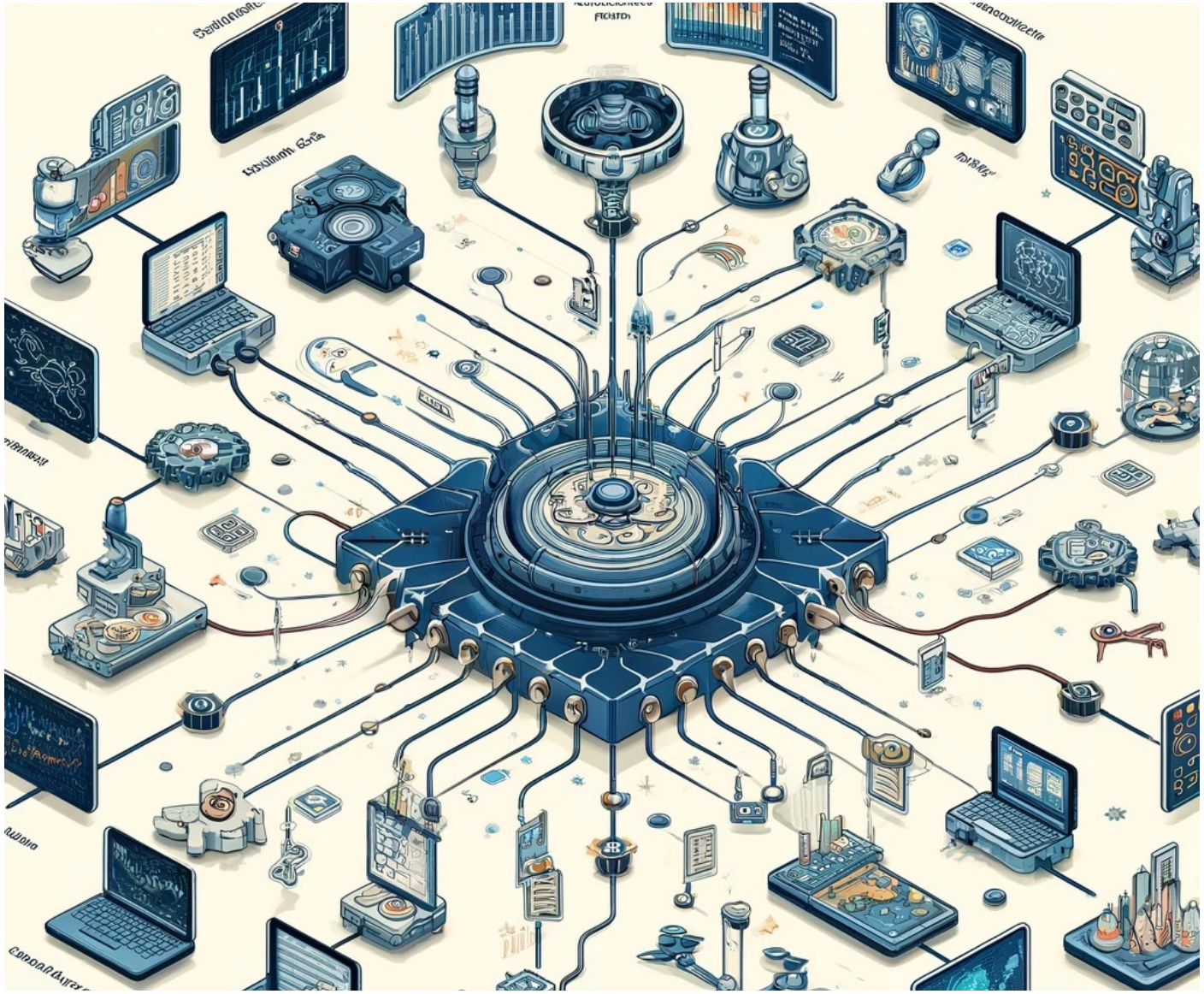
Exploring Scientific Workflows with CWL and dispel4py

Module 1.b

- Dr. Rosa Filgueira
- Lecturer at the School of Computer Science
- University of St Andrews
- rf208@st-andrews.ac.uk
- rosa.filgueira.vicente@gmail.com



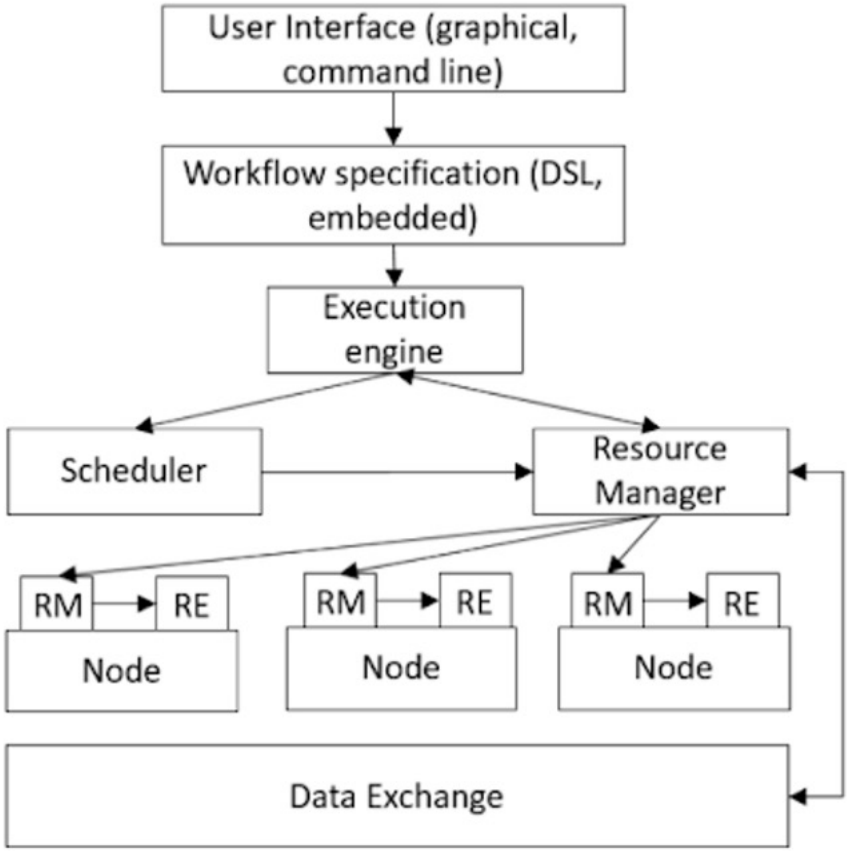
Module 1.b - Introduction to Scientific Workflows (II)

1. Recap
2. Beyond Basic Concepts
    i. Understanding DAG in Workflows
    ii. Levels of Composition
    iii. Levels of Workflow Abstraction
    iv. Abstract vs Concrete Workflows
    v. Computing Focus vs Semantic Focus
    vi. Levels of compositions
    vii. Sharing Workflows
    viii. Open Science Workflows
    ix. Current Challenges
3. Research Directions
    1. Serverless Frameworks
    2. Machine Learning
    3. Laminar Serverless Stream-based Framework with Deep Learning Features
    4. Dynamic and Adaptive Workflows
    5. dispel4py: Dynamic Deployment Techniques
    6. Other Research Topics

Recap - What is a scientific workflow?

- Goals: automate scientist's repetitive data management and analysis tasks
- Typical Phases:
  - Data collection, Data Processing, Data Analsys,

