PROCESS

PROviding Computing solutions for ExaScale ChallengeS

Towards a New Paradigm for Programming Scientific Workflows

Reginald Cushing, Onno Valkering, Adam Belloum, Cees de Laat

24th Sep 2019, eScience conference – BD2DC, San Diego

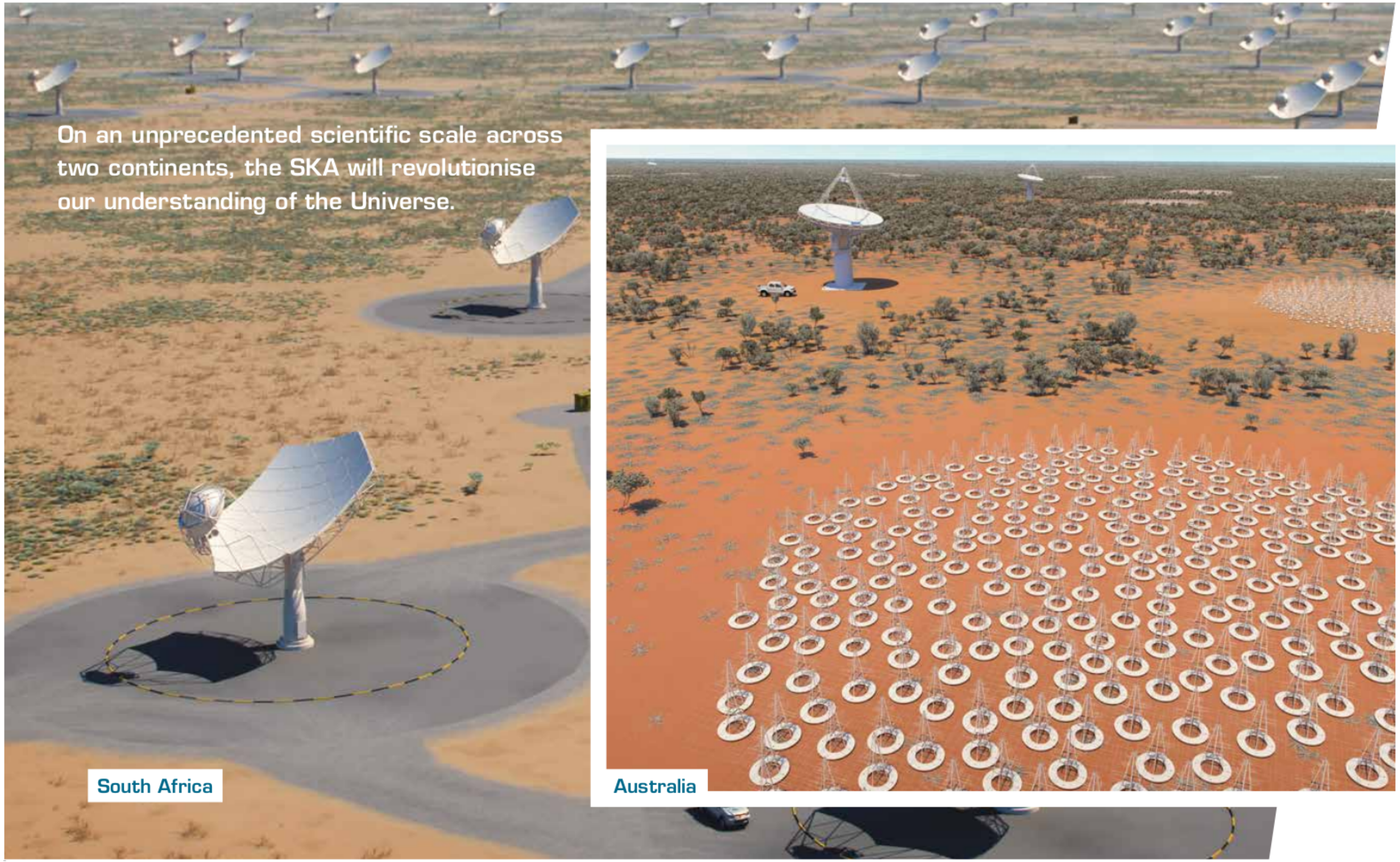
This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 777533.

LOFAR: Low Frequency Array

Images courtesy of: ASTRON

LOFAR radio telescope – is a “distributed software telescope” consisting of ~88.000 antennas in ~51 stations scattered over Europe. It produces up to **1.6 TB/s of raw data**, processed real time to combine the signals of multiple antennas (correlation). **This results in up to 35 TB/h of intermediate data** (visibilities) which is stored for further analysis.

