Programming Distributed Computing Platforms with COMPSs

Rosa M. Badia, Javier Conejero

Workflows & Distributed Computing Group

07/02/2023

Winter School, Barcelona

Outline

Agenda

- Presentation of the tutorial presenters
- Introduction to COMPSs (20 min)
- PyCOMPSs: Writing Python applications (1hour)
- Break (15 min)
- Hands-on MN (1 hour)
- PyCOMPSs installation (15 min)
- SLIDES
  - http://compss.bsc.es/releases/tutorials/tutorial-WINTER_SCHOOL_2023/

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

INTRODUCTION

Motivation

- New complex architectures constantly emerging
  - With their own way of programming them
    - Fine grain: e.g. Programming models and APIs to run with GPUs, NVMs (Non-Volatile Memories)
    - Coarse grain: e.g. APIs to deploy in Clouds
  - **Difficult** for programmers
    - Higher learning curve / Time To Market (TTM)
    - What about non computer scientists???
  - **Difficult** to understand what is going on during execution
    - Was it fast? Could it be even faster? Am I paying more than I should? (**Efficiency**)
  - Tune your application for each architecture (or cluster)
    - E.g. partitioning data among nodes

Motivation

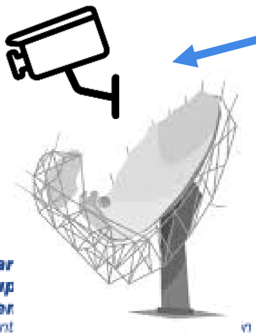
- Resources that appear and disappear
  - How to dynamically add/remove nodes to the infrastructure
- Heterogeneity
  - Different HW characteristics (performance, memory, etc)
  - Different architectures -> compilation issues
- Network
  - Different types of networks
  - Instability
- Trust and Security
- Power constraints from the devices in the edge
- Data & Storage