    Storage and visualization
  - Design, test, share, deploy, execute, reuse WF's

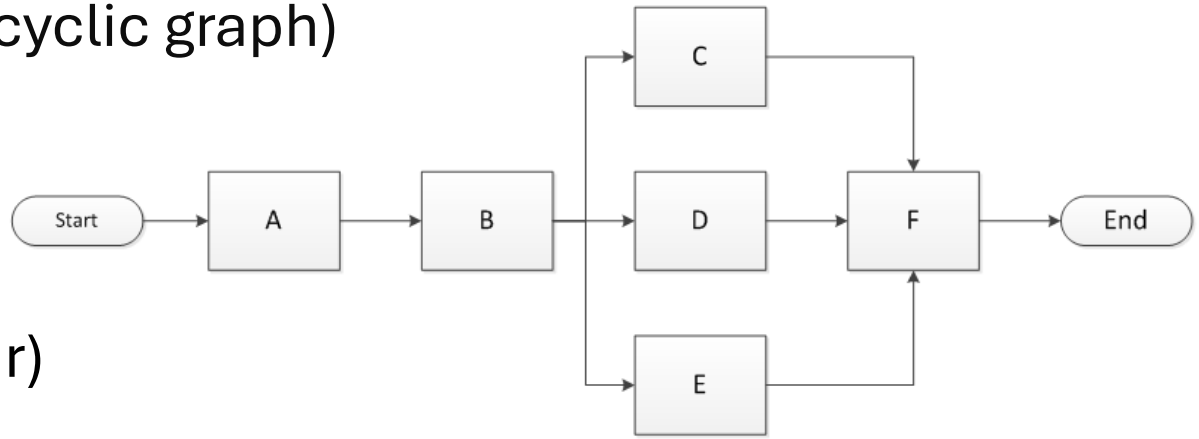


Main components of a typical distributed infrastructure steered by a WMS

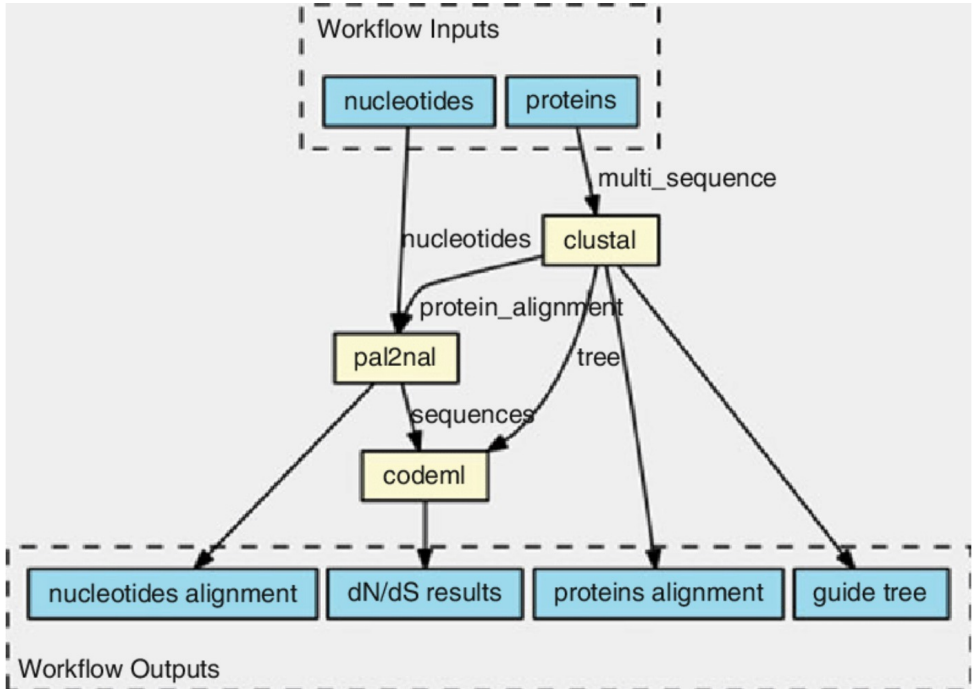
2. Beyond Basic Concepts

Understanding DAG in Workflows

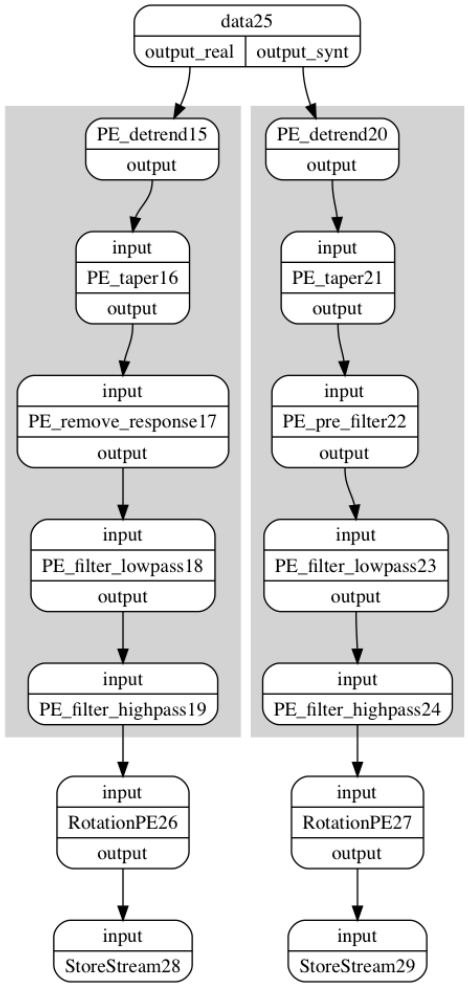
- Most WMS are based on DAG (directed acyclic graph)
  - vertices denote tasks
  - edges denote dependencies
  - unidirectional
  - never form a closed loop (non-circular)
- This non-circular structure ensures:
  - that each task is executed once
  - its prerequisites are met
  - efficient parallelization and scheduling



Understanding DAG in Workflows



Genomics CWL DAG workflow



Seismology dispel4py DAG workflow

Levels of Composition

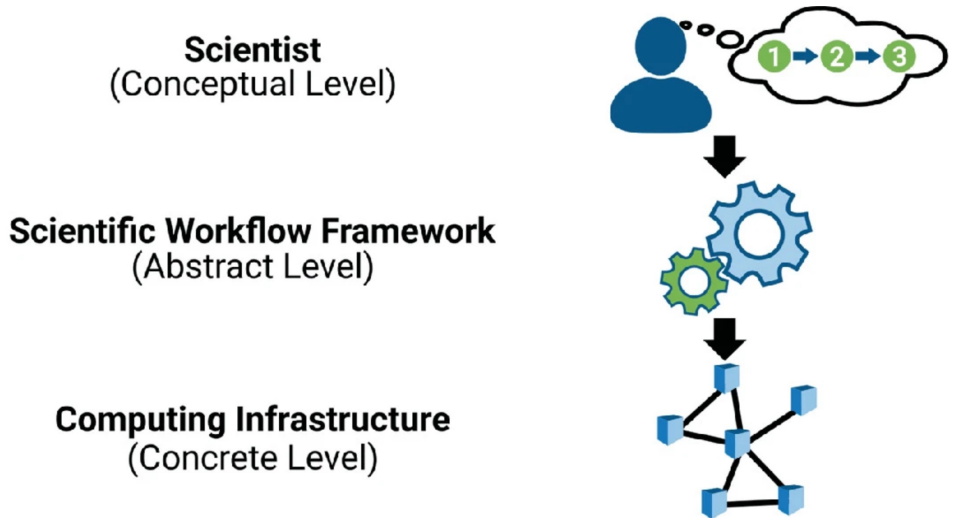
- Script-Gluing WMS:
  - **Description:** These workflows act as a 'glue' to connect standalone scripts/programs into a cohesive sequence.
  - **Usage:** Ideal for chaining existing programs or scripts that perform specific tasks into a workflow without modifying the original code.
  - **Examples:** Pegasus, CWL, Galaxy

Levels of Composition

- Programming Language-Like WMS:
  - **Description**: These workflows offer a programming language environment
    - Based on Python, C, Java, etc.
  - **Usage**: Suitable for workflows requiring complex decision-making, dynamic task generation, or intricate processing logic akin to software development.
  - **Examples**: dispel4py, PyCOMPSs, Dask, Nextflow

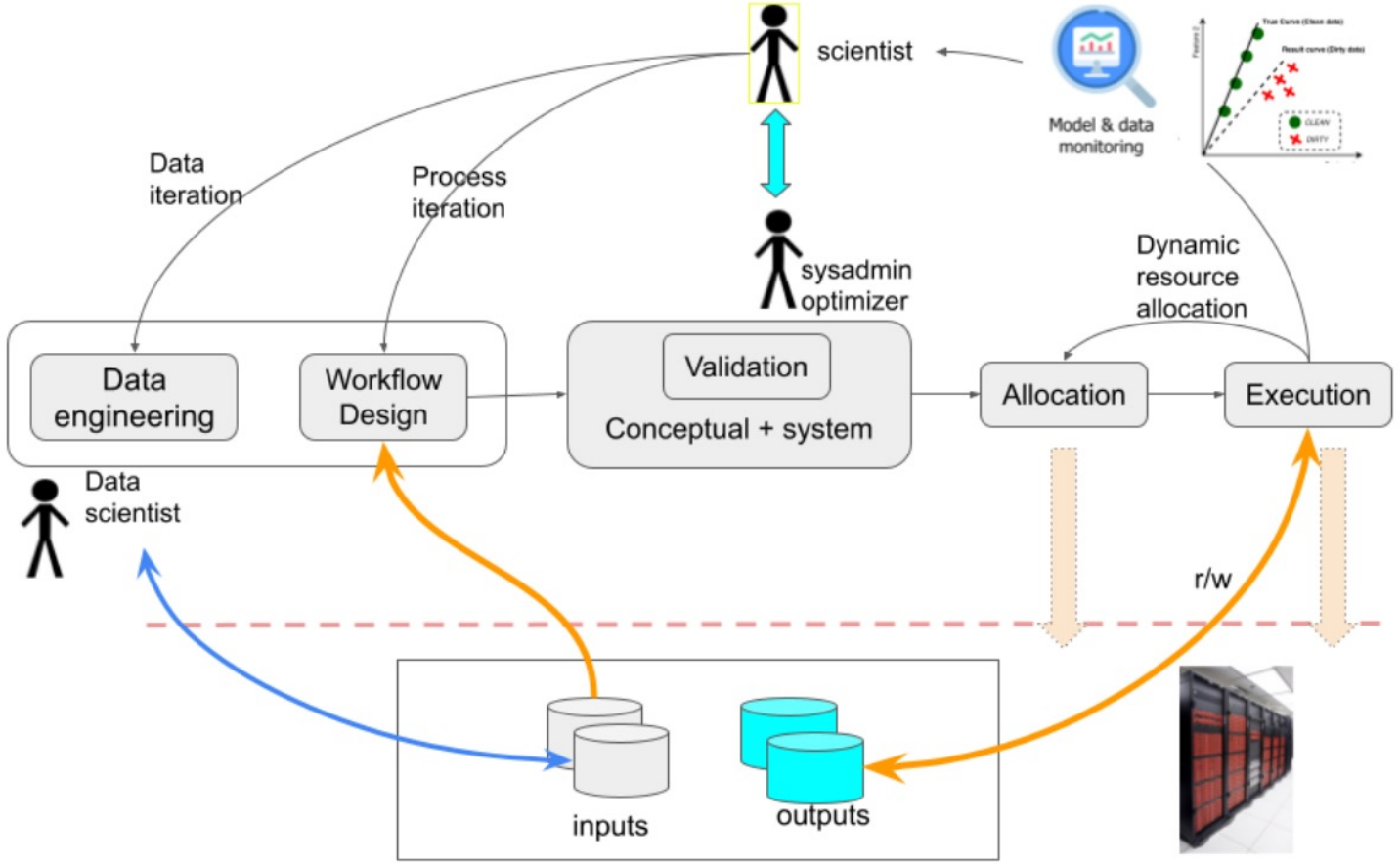
Levels of Workflow Abstraction

- **Conceptual Level:**
  - Captures the essence of the workflow's intent and the knowledge creation process.
  - Ideal for documenting and communicating the scientific approach in a structured manner.
- **Abstract Level**:
  - Defines the workflow structure and logic without binding to specific computational resources or environments.
  - Facilitates the interface between the scientific method and diverse computing infrastructures.
- **Concrete Level**:
  - Specifies exact details for execution, including data paths, environmental parameters, and computational resources.
  - Ensures workflows are ready to run in specific settings and are crucial for exploiting a wide range of data and computational platforms