SKA: Square Kilometer Array



Operational in 2022+

130K ~ 1M (LOFAR-style) antenna in Australia + 200 ~ 2000 dishes in South Africa

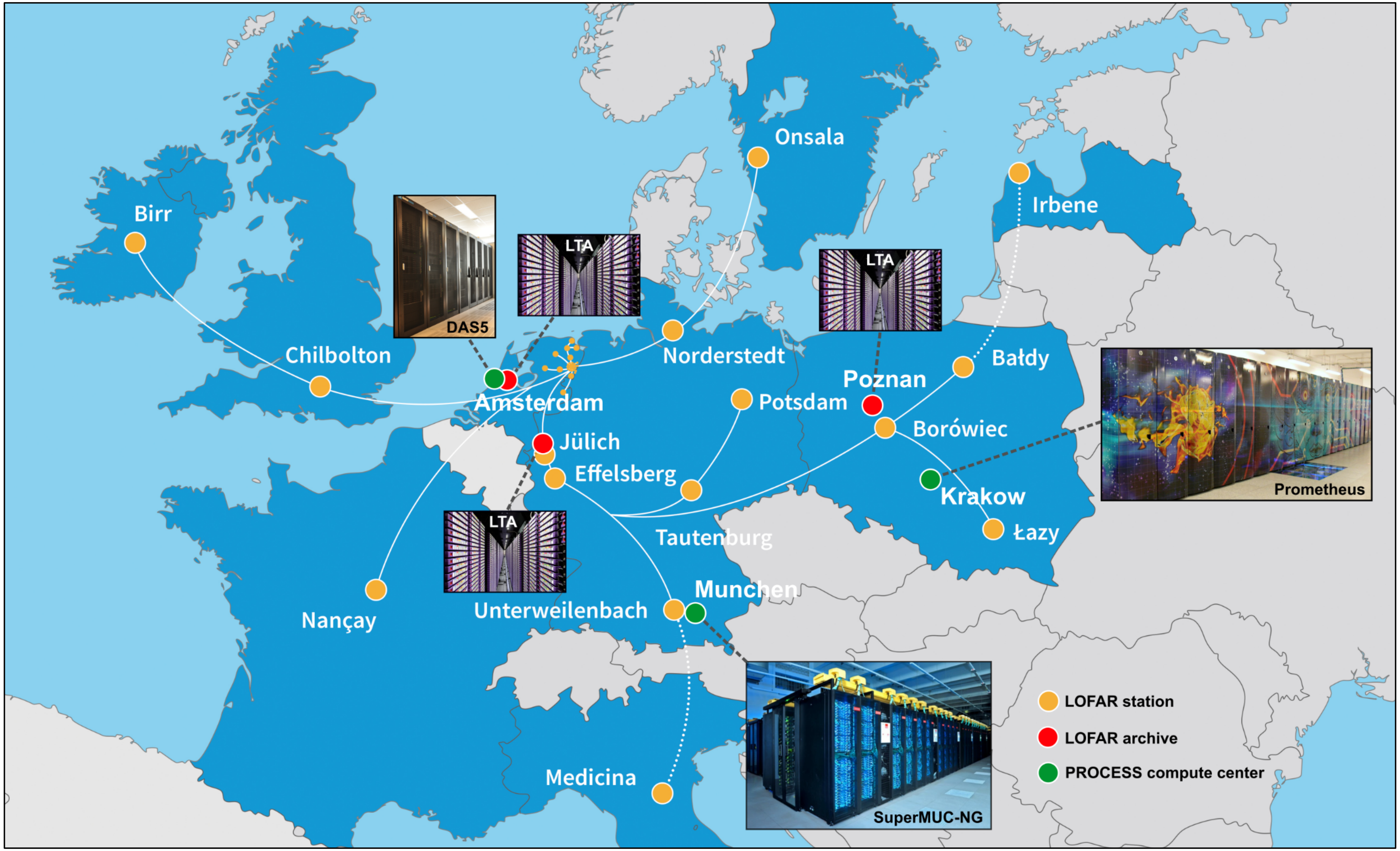
Wider frequency range and higher sensitivity and survey speed than existing telescopes