AI everywhere

HPC
Exascale computing
Cloud

Fog devices

Edge devices

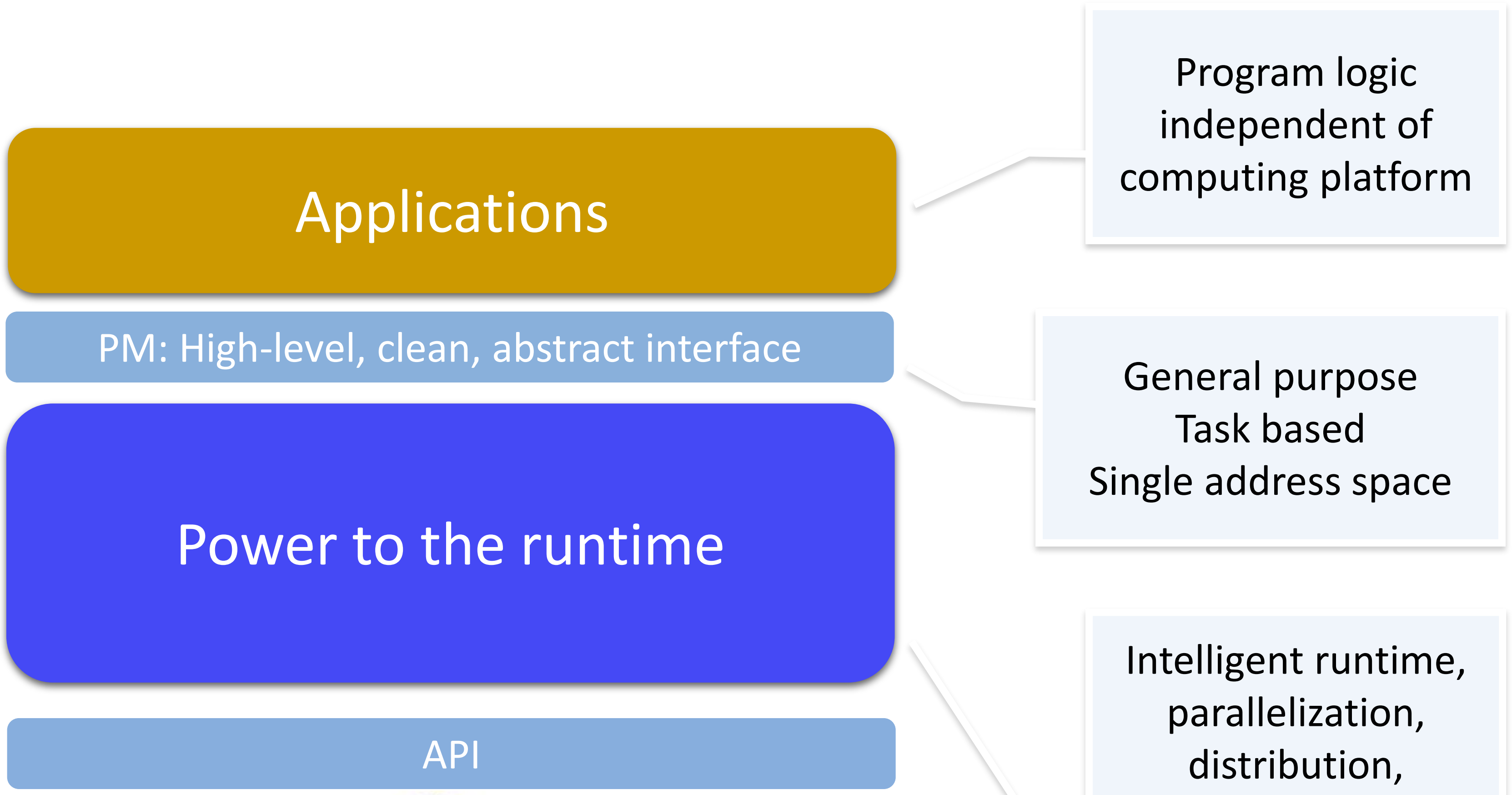


Sensors
Instruments
Actuators

Motivation

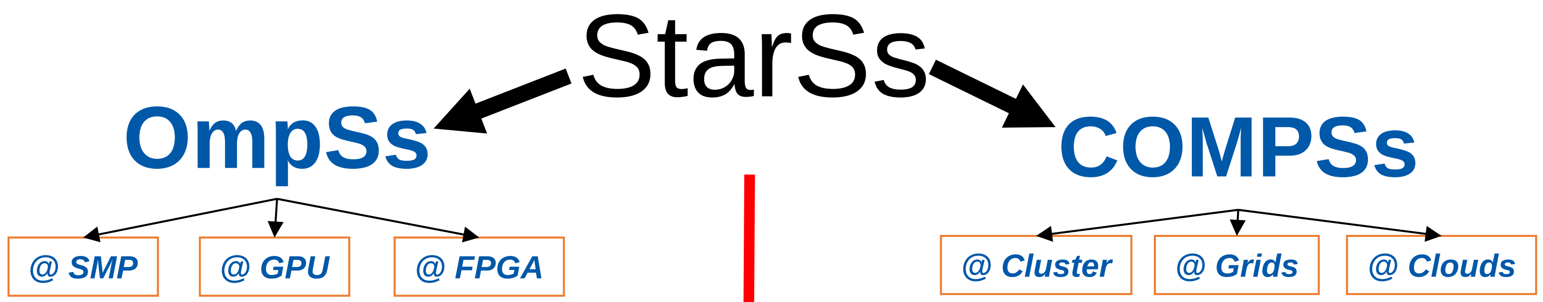
- Create tools that make developers’ life **easier**
  - Allow developers to focus on their problem
  - Intermediate layer: let the difficult parts to those tools
    - Act on behalf of the user
    - Distribute the work through resources
    - Deal with architecture specifics
    - Automatically improve performance
  - Tools for visualization
    - Monitoring
    - Performance analysis
  - Integration of computational workloads, with machine learning and data analytics