Data Analysis and Exploration with Scientific Workflows

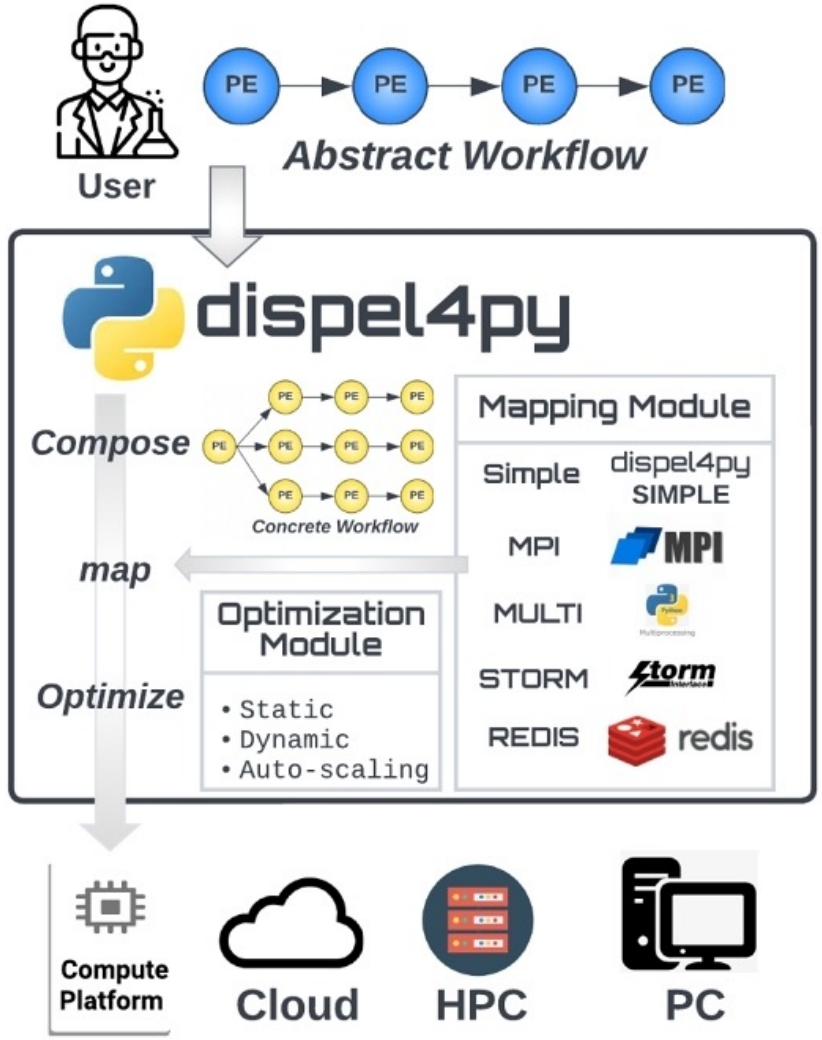
Levels of Workflow Abstraction



Simplified Framework of Stakeholder Roles in HPC Workflows

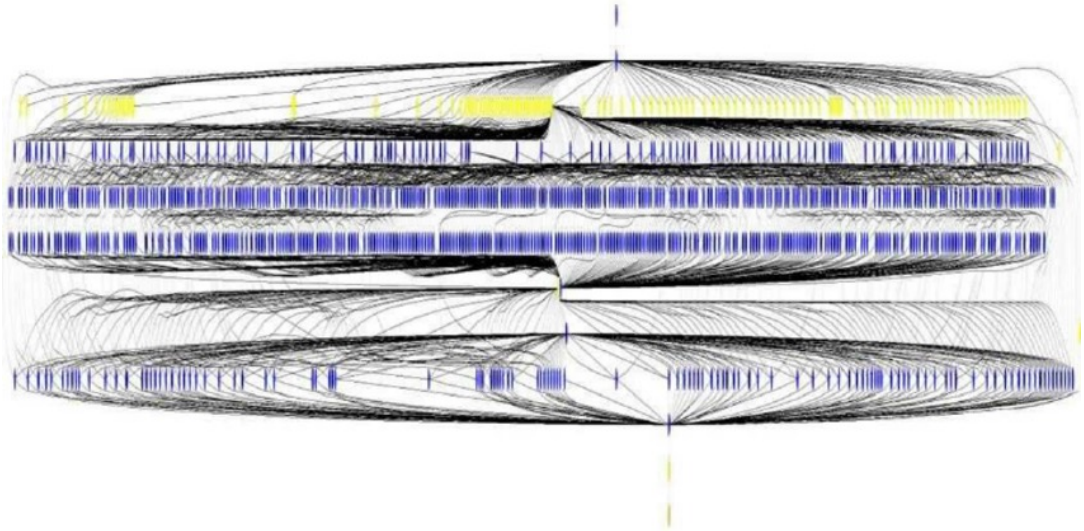
Abstract vs Concrete Workflows

- Abstract Workflows:
  - Focus on the tasks and their dependencies
    - without tying them to execution details
  - Provide flexibility to adapt to different computing environments.
  - Example: dispel4py
    - Enables the creation of abstract workflows
    - Allowing scientists to define the workflow logic once
    - Mappings: Create different concrete workflows on-the-fly depending on the “mapping” to use



Abstract vs Concrete Workflows

- Concrete Workflows:
  - Defined with specific details on execution environment
    - data paths, and computational resources
  - Ideal for workflows that run in a controlled or known environment
  - Many WMS create automatically the concrete workflows from the users abstract workflows
  - Precise details on how and where tasks should be executed



Concrete Montage workflow produced by Pegasus.
The light colored-nodes represent data stage-in
The dark colored nodes, computation.

Computing vs. Semantic Focus

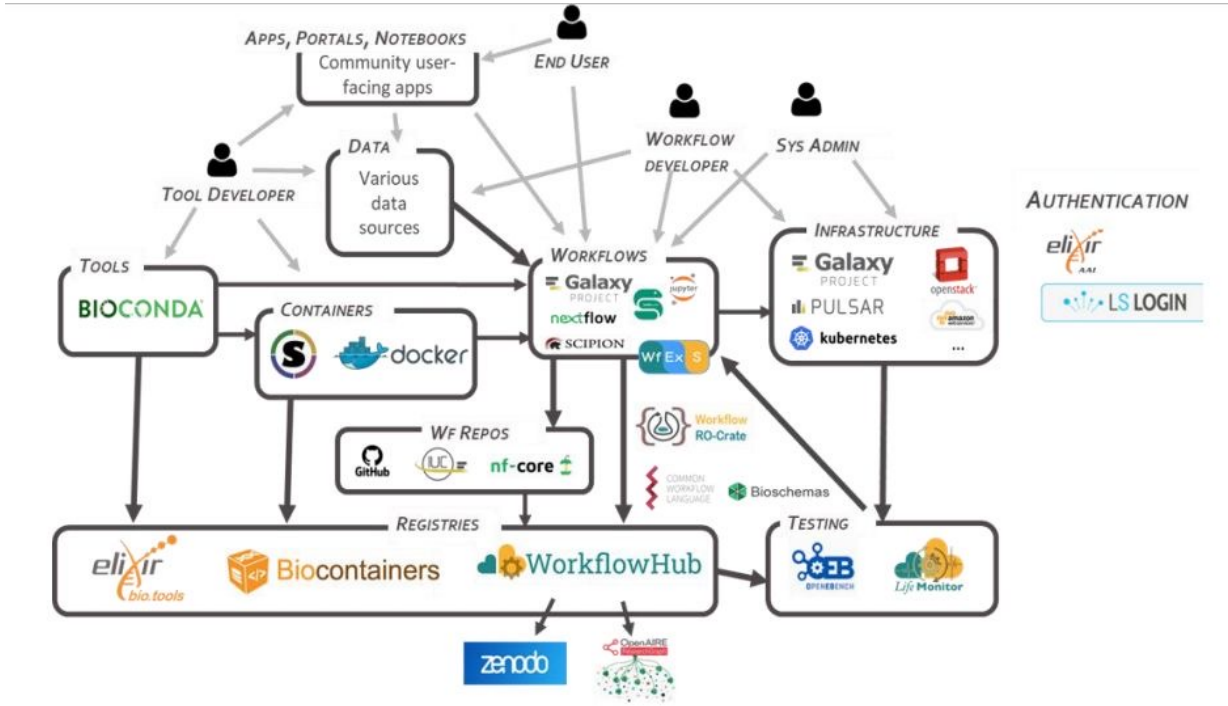
- Computing-centric workflows:
  - **Resource Optimization**: Streamlines computational resource use to
    - enhance performance
    - reduce costs
    - or both
  - **Execution Efficiency**: Focuses on the speed and reliability of tasks execution
    - incorporating strategies for error recovery and load balancing
  - **Examples**: dispel4py, Pegasus, Taverna

Computing vs. Semantic Focus

- Semantic-Focused Workflows:
  - **Data Meaning and Context**: Emphasizes the understanding of data by incorporating domain knowledge
    - often using ontologies to enrich data meaning.
  - **Data Integration and Interoperability**: Aims to seamlessly combine data from diverse sources and formats
    - facilitating collaboration across different scientific domains.
  - **Examples**: CWL, Wings
    - formal descriptions of data and operations
    - workflows are understandable by both machines and humans
    - supporting collaborative research efforts.