Zettabytes/year **raw** data volumes

130~300PB/year of correlated data

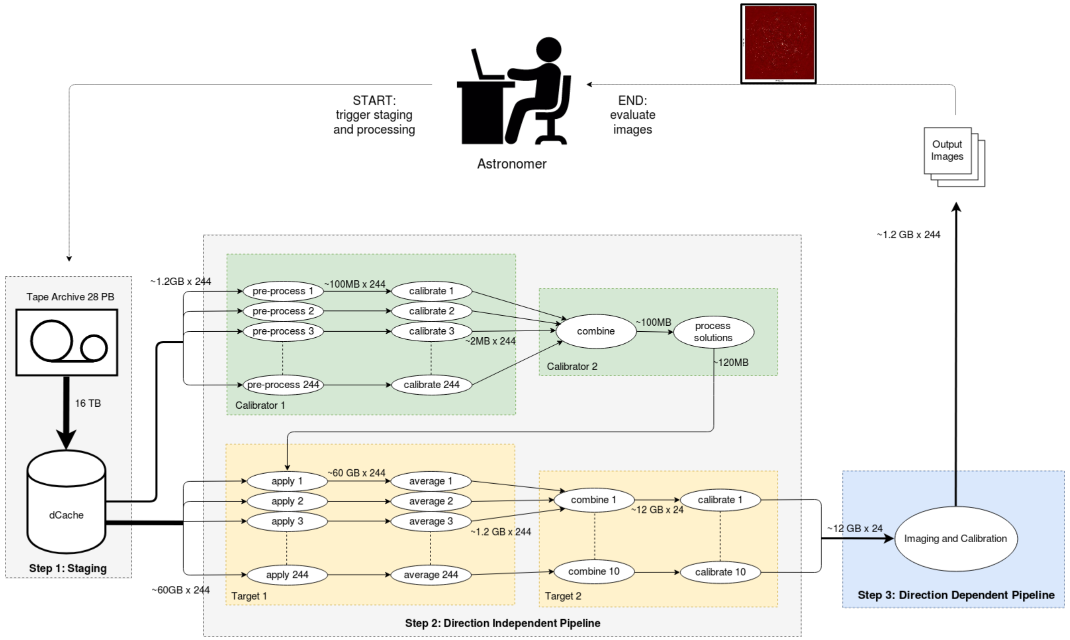
Huge data and processing problem

Overview of hardware locations



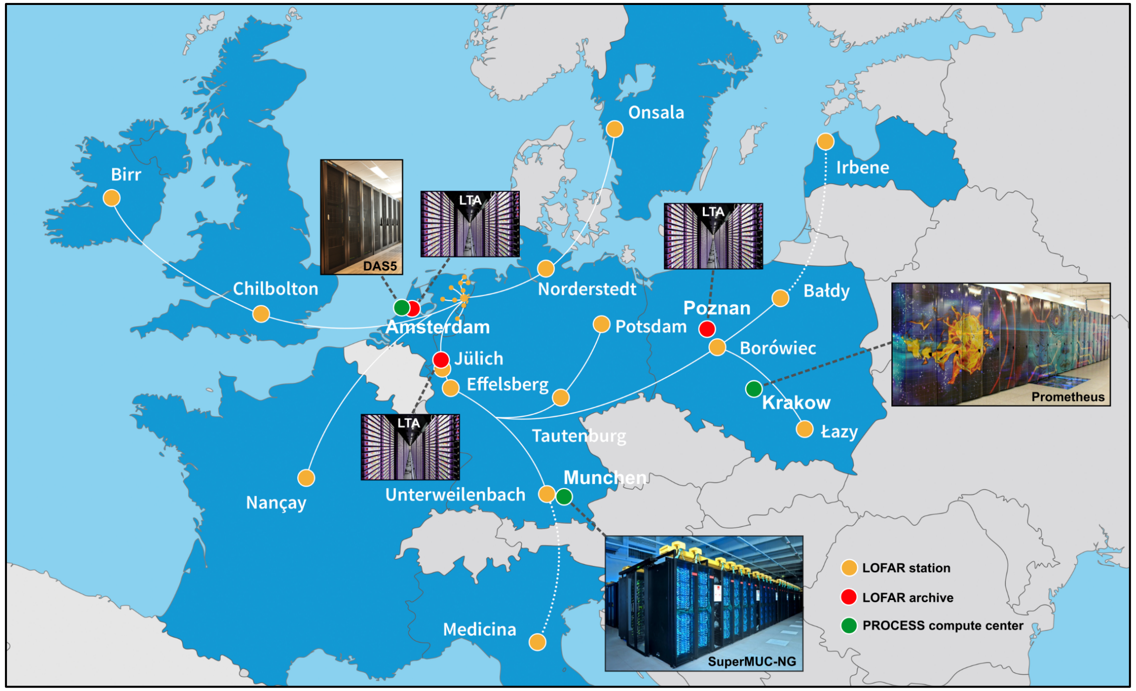
PROCESS Running applications on dispersed resources

Applications



???

