Overview and Innovation

Project coordinator: Jean-Thomas, Acquaviva, DDN, jtacquaviva@ddn.com

Scientific director: Angelos Bilas, FORTH, bilas@ics.forth.gr

Contact Info: info@evolve-h2020.eu

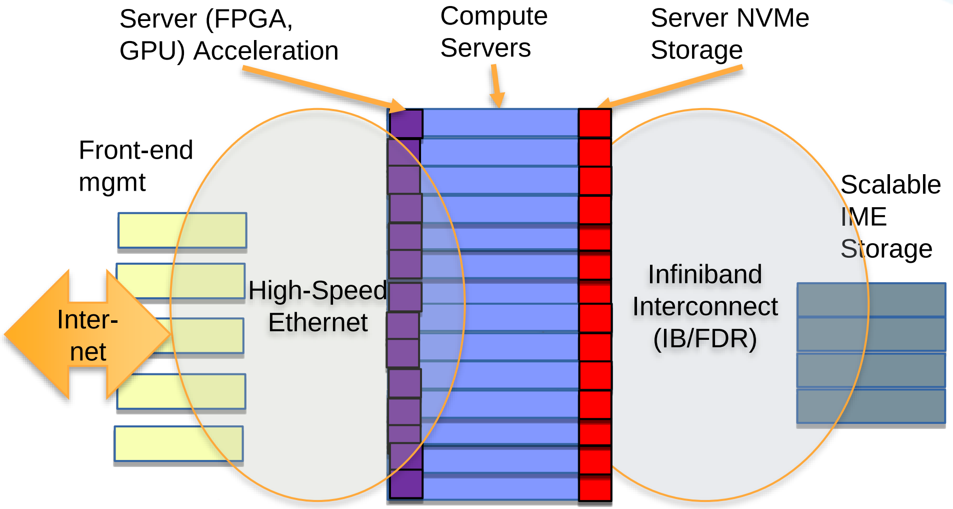
www.evolve-h2020.eu

May 26, 2021

Evolve: At the Crossroads of BD-Cloud-HPC

- Goal and scope of Evolve
  - Process “large” datasets
  - Do so “interactively” for user-facing applications
- 1. BD → Productivity
  - Decouple platform from users
- 2. Cloud → Resource cost
  - Consolidation, resource provisioning, shared use of resources
- 3. HPC → Performance
  - Vertical integration, storage, interconnects, accelerators

HPC Resources → Evolve Heterogeneous HW Platform

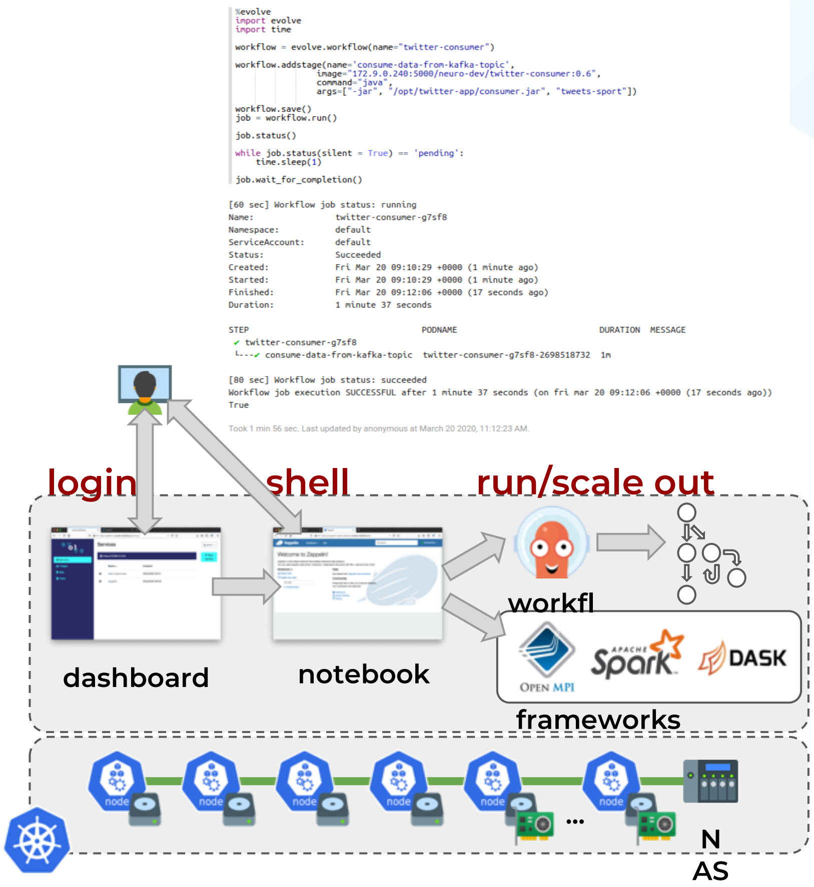


Outline

- Front-end: Decouple users from the platform
- Back-end: Resource use and efficiency
- Summary and Reflections