Sharing Workflows: Facilitating Collaboration

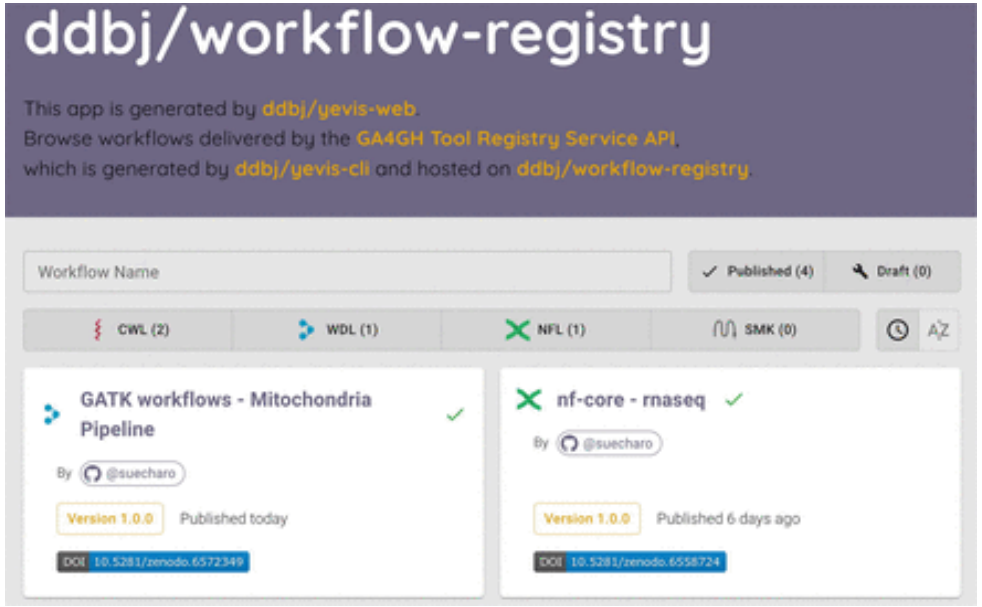
- **Collaborative Research Enhancement**:
  - Facilitate knowledge exchange and accelerate problem-solving
- **Promotion of Reproducibility**:
  - Shared workflows are a blueprint for replicating scientific results
  - Ensure transparency and verification of research processes
- **Benefits** :
  - Encourages the standardization of methods
  - Reduces duplication of effort across research groups
  - Enhances the visibility and impact of research outputs



Tools & Workflows & Sharing Facilities

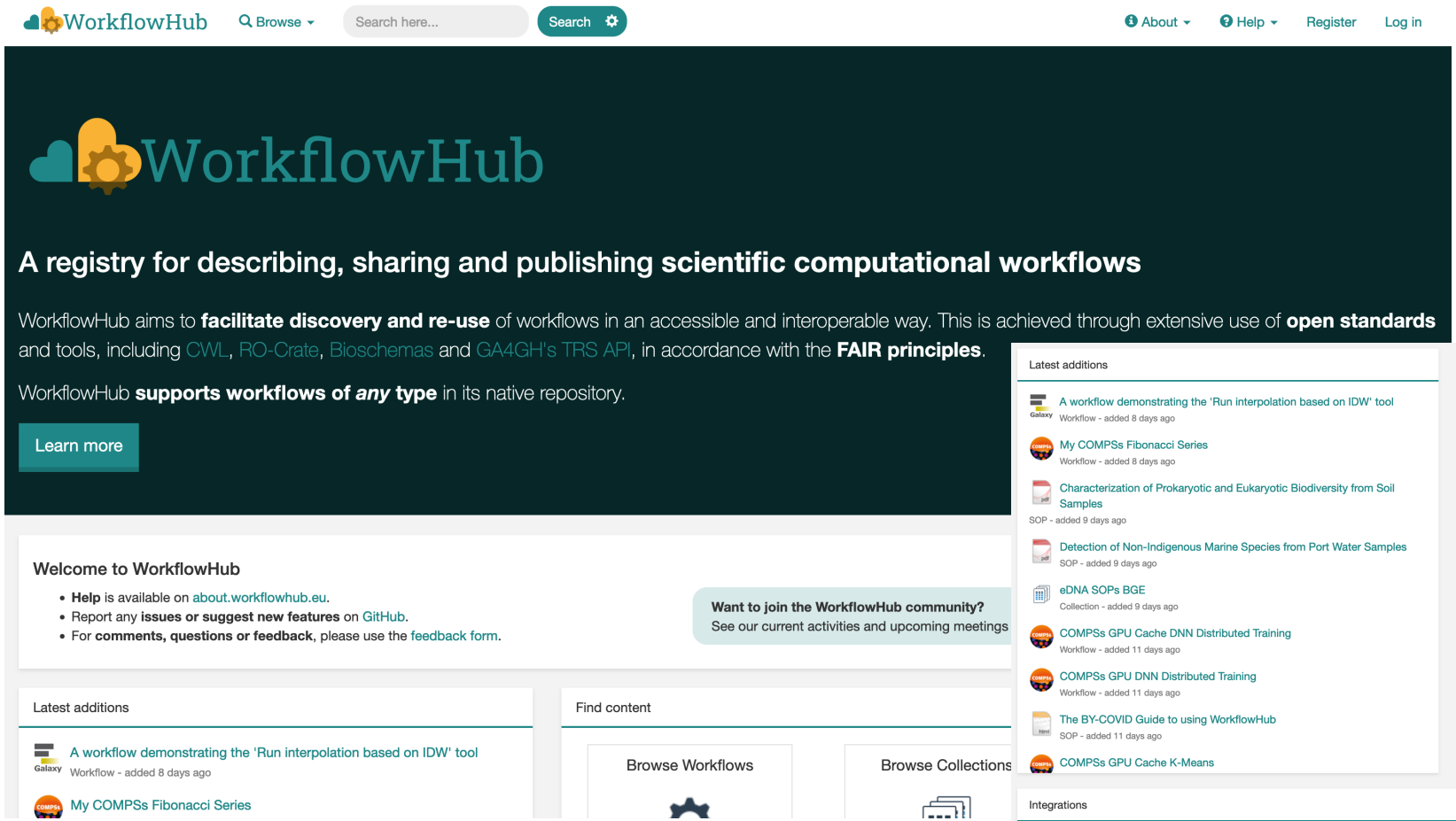
Sharing Workflows: Facilitating Collaboration

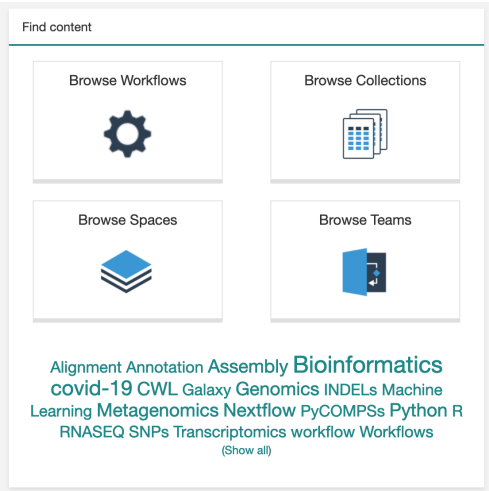
- **Workflow Registries:**
  - Serve as centralized repositories to store, manage, and retrieve workflows and their components.
  - Enable version control, access control, and the tracking of usage metrics.
- **Platforms for Workflow Sharing:**
  - WorkflowHub: A registry focused on the description, sharing, and publication of scientific workflows.
    - Facilitates community contributions and provides rich metadata.
  - MyExperiment: A collaborative environment for sharing computational workflows.
    - Offers a social platform model that supports user communities and workflow experimentation.

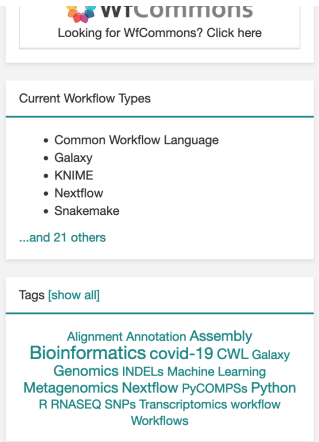


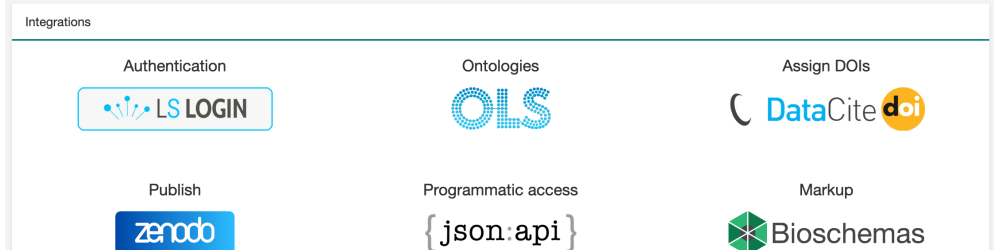
Example of a Workflow Registry

Sharing Workflows: Facilitating Collaboration





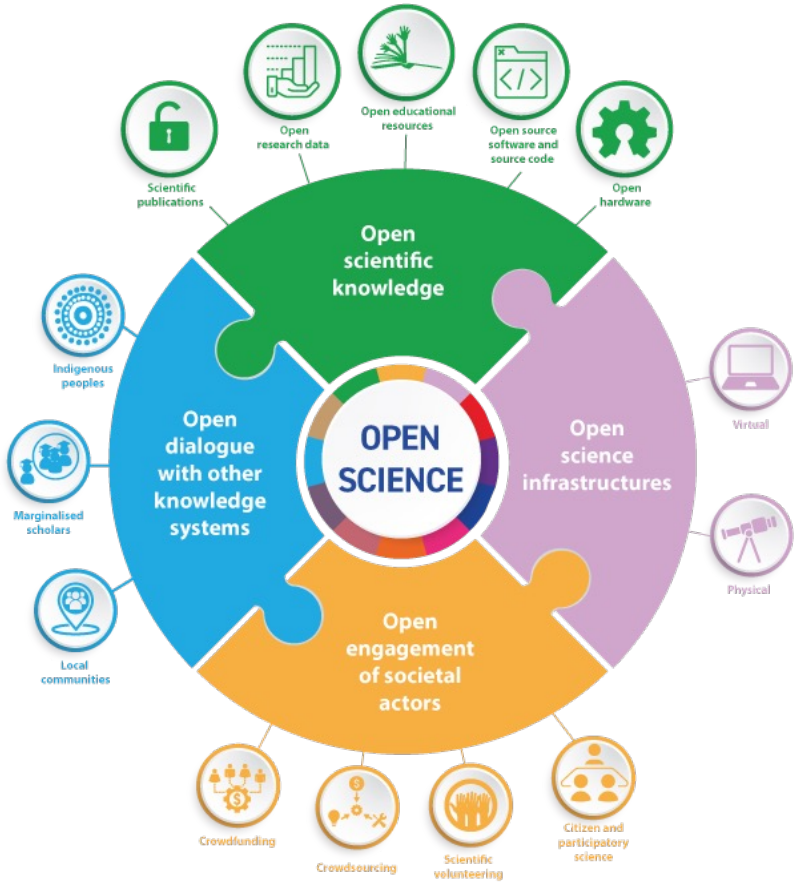




[WorkflowHub]