BSC vision on programming models


Applications
PM: High-level, clean, abstract interface
Power to the runtime
API
Program logic independent of computing platform
General purpose
Task based
Single address space
Intelligent runtime, parallelization, distribution, interoperability

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

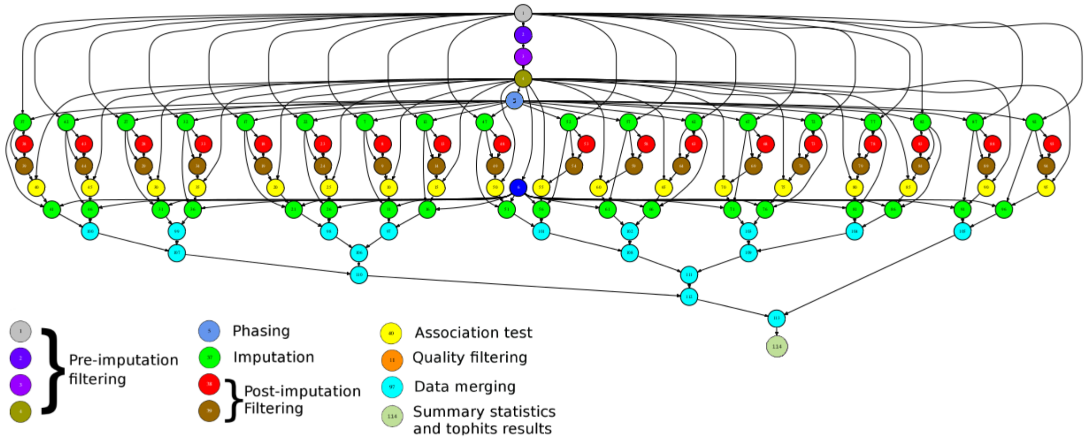
BSC vision on programming models



Average task Granularity: 100 us – 10 ms	10 ms - 1 day
Dependences: Memory address space	Files, Objects
Language bindings: C, C++, FORTRAN	Java, C/C++, Python
Intranode	**Distributed**

Programming with COMPSs

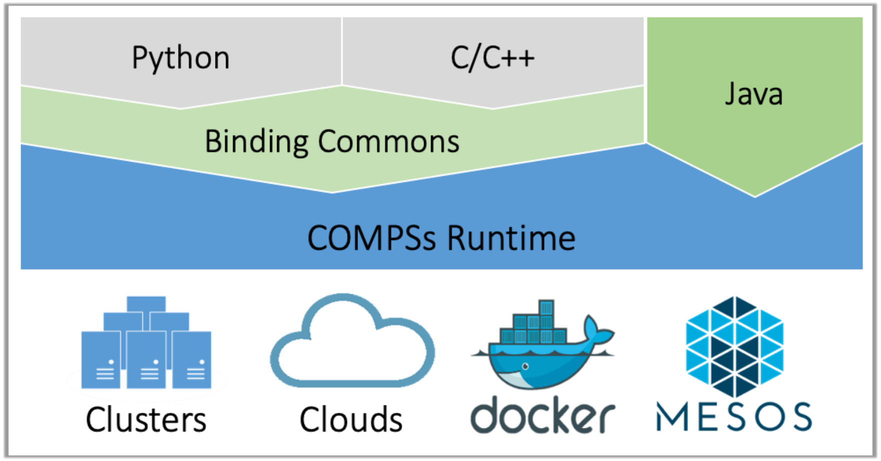
- Sequential programming
- General purpose programming language + annotations/hints
  - To identify tasks and directionality of data
- Task based: task is the unit of work
- Simple linear address space
- Builds a task graph at runtime that express potential concurrency
  - Implicit workflow
- Exploitation of parallelism
  - … and of distant parallelism
- Agnostic of computing platform
  - Enabled by the runtime for clusters, clouds and grids