Dispersed Resources



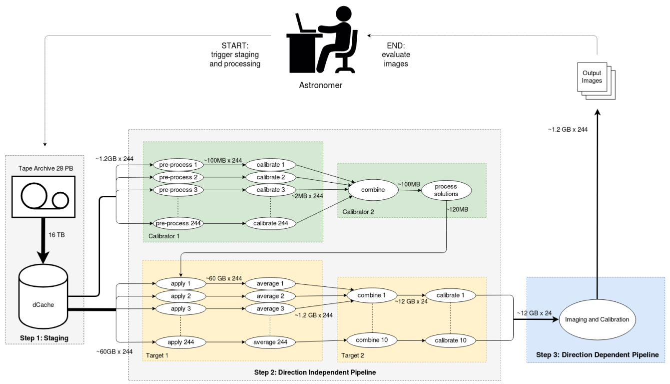
Complex Mapping

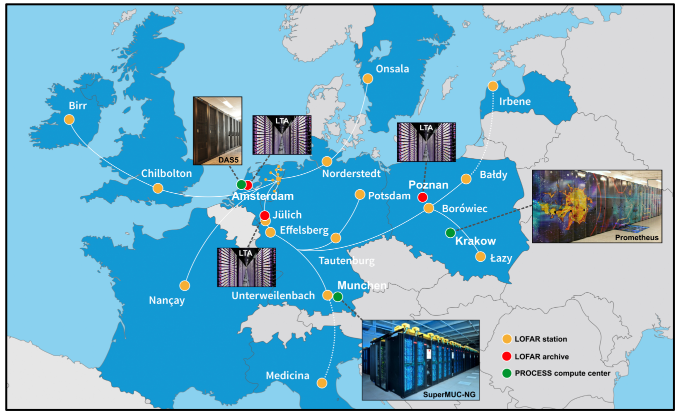
Challenges

- **Application**
  - How to facilitate rapid pipeline development and deployment on heterogenous resources?

- **Infrastructure**
  - How to setup complex infrastructures that can facilitate applications in their requirements?

- **Middleware**
  - How to scale applications from locally tuned clusters to multi-datacenter deployments?

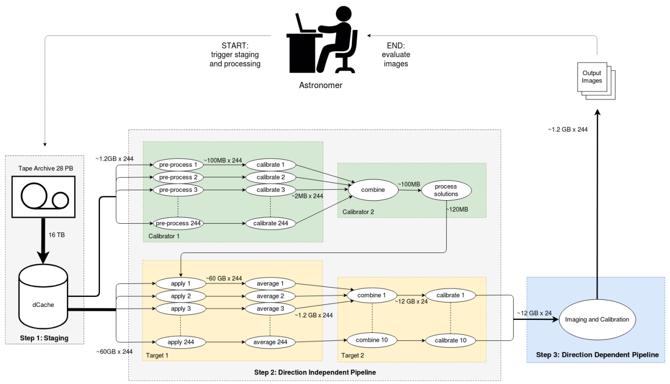


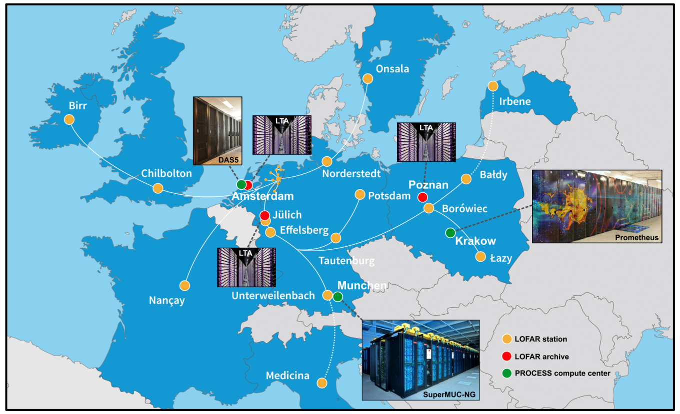


Challenges

- **Data**
  - How to smartly transfer data to make the most of the underlying infrastructure.