Fostering Open Science Workflows

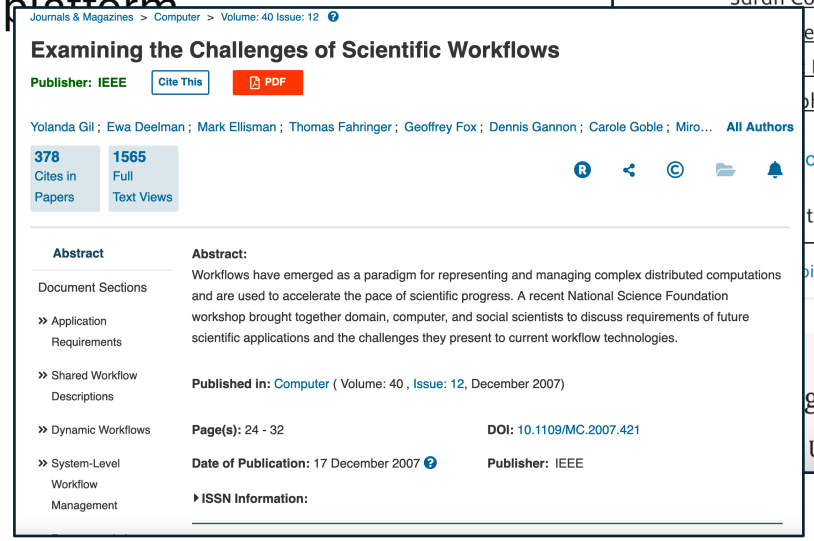
- **Open Science Principles**:
  - Emphasize transparency, accessibility, and sharing in research
  - Encourage collaboration and communication across disciplinary boundaries
- **Reproducibility in Workflows**:
  - Workflows should be repeatable with the same results by different teams
  - Incorporating detailed documentation
  - Versioning of data and code
  - Clear definitions of all workflow steps
- **Designing for Reproducibility**:
  - Utilize open formats and standards for workflow descriptions.
  - Integrate open-source tools and environments to ensure accessibility
- **Community Engagement**:
  - Recognize and reward the publication of workflows.
  - Establish repositories and registries for depositing workflows

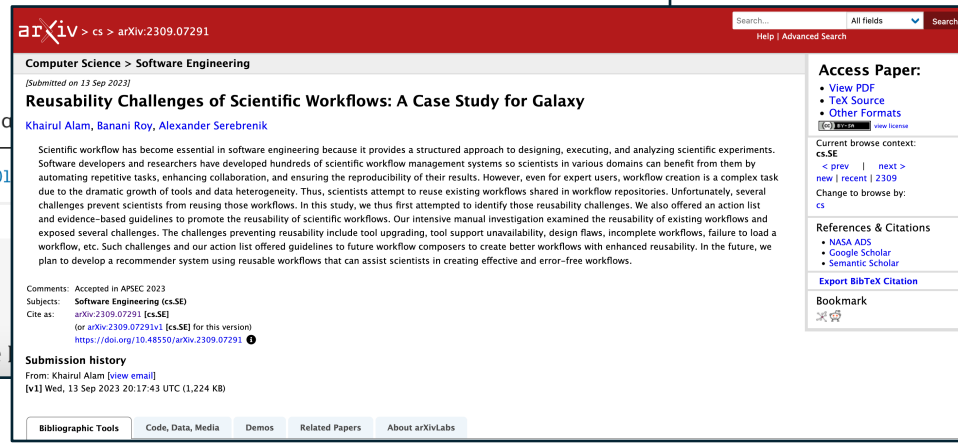


More at [Open Science Workflows]

Current Challenges

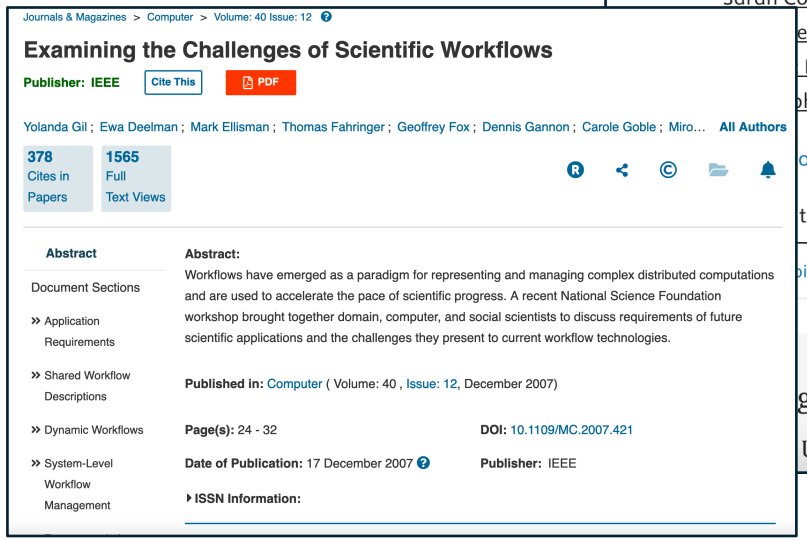
- **Scalability:**
  - Ensuring WMS needs to efficiently handle large-scale, distributed computations without performance degradation
- **Integration of Heterogeneous Data Sources:**
  - Combining data from varied formats, standards, and domains to provide a unified analysis platform

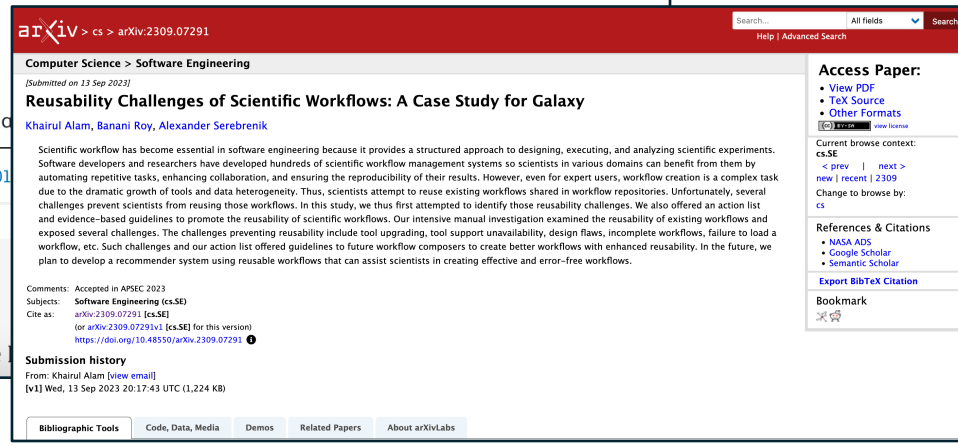




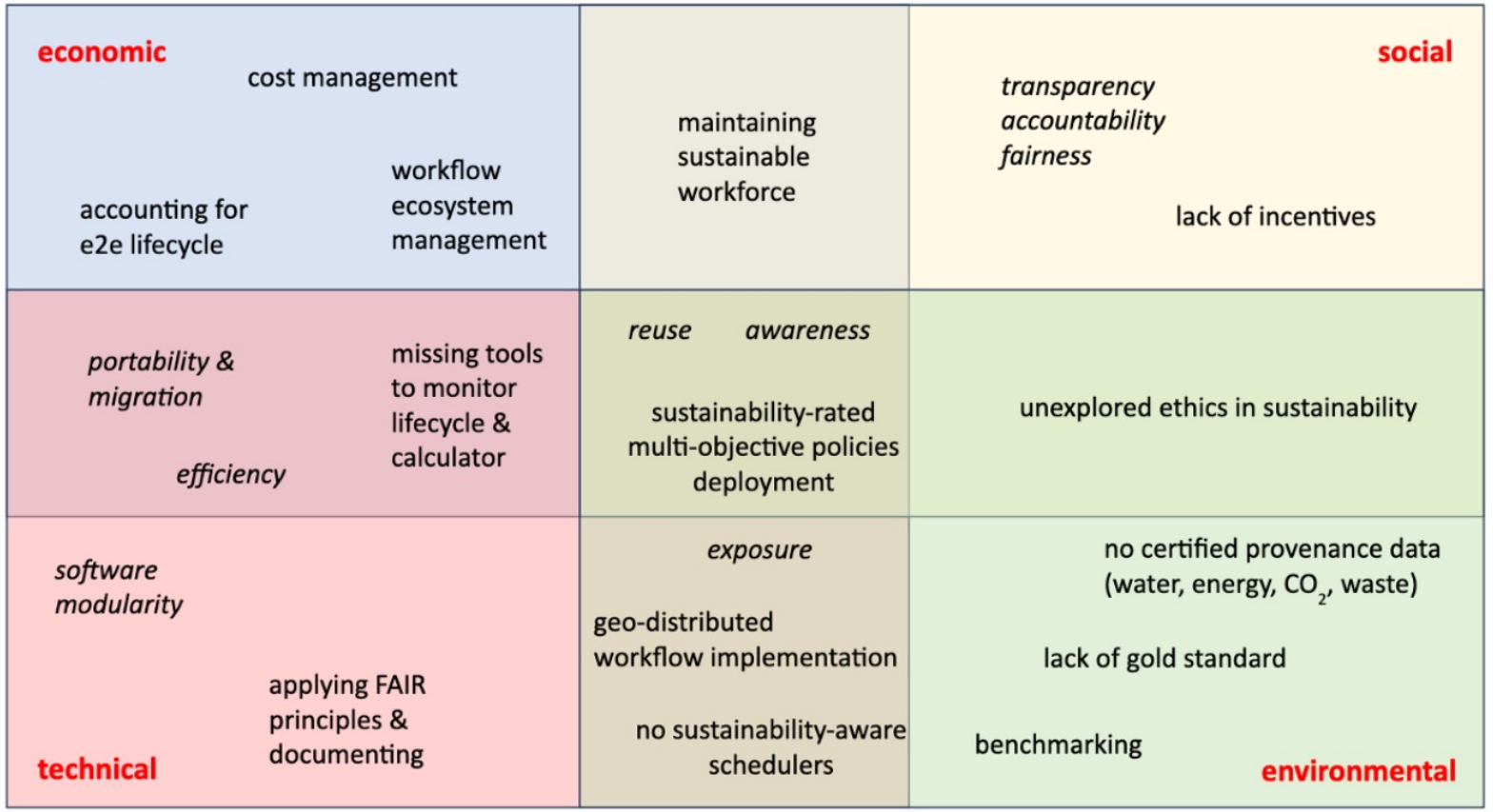
Current Challenges

- **Ensuring Security and Privacy:**
  - Protecting sensitive data and intellectual property while allowing shared access to workflow components.
- **Intuitive WMS Interfaces**
  - Creating user-friendly interfaces that cater to researchers with diverse technical backgrounds.