Barcelona Supercomputing Center
Centro Nacional de Supercomputación

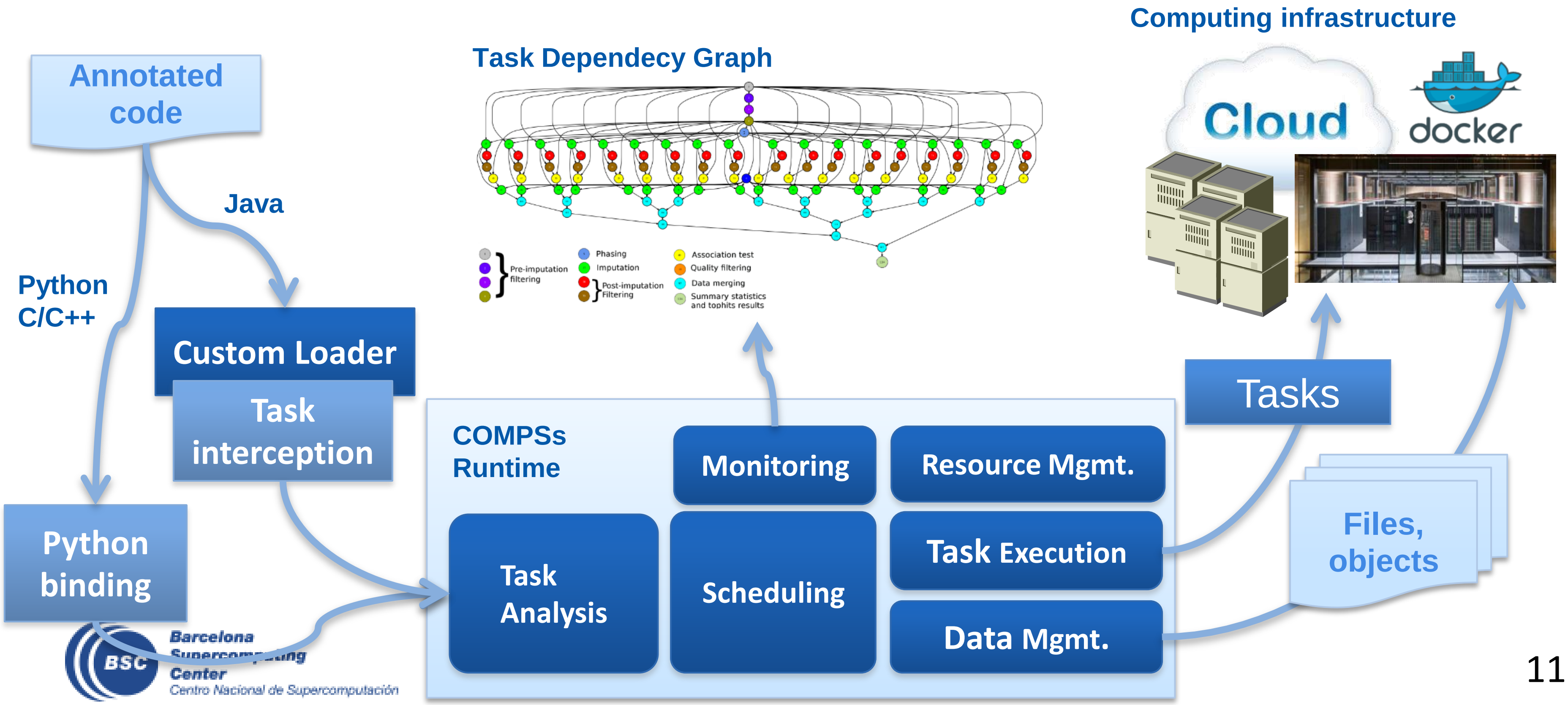
Programming with COMPSs

- Support for other types of parallelism
  - Threaded tasks (I.e., MKL kernels)
  - MPI applications -> tasks that involve several nodes
  - Integration with BSC OmpSs
  - Streaming tasks for data flow executions
- Support to Failure Management
- Parallel Machine Learning with dislib
- Available in MareNostrum and other supercomputers in Europe, in the EGI Federated Cloud and in Chameleon Cloud