- **Security & Privacy**
  - How to create secure scientific pipelines that process sensitive data on distributed infrastructures using dispersed resources?





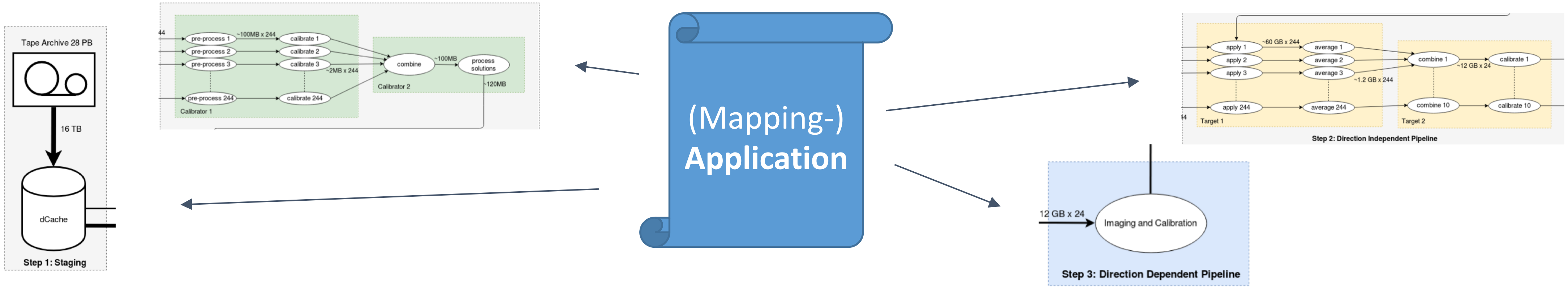
A full-stack approach

- **Challenges cover all layers**
  - In order to address these challenges it is required to control the entire stack, i.e. both applications and networking.
  - A full-stack approach lets us fine-tune the interaction between application and dispersed resources.
- **Workflow Management Systems ?**
  - Traditional workflow systems don't model the underlying infrastructure, and usually target a static deployment.
  - We need a more expressive language than typical flow-charts.

PROCESS Programmable infrastructure

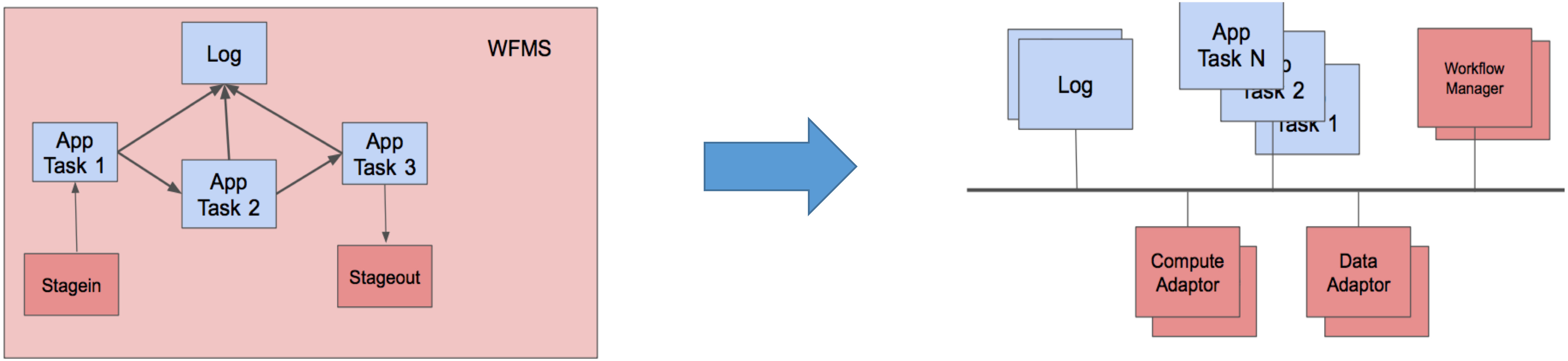
- **Programmability**
  - Higher-level programming languages and DSLs are known to be very expressive, while still being relatively user-friendly.