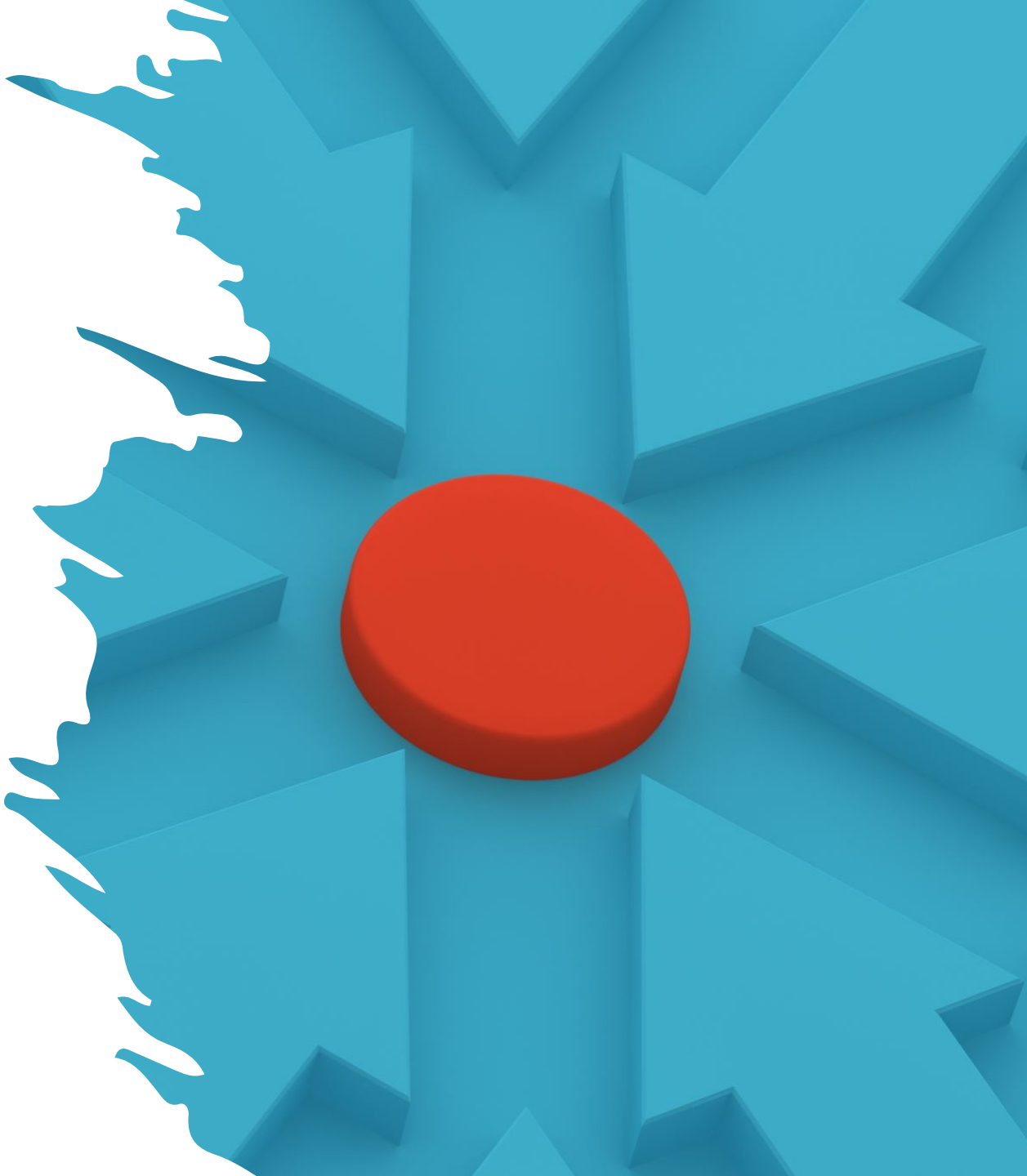


Current Challenges



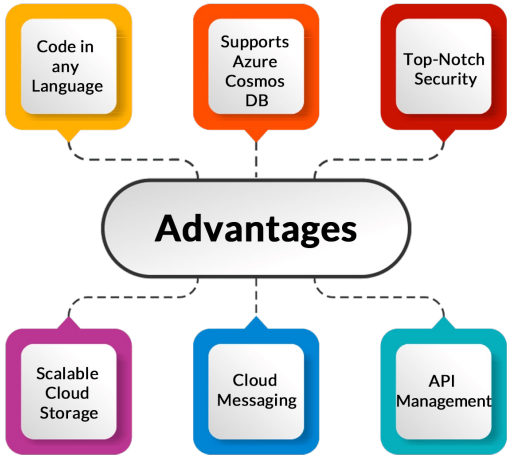
Categorized sustainability challenges in HPC+AI workflows

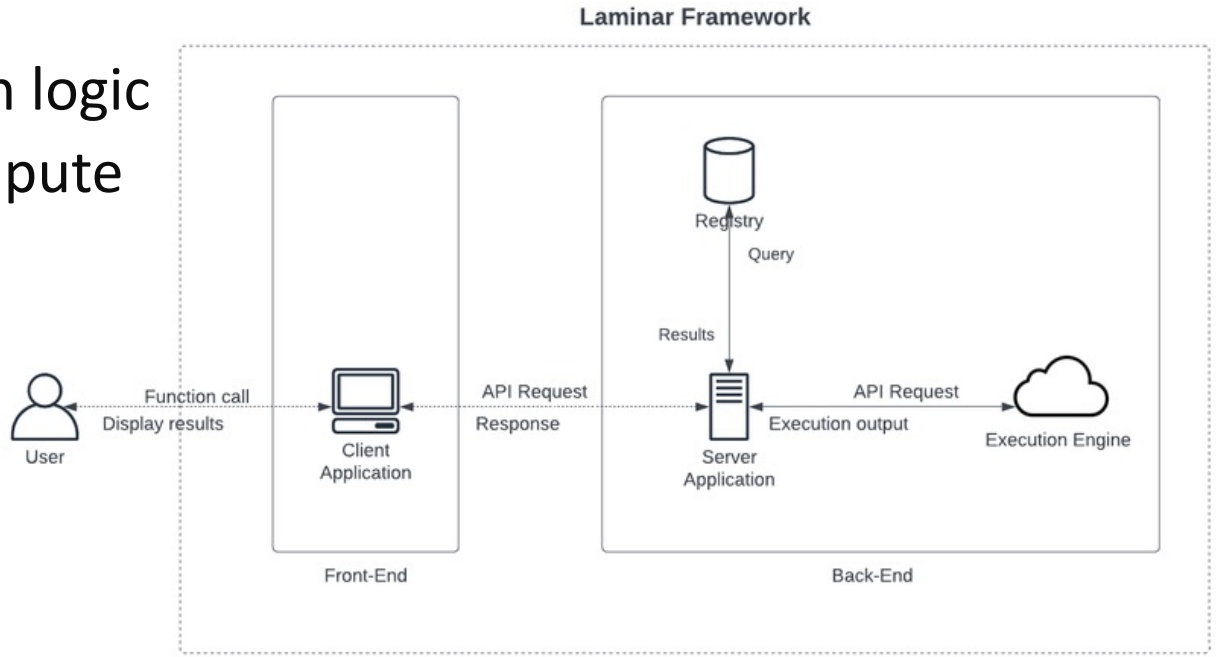
3. Research Directions



Serverless Frameworks & Workflows

- Exploring serverless computing
  - Enabling scientists to focus on research logic
  - Without managing the underlying compute resources





[Laminar: A New Serverless Stream-based Framework with Semantic Code Search and Code Completion]

ML & NLP in Workflows

- ML & NLP – smarter, efficient and searchable workflows :
  - predictive task scheduling
  - resource allocation
  - error handling
  - semantic searches of tasks/ workflows:
    - Code search
    A. Text search
  - Task code auto-completion

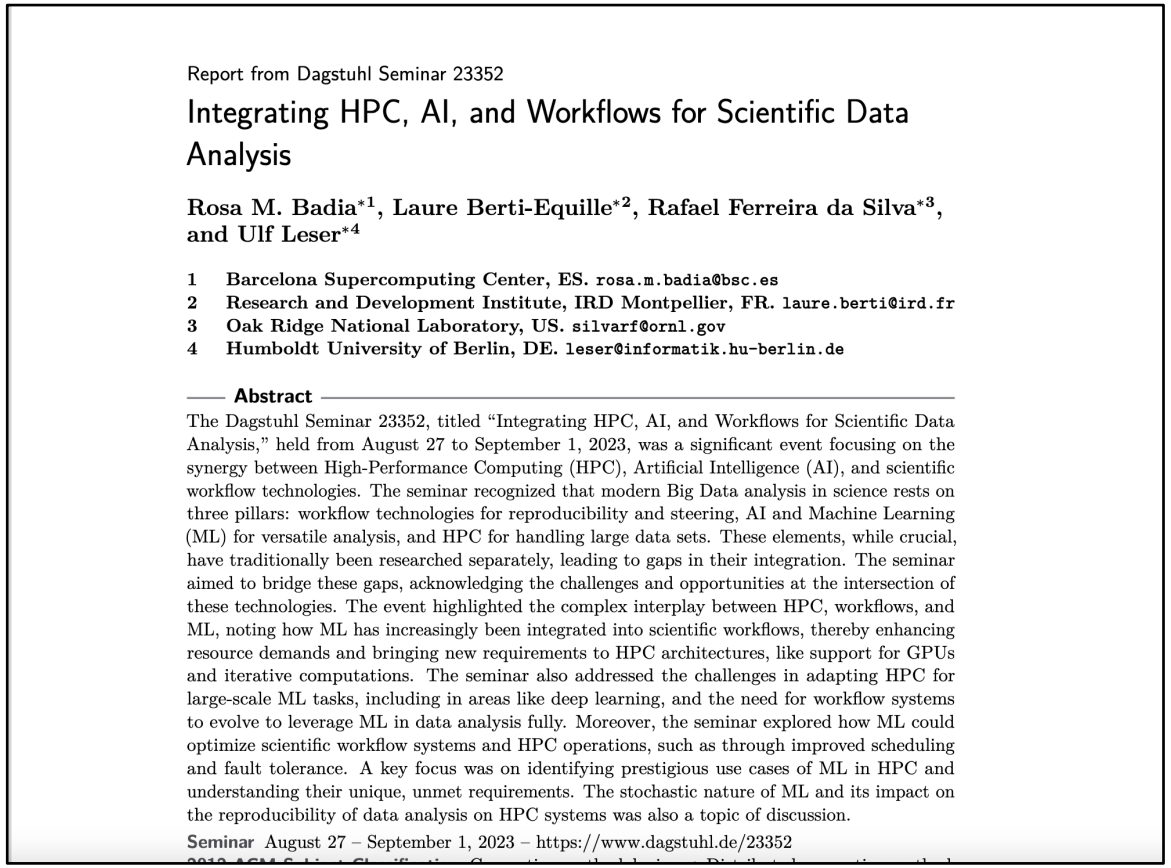
Dagstuhl Seminar 23352

Integrating HPC, AI, and Workflows for Scientific Data Analysis

(Aug 27 – Sep 01, 2023)

