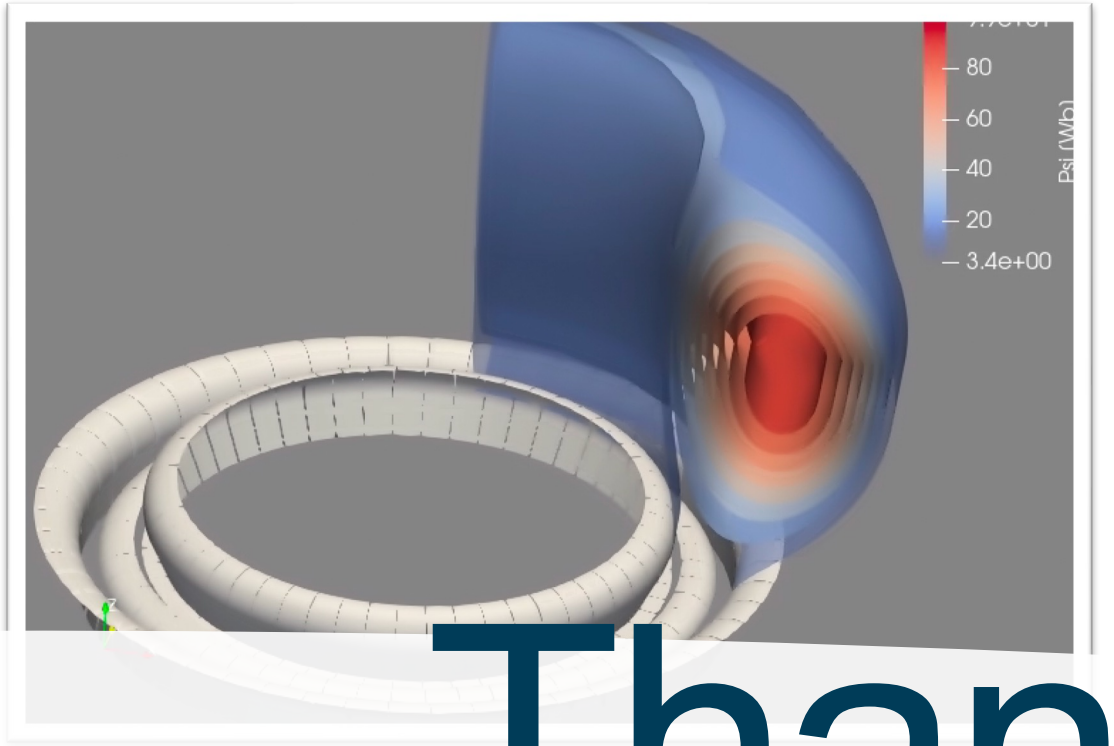
- JOREK data
- Animation of the electron temperature and wall currents.
- Data provided by J. Artola.



Future work

- Workflows:
  - Realistic **input data**:
    - Use **synthetic diagnostic models** as input to the workflows (see presentation from S. Pinches).
    - Use **mapped experimental data** from other devices.
  - Integrate with **MUSCLE3** (persistent actor framework):
    - Work by F. Poli based on the ETS-PAF
    - H&CD workflow by M. Schneider
- 3D visualization: further improvements based on user request





Thank you!

In-pulse analysis at ITER:
workflow managers and 3D visualization tools

Paulo Abreu, ITER Organization
Sixth IAEA Technical Meeting on Fusion Data Processing, Validation and Analysis
September 2025, Shanghai, China

iter
china eu india japan korea russia usa