  - These can be used to describe the complex mapping between applications and resources. Essentially making the turning the (mapping-)application into the orchestrator:



PROCESS Virtualized micro-architecture

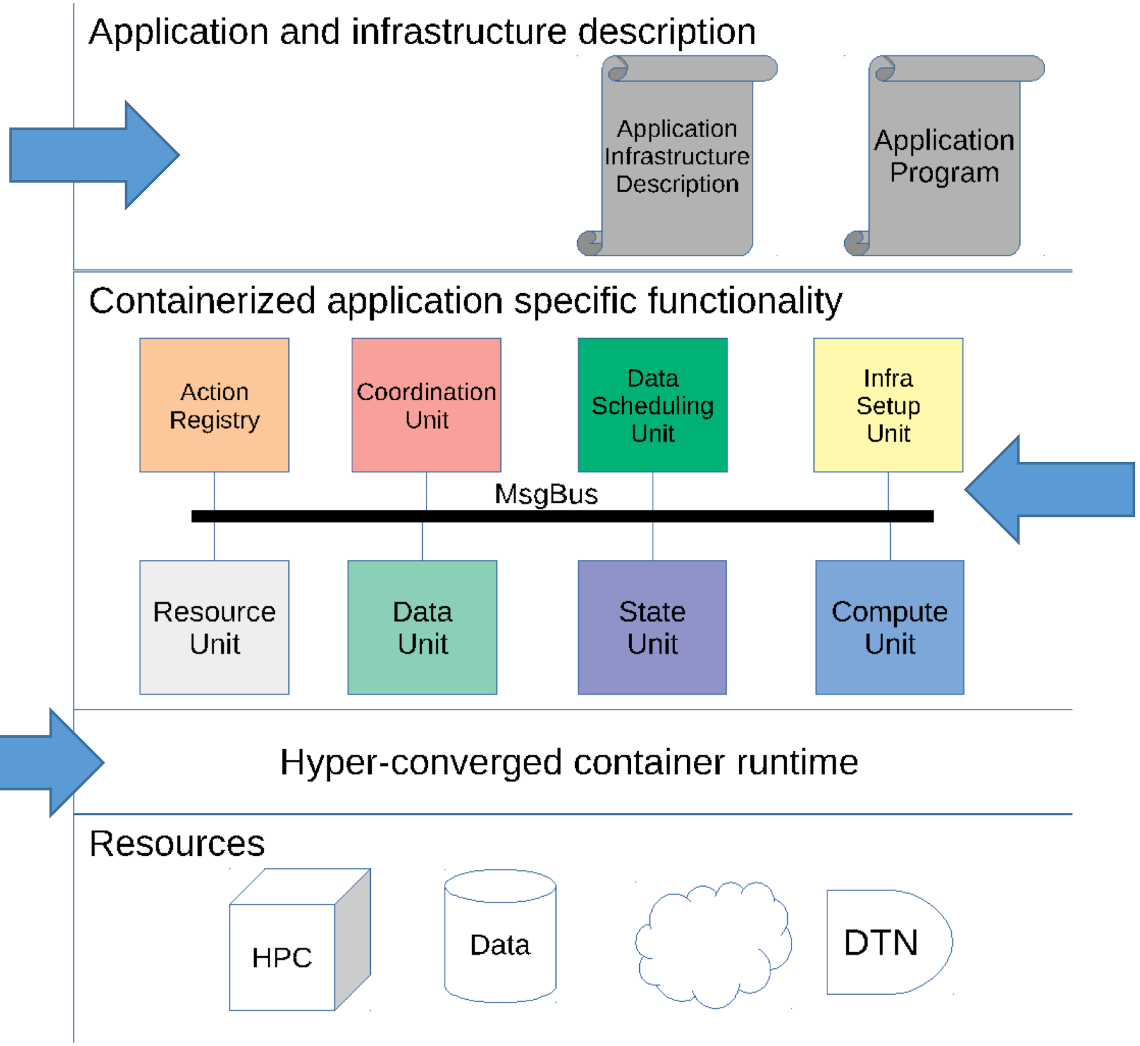
- Move towards a flat and container-centric micro-architecture.



- **Adaptable** through a hyper-converged container runtime.
- **Dynamic** through programmable application orchestration.
- **Efficient** through on-demand provisioning and elastic scaling.

PROCESS Reference architecture

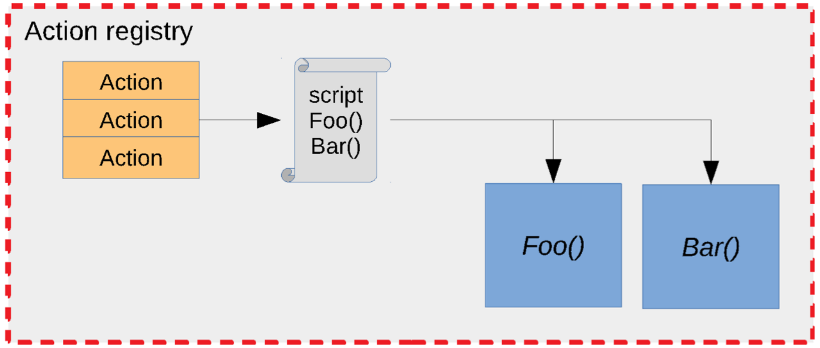
Applications are leading for what components will be provisioned.