ML & NLP in Workflows

- ML & NLP – smarter, efficient and searchable workflows :
  - predictive task scheduling
  - resource allocation
  - error handling
  - semantic searches of tasks/ workflows:
    - Code search
    - Text search
  - Task code auto-completion



Report from Dagstuhl Seminar 23352

Integrating HPC, AI, and Workflows for Scientific Data Analysis

Rosa M. Badia[*1], Laure Berti-Equille[*2], Rafael Ferreira da Silva[*3], and Ulf Leser[*4]

1 **Barcelona Supercomputing Center, ES.** rosa.m.badia@bsc.es
2 **Research and Development Institute, IRD Montpellier, FR.** laure.berti@ird.fr
3 **Oak Ridge National Laboratory, US.** silvarf@ornl.gov
4 **Humboldt University of Berlin, DE.** leser@informatik.hu-berlin.de

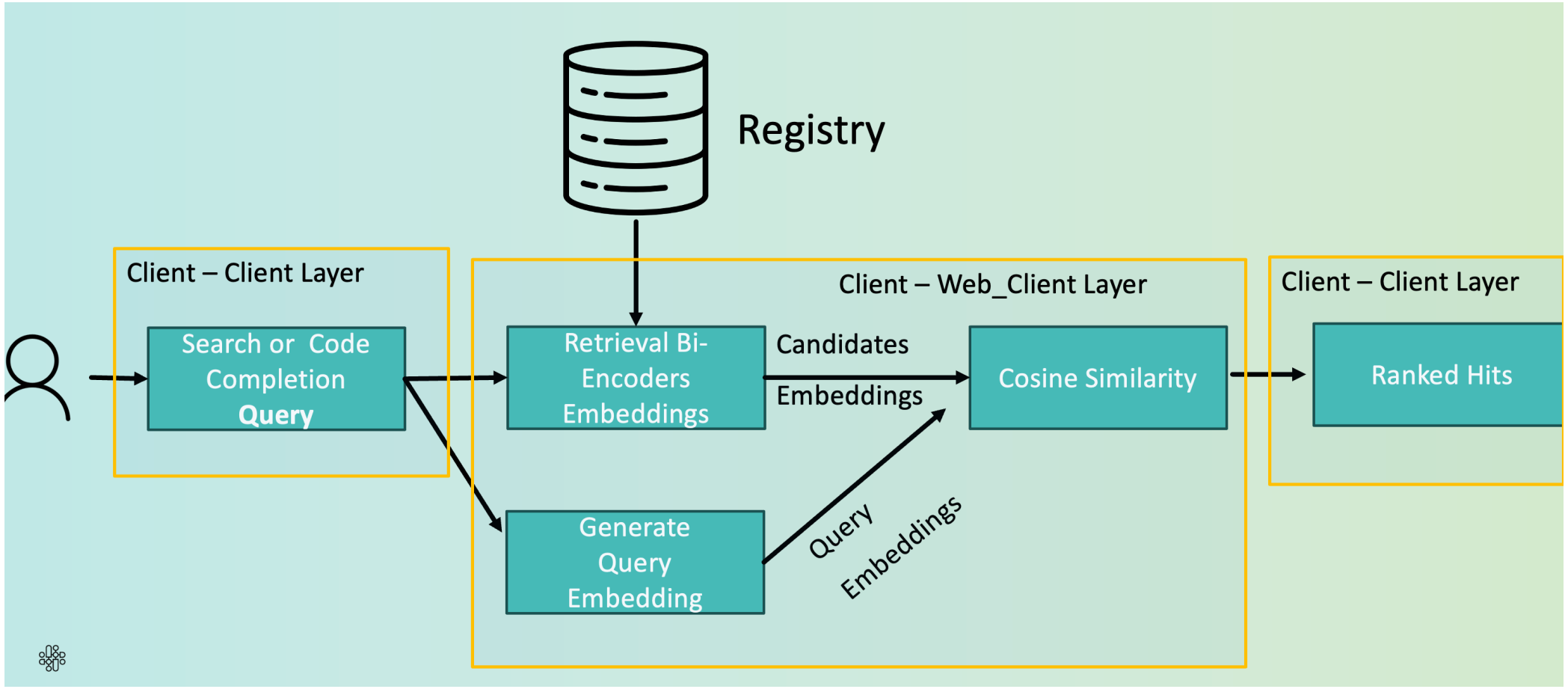
Abstract

The Dagstuhl Seminar 23352, titled "Integrating HPC, AI, and Workflows for Scientific Data Analysis," held from August 27 to September 1, 2023, was a significant event focusing on the synergy between High-Performance Computing (HPC), Artificial Intelligence (AI), and scientific workflow technologies. The seminar recognized that modern Big Data analysis in science rests on three pillars: workflow technologies for reproducibility and steering, AI and Machine Learning (ML) for versatile analysis, and HPC for handling large data sets. These elements, while crucial, have traditionally been researched separately, leading to gaps in their integration. The seminar aimed to bridge these gaps, acknowledging the challenges and opportunities at the intersection of these technologies. The event highlighted the complex interplay between HPC, workflows, and ML, noting how ML has increasingly been integrated into scientific workflows, thereby enhancing resource demands and bringing new requirements to HPC architectures, like support for GPUs and iterative computations. The seminar also addressed the challenges in adapting HPC for large-scale ML tasks, including in areas like deep learning, and the need for workflow systems to evolve to leverage ML in data analysis fully. Moreover, the seminar explored how ML could optimize scientific workflow systems and HPC operations, such as through improved scheduling and fault tolerance. A key focus was on identifying prestigious use cases of ML in HPC and understanding their unique, unmet requirements. The stochastic nature of ML and its impact on the reproducibility of data analysis on HPC systems was also a topic of discussion.

Seminar August 27 – September 1, 2023 – https://www.dagstuhl.de/23352

Integrating HPC, AI, and Workflows for Scientific Data Analysis

Laminar: Serverless Stream-based Framework with Deep Learning Features



[Laminar: A New Serverless Stream-based Framework with Semantic Code Search and Code Completion]

Laminar: Serverless Stream-based Framework with Deep Learning Features

Text-based Search

```
client.search_Registry("prime", "workflow")
```

```
Searched for "prime"

REGISTRY

Result 1: ID: 2
Workflow Name: isPrime
Description: Workflow that prints random prime numbers
```

Semantic PE Code Search

```
client.search_Registry("A PE that checks if
                        a number is prime",
                        "pe","text")
```

Searched for "A PE that check if a number is prime"

peId	peName	description	similarity
3	IsPrime	check if the input is a prime or not	0.796563
17	TestProducer	This PE produces a range of numbers	0.502303
10	InternalExtinction	function to calculate internal extinction from...	0.216504
19	TestDelayOneInOneOut	This method is called by the PE base class to ...	0.213435
18	TestOneInOneOut	This PE copies the input to an output	0.186635

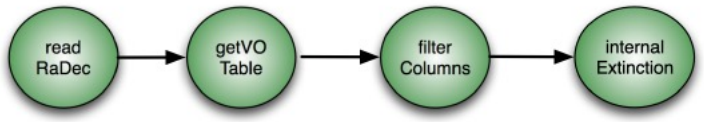
Automatic Code Completion

```
client.search_Registry("random.randint(1, 1000)",
                        "pe","code")
```

Searched for "random.randint(1, 1000)"

peId	peName	description	similarity
4	NumberProducer	This function is called to generate a random s...	0.752691
5	PrintPrime	Process the sequence of words in the sequence ...	0.671739
22	TestMultiProducer	Process the sequence of sequence sequence sequ...	0.635483
3	IsPrime	check if the input is a prime or not	0.619670
17	TestProducer	This PE produces a range of numbers	0.552182