Productivity → Evolve Dashboard

- Front-end for all users
- Offers secure and private user management
    - Isolated namespaces
- Provides notebooks as a main abstraction
    - Templates + Programmatic API
- Integrates Evolve and third-party services
    - Container-related management
- Significant integration effort



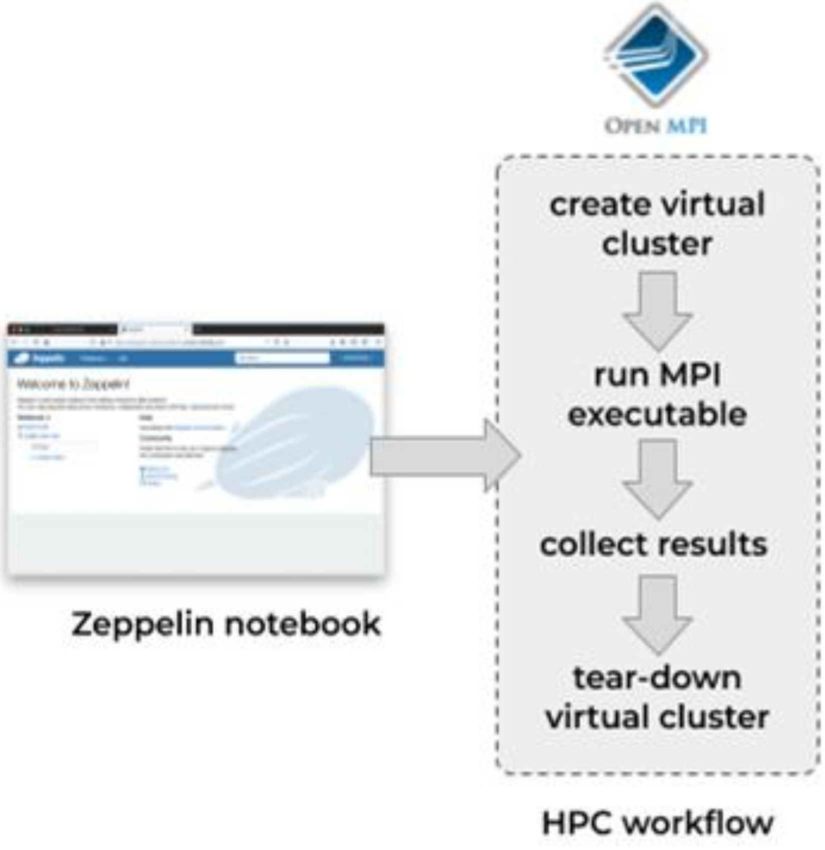
```
%evolve
import evolve
import time

workflow = evolve.workflow(name="twitter-consumer")

workflow.addstage(name='consume-data-from-kafka-topic',
                  image="172.9.0.240:5000/neuro-dev/twitter-consumer:0.6",
                  command="java",
                  args=["-jar", "/opt/twitter-app/consumer.jar", "tweets-sport"])

workflow.save()
job = workflow.run()

job.status()

while job.status(silent = True) == 'pending':
    time.sleep(1)

job.wait_for_completion()
```

```
[60 sec] Workflow job status: running
Name:                 twitter-consumer-g7sf8
Namespace:            default
ServiceAccount:       default
Status:               Succeeded
Created:              Fri Mar 20 09:10:29 +0000 (1 minute ago)
Started:              Fri Mar 20 09:10:29 +0000 (1 minute ago)
Finished:             Fri Mar 20 09:12:06 +0000 (17 seconds ago)
Duration:             1 minute 37 seconds

STEP                                   PODNAME                                DURATION  MESSAGE
 ✔ twitter-consumer-g7sf8
 └---✔ consume-data-from-kafka-topic  twitter-consumer-g7sf8-2698518732  1m

[80 sec] Workflow job status: succeeded
Workflow job execution SUCCESSFUL after 1 minute 37 seconds (on fri mar 20 09:12:06 +0000 (17 seconds ago))
True
```

Took 1 min 56 sec. Last updated by anonymous at March 20 2020, 11:12:23 AM.

Productivity → Evolve Workflows

- Workflows are end-to-end synthetic descriptions of computation
- Workflows are important
  - This is what users care about
  - More room to optimize execution
- Figurative Evolve workflow
  - d1 = open(“dataset”)
  - d2 = spark.clean(d1, params)
  - d3 = tensorFlow.train(d2, params)
  - d4 = mpi.simulate(d3, d2)
  - d5 = spark.map(d1, f())
  - result = viz(d4, d5, params)