COMPSs runtime

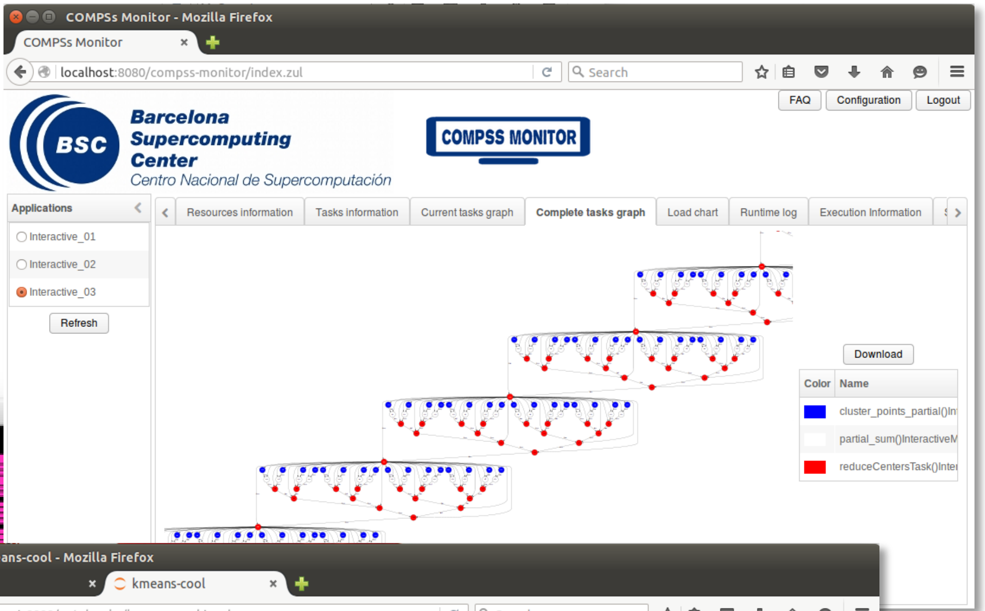
- PyCOMPSs/COMPSs applications executed in distributed mode following the master-worker paradigm
- Sequential execution starts in master node
- Tasks are offloaded to worker nodes
- All data scheduling decisions and data transfers are performed by the runtime



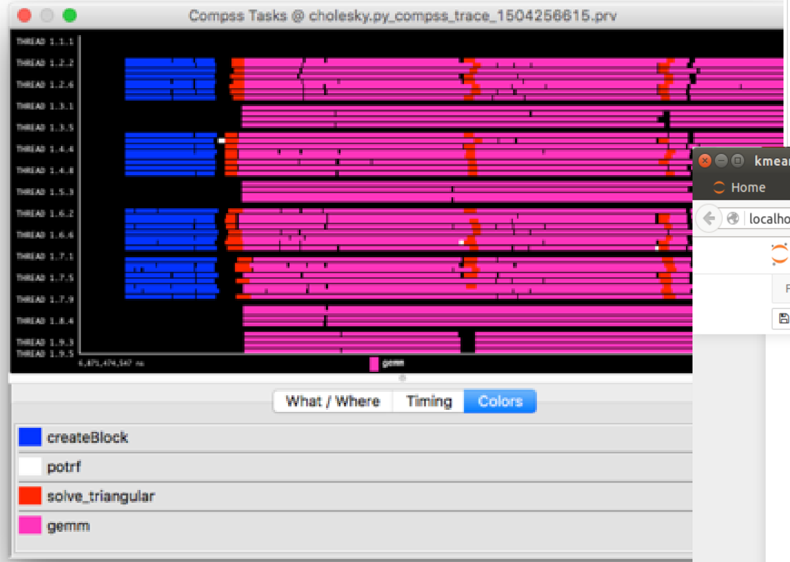
Barcelona Supercomputing Center
Centro Nacional de Supercomputación