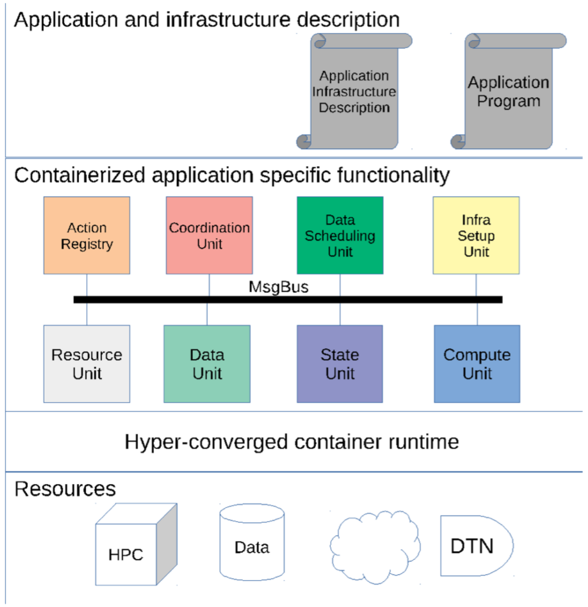
Choreography is effectuated though an **reactive** and **event-driven** (mapping)application.

Container runtime abstracts away from **heterogeneous** and **dispersed** resources.

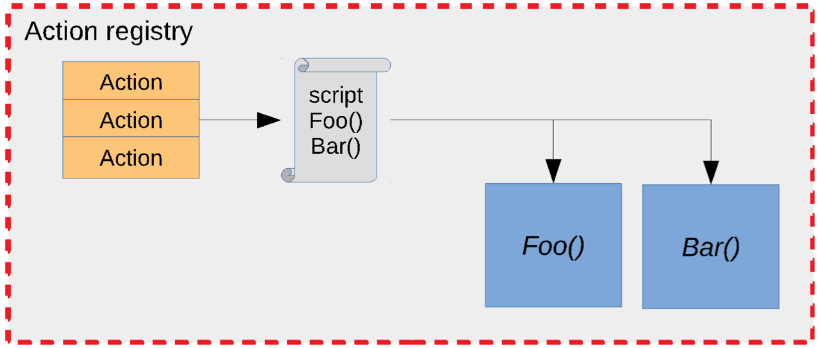
- **Containerized functions**
  - Encapsulates arbitrary compute routines
  - Docker for Cloud, Singularity for HPCs
  - Specify the functions that are exposed by the services or compute routine.



- **Registry of functions**
  - Keeps track of the available functions
  - Used for discovery and documentation
  - Composable building blocks

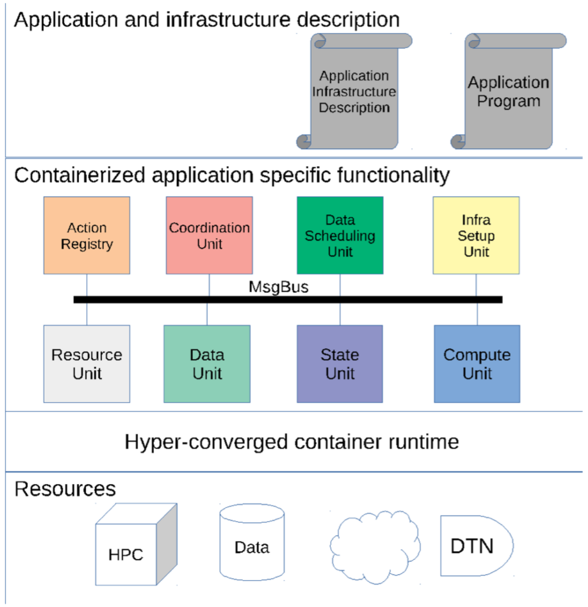


- **Coordination scripts**
  - Hierarchical compute routines
  - Using an existing/familiar language
  - Registered in the registry



  - **Import containerized functions**
    - Generated stubs are used to represent containerized functions inside the IDE.