Laminar: Serverless Stream-based Framework with Deep Learning Features



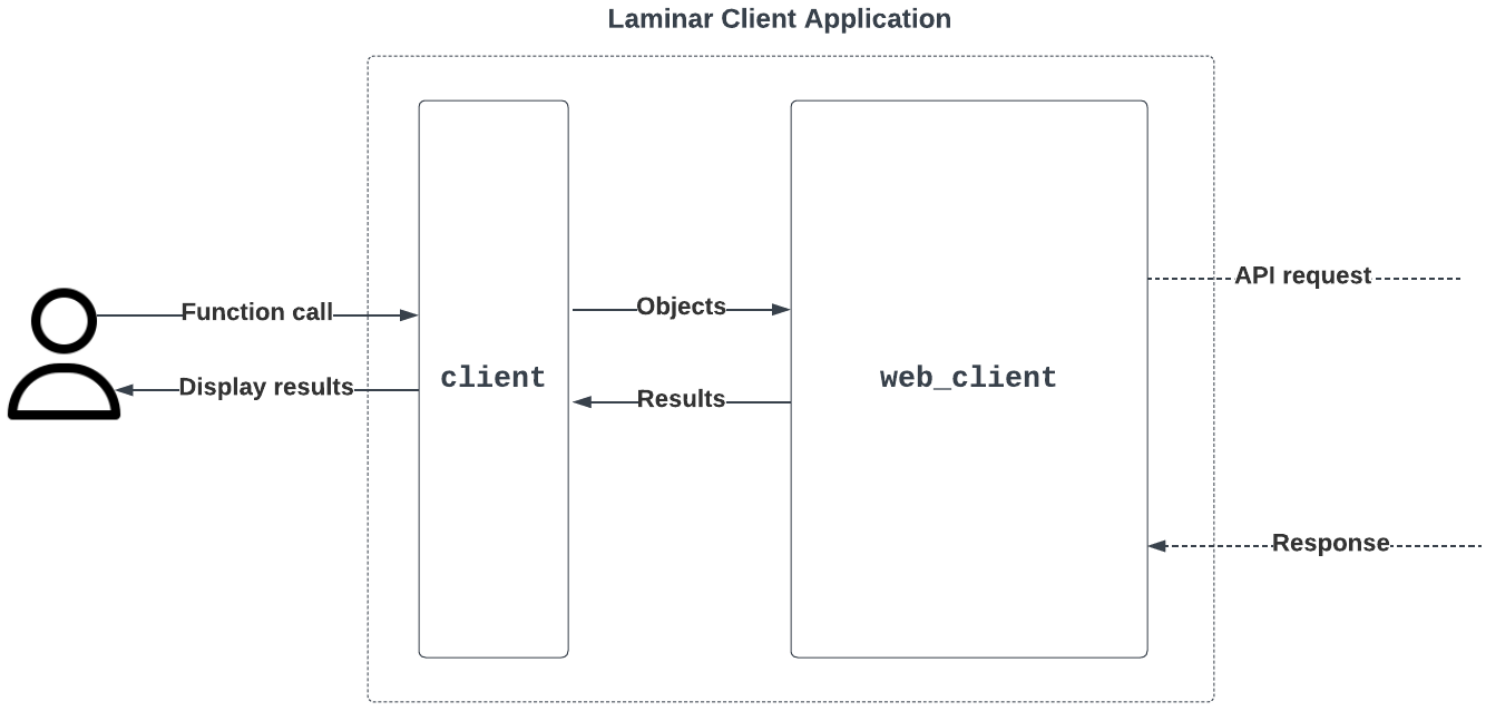
Streaming workflow for calculating the internal extinction of galaxies

```
graph.connect(read, 'output', votab, 'input')
graph.connect(votab, 'output', filt, 'input')
graph.connect(filt, 'output', intext, 'input')
```

```
client.register_Workflow(
    graph,
    "Astrophysics",
    "A workflow to compute the
    internal extinction of galaxies")
```

```
workflow = client.get_Workflow("Astrophysics")
```

```
client.run(workflow,
   input=[{"input":"resources/coordinates.txt"}],
   process=REDIS,
   args={'num':10}
   resources=True)
```



Dynamic and Adaptive Workflows

- Understanding how workflows can dynamically adjust to
  - changing data size
  - changing velocity of data
  - computational environments
- Leading to more resilient scientific processes

dispel4py: Dynamic Deployment Techniques

Optimization towards Efficiency and Stateful of dispel4py

Liang Liang*, Heting Zhang†, Guang Yang*, Thomas Heinis*, Rosa Filgueira†
* Imperial College London
† University of St Andrews

ABSTRACT

Scientific workflows bridge scientific challenges with computational resources. While dispel4py, a stream-based workflow system, offers mappings to parallel enactment engines like MPI or Multiprocessing, its optimization primarily focuses on dynamic process-to-task allocation for improved performance. An efficiency gap persists, particularly with the growing emphasis on conserving computing resources. Moreover, the existing dynamic optimization lacks support for stateful applications and grouping operations.

To address these issues, our work introduces a novel hybrid approach for handling stateful operations and groupings within workflows, leveraging a new Redis mapping. We also propose an auto-scaling mechanism integrated into dispel4py's dynamic optimization. Our experiments showcase the effectiveness of auto-scaling optimization, achieving efficiency while upholding performance. In the best case, auto-scaling reduces dispel4py's runtime to 87% compared to the baseline, using only 76% of process resources. Importantly, our optimized stateful dispel4py demonstrates a remarkable speedup, utilizing just 32% of the runtime compared to the contender.

KEYWORDS

scientific workflow, stream-based workflow, workflow optimization, auto-scaling, stateful application, dispel4py

ACM Reference Format:

Liang Liang*, Heting Zhang†, Guang Yang*, Thomas Heinis*, Rosa Filgueira†, * Imperial College London, † University of St Andrews . 2023. Optimization

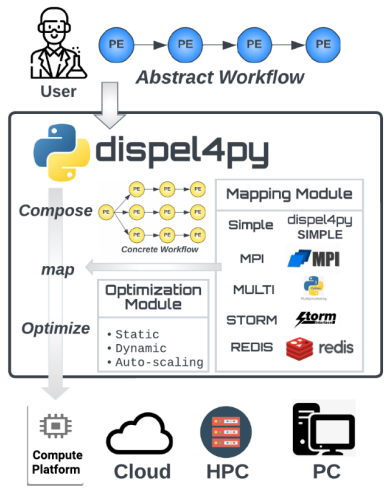
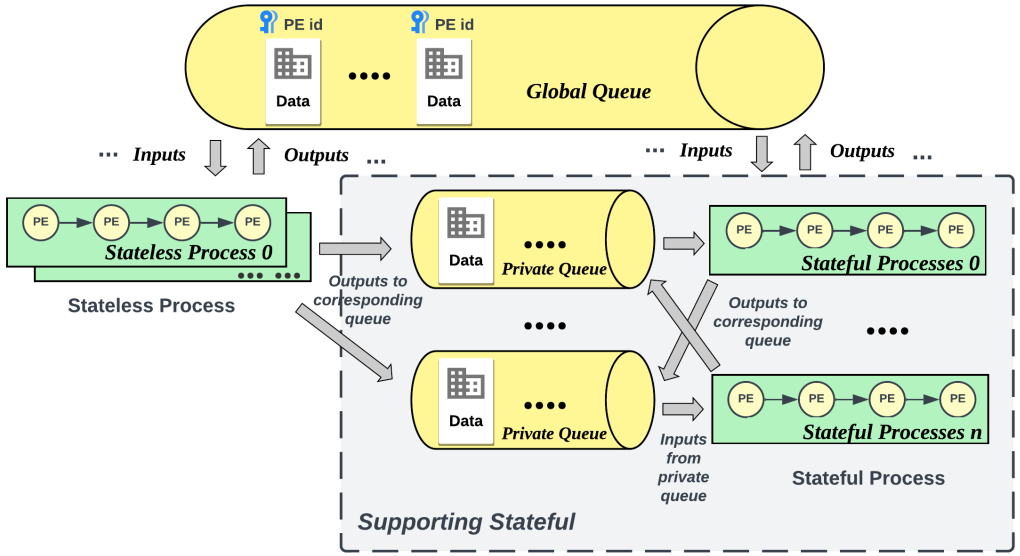


Figure 1: dispel4py overview. From abstract workflow design to automatic concrete workflow generation and execution.

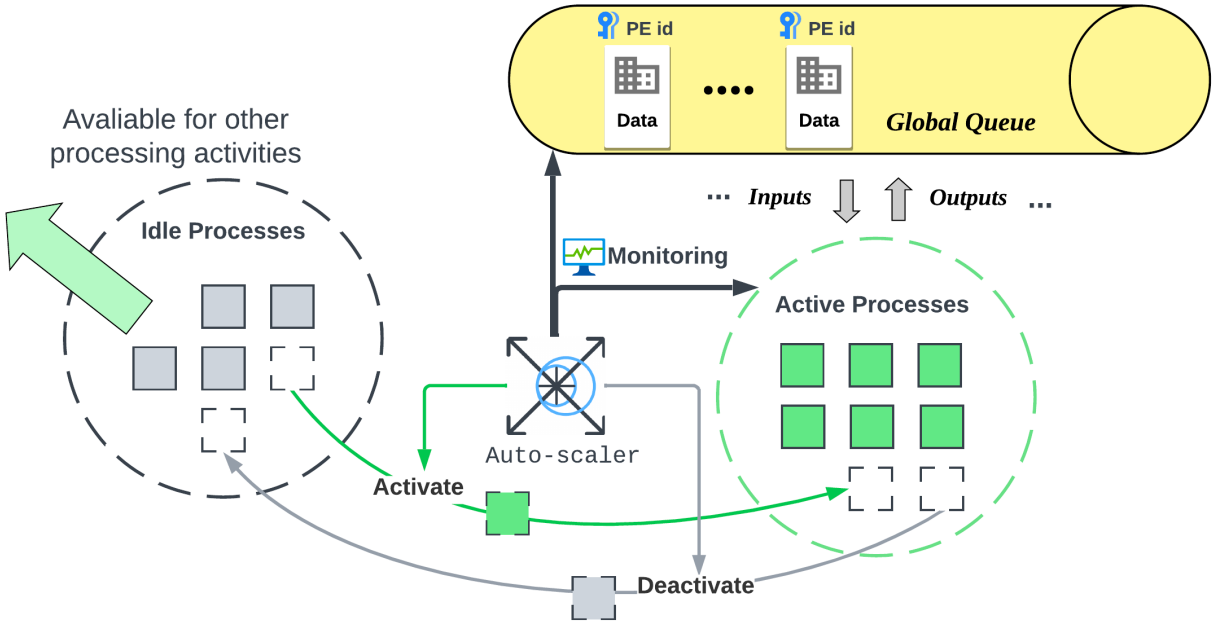
are promoted as a means to design and propose various workflow systems, effectively concealing intricate technical details, in order

[Optimization towards Efficiency and Stateful of dispel4py]

dispel4py: Dynamic Deployment Techniques



dispel4py's dynamic deployment



dispel4py's auto-scaling

New: Currently – developing new Heuristics for energy efficiency

Others Research Topics

- **FAIR workflows**:
  - Delving into the principles of Findability, Accessibility, Interoperability, and Reusability (FAIR)
- **Exascale workflows**
  - Exascale computing brings new possibilities and challenges to workflows
- **Energy-Efficient Workflow Design**
  - New strategies for designing workflows that minimize energy consumption without compromising performance
- And more …

Acknowledgements

- *Tools & Workflows*, EOSC
- Introduction to CWL, Melbourne Bioinformatics
- Pegasus: Mapping Scientific Workflows onto the Grid, Pegasus Team
- Integrating HPC, AI, and Workflows for Scientific Data Analysis , Dagstuhl Seminar 23352
- *Workflows Community Summits*
- *Open Science Workflows*
- *Optimization towards Efficiency and Stateful of dispel4py*
- Laminar: A New Serverless Stream-based Framework with Semantic Code Search and Code Completion