PyCOMPSs development environment

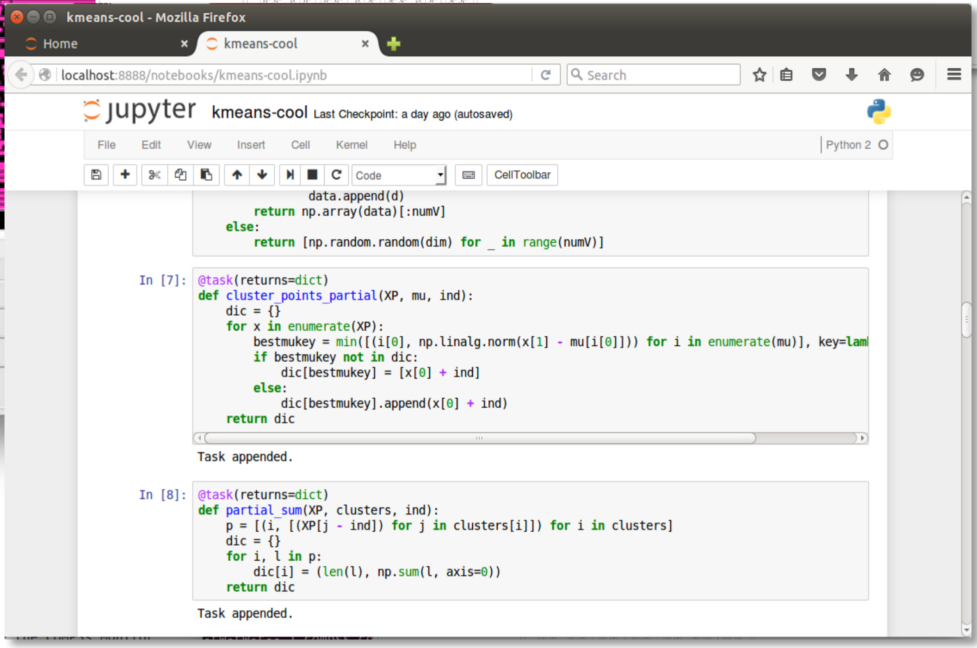
- Runtime monitor
- Paraver traces
- Jupyter-notebooks integration



Color	Name
🟦	cluster_points_partial()In
⬜	partial_sum()InteractiveM
🟥	reduceCentersTask()Inte



🟦	createBlock
⬜	potrf
🟥	solve_triangular
🟪	gemm


kmeans-cool - Mozilla Firefox
Home
kmeans-cool
localhost:8888/notebooks/kmeans-cool.ipynb
jupyter kmeans-cool Last Checkpoint: a day ago (autosaved)
File
Edit
View
Insert
Cell
Kernel
Help
Python 2
Code
CellToolbar