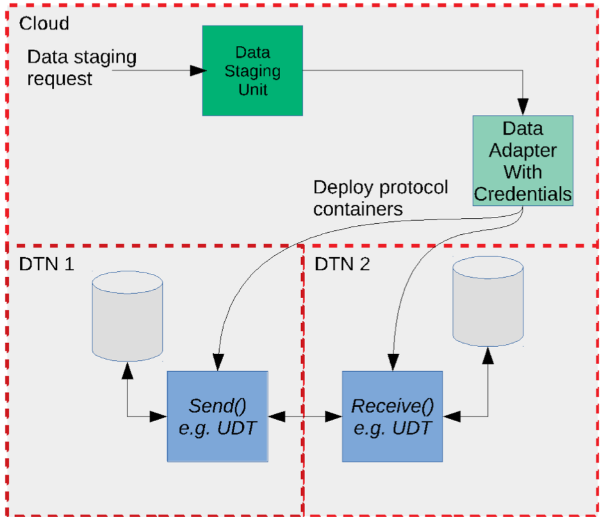
```
import { stage } from './lofar-lta';

stage(2432018)
  stage(observation: number): string[]

  The ID of the observation
  This function stages an observation at the LOFAR LTA.
  @return — An array of SURLs of the staged files.
```

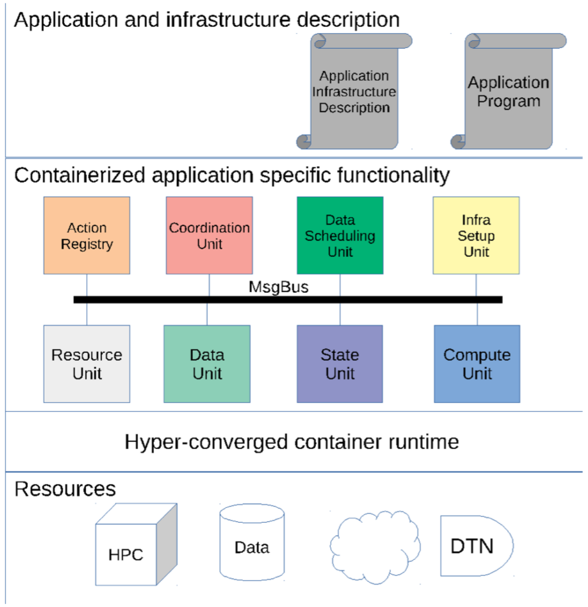


Programming with containers

- **Coordinator / Infrastructure Units**
  - Starts the application by generating initial events.
  - Contributes the event-loop to process events.
  - Resolves events to function-execution invocations.

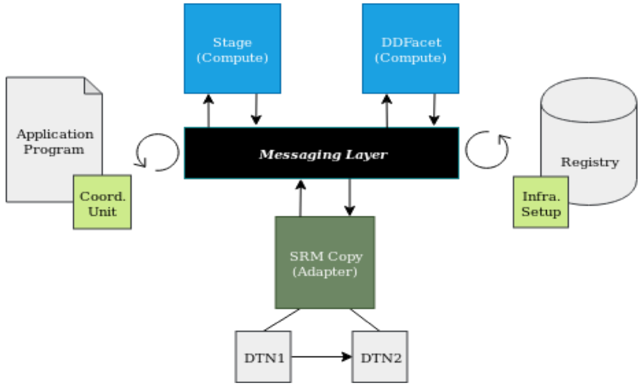
- **Data Unit**
  - Resource access is abstracted through adapters containers.
  - Transfer between dispersed locations using DTNs.





PROCESS Conclusions

- Challenges of future scientific applications stem from all layers of the stack. Programmable micro-infrastructures allow us to address all of these challenges.
- Outlined the basic components that are needed to create isolated, portable and scalable micro-infrastructures.
- **Future work**
  - Implementing a proof of concept based on the reference architecture.
  - Open-source and available on GH: https://github.com/brane-ri



PROCESS

PROviding Computing solutions for ExaScale ChallengeS

Towards a New Paradigm for Programming Scientific Workflows

Reginald Cushing, Onno Valkering, Adam Belloum, Cees de Laat

24th Sep 2019, eScience conference – BD2DC, San Diego

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 777533.

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN · Universiteit van Amsterdam · netherlands eScience center · Hes·so · Lufthansa Systems · inmark europa · Ústav informatiky Slovenská akadémia vied · AGH · CYFRONET