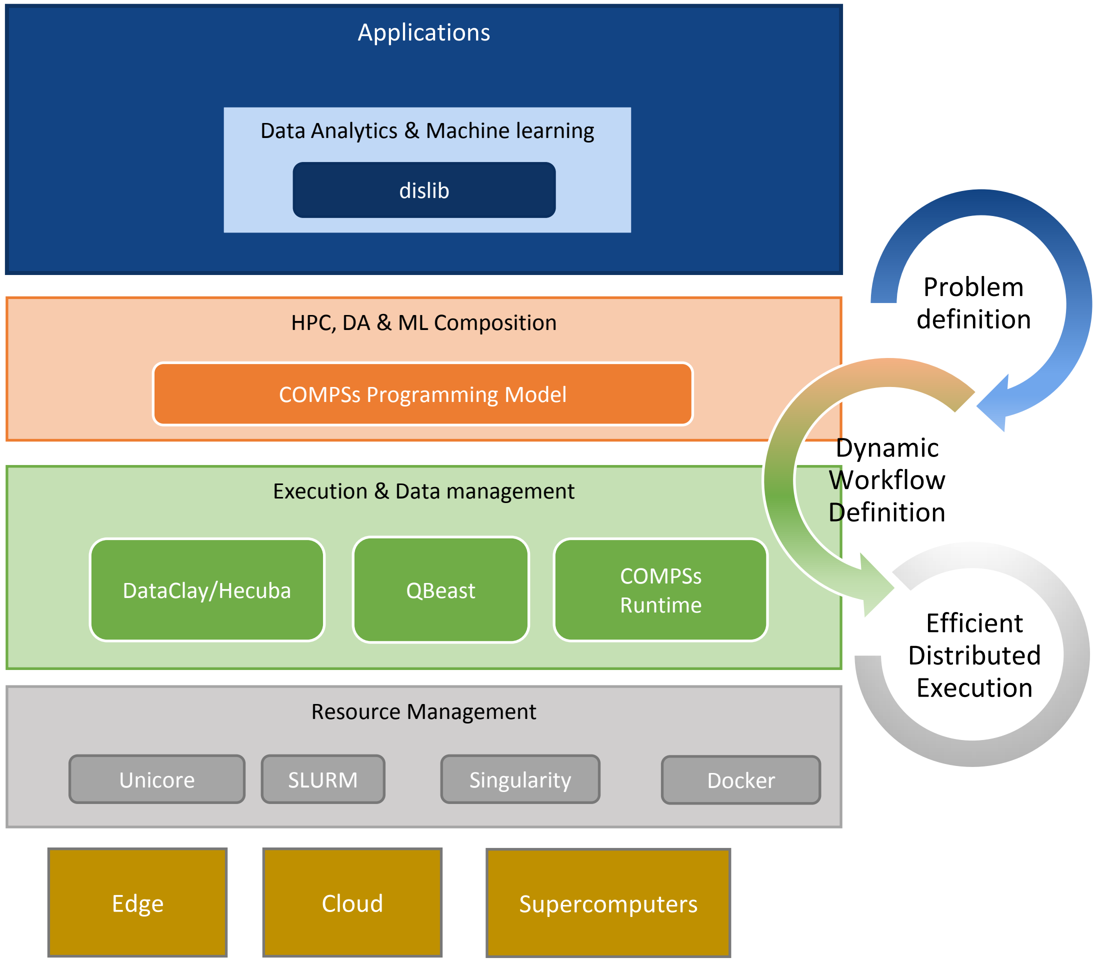
```
            data.append(d)
        return np.array(data)[:numV]
    else:
        return [np.random.random(dim) for _ in range(numV)]
```

```
In [7]: @task(returns=dict)
        def cluster_points_partial(XP, mu, ind):
            dic = {}
            for x in enumerate(XP):
                bestmukey = min([(i[0], np.linalg.norm(x[1] - mu[i[0]])) for i in enumerate(mu)], key=lam
                if bestmukey not in dic:
                    dic[bestmukey] = [x[0] + ind]
                else:
                    dic[bestmukey].append(x[0] + ind)
            return dic

        Task appended.
```

```
In [8]: @task(returns=dict)
        def partial_sum(XP, clusters, ind):
            p = [(i, [(XP[j - ind]) for j in clusters[i]]) for i in clusters]
            dic = {}
            for i, l in p:
                dic[i] = (len(l), np.sum(l, axis=0))
            return dic

        Task appended.
```

Conclusions

- COMPSs provides a workflow environment that enables the integration of HPC simulation and modelling with big data analytics and machine learning
- Support for dynamic workflows that can change their behaviour during the execution
- Support for dynamic resource management depending on the actual workload needs
- Support for data-streaming enabling the combination of task-flow and data-flow in the same workflow
- Support for persistent storage beyond traditional file systems.



Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Projects where COMPSs is used/developed

AI SPRINT
AI IN SECURE PRIVACY-PRESERVING COMPUTING CONTINUUM

CAELESTIS

ICOS

DT-GEO

Per Med CoE

HPC/Exascale Centre of Excellence in Personalised Medicine

MEEP | MareNostrum Experimental Exascale Platform

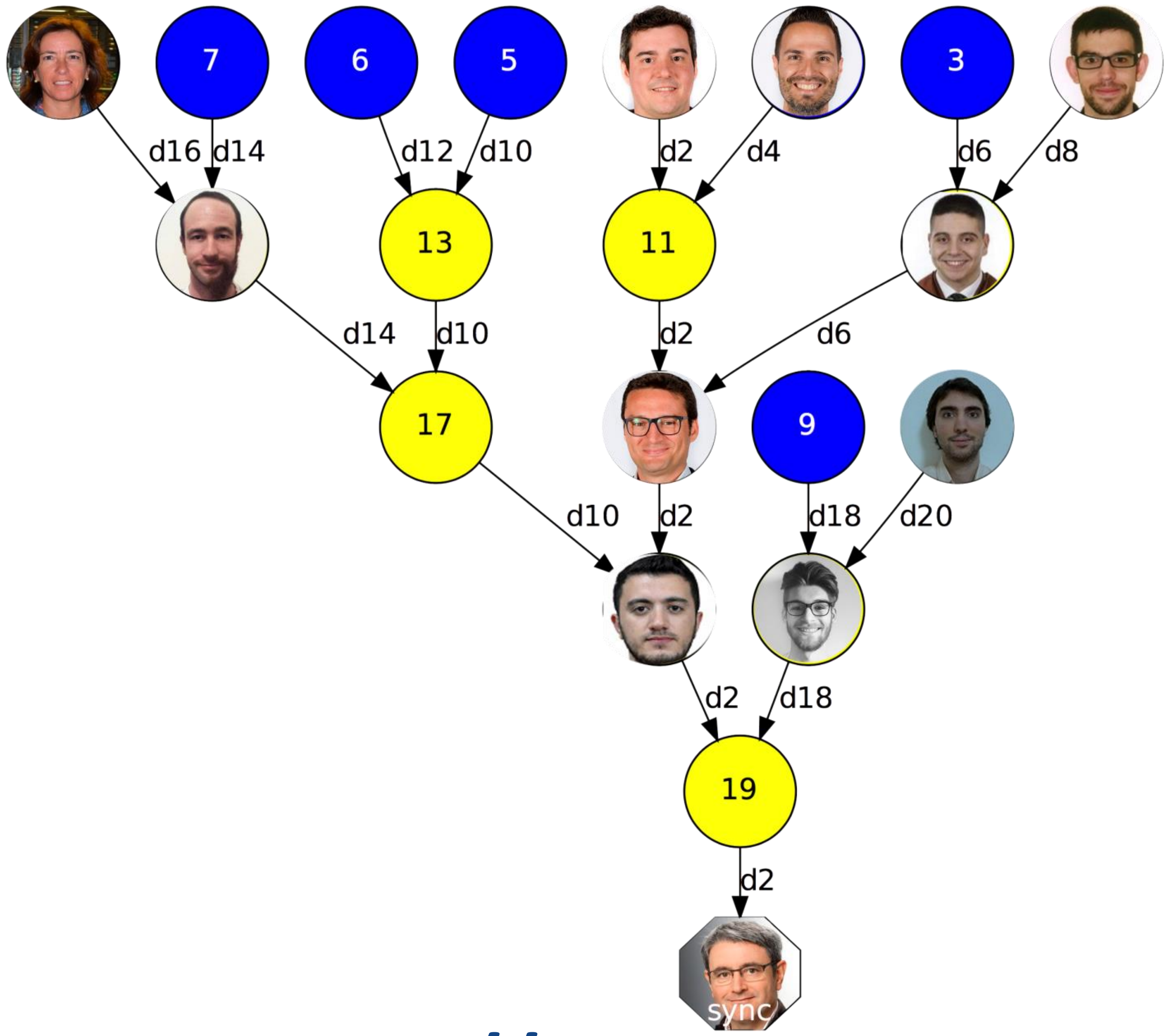
eFlows4HPC

Joint Laboratory for Extreme-Scale Computing

Inria NCSA Argonne Barcelona Supercomputing Center JÜLICH RIKEN

BSC Barcelona Supercomputing Center
Centro Nacional de Supercomputación

The WDC team



http://compss.bsc.es