Zeppelin notebook

create virtual cluster

run MPI executable

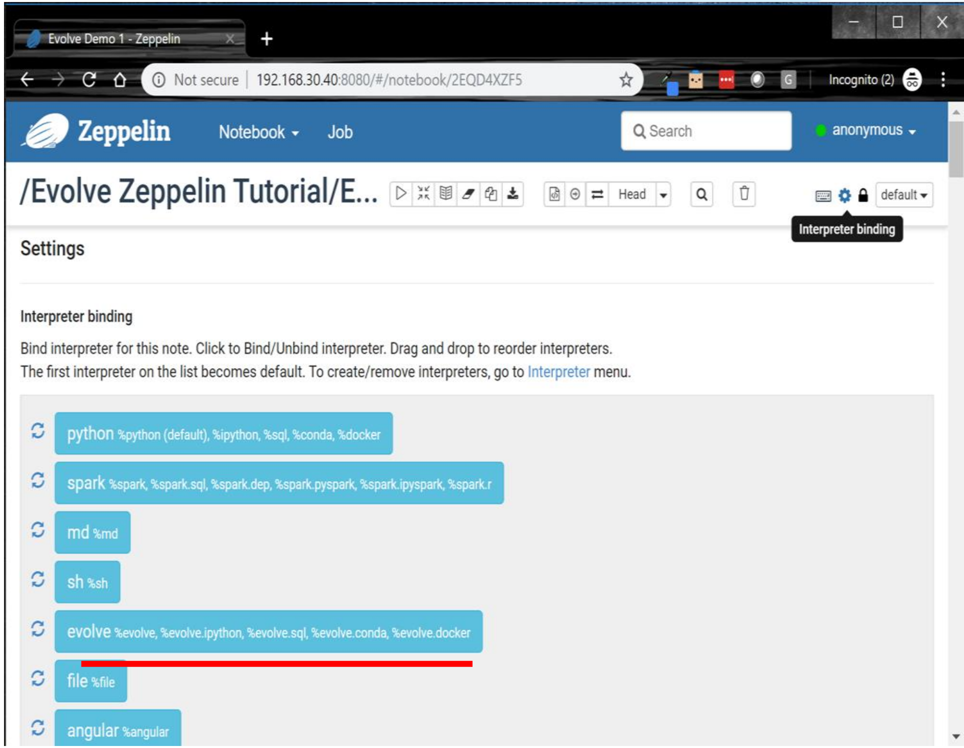
collect results

tear-down virtual cluster

HPC workflow

Custom Evolve Interpreter for Workflows

- Simplified generation of multi-stage workflows for Argo engine
- Support for control state of the K8s cluster
- Manage container images in docker repository
- Manage data files via Python or shell interpreter



192.168.30.40:8080/#/notebook/2EQD4XZF5

/Evolve Zeppelin Tutorial/E...

Settings

Interpreter binding

Bind interpreter for this note. Click to Bind/Unbind interpreter. Drag and drop to reorder interpreters.
The first interpreter on the list becomes default. To create/remove interpreters, go to Interpreter menu.

python %python (default), %ipython, %sql, %conda, %docker

spark %spark, %spark.sql, %spark.dep, %spark.pyspark, %spark.ipyspark, %spark.r

md %md

sh %sh

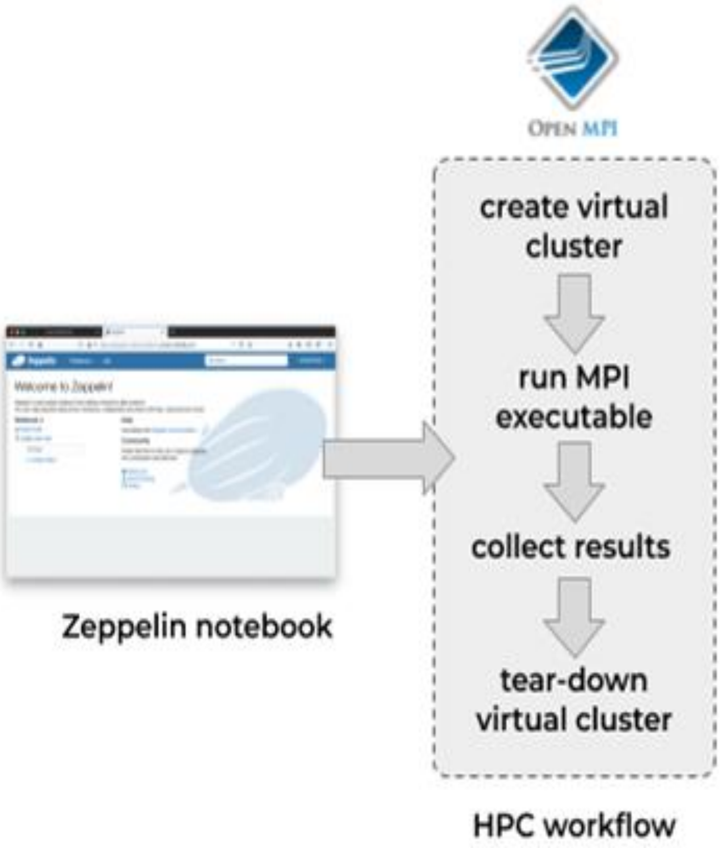
evolve %evolve, %evolve.ipython, %evolve.sql, %evolve.conda, %evolve.docker

file %file

angular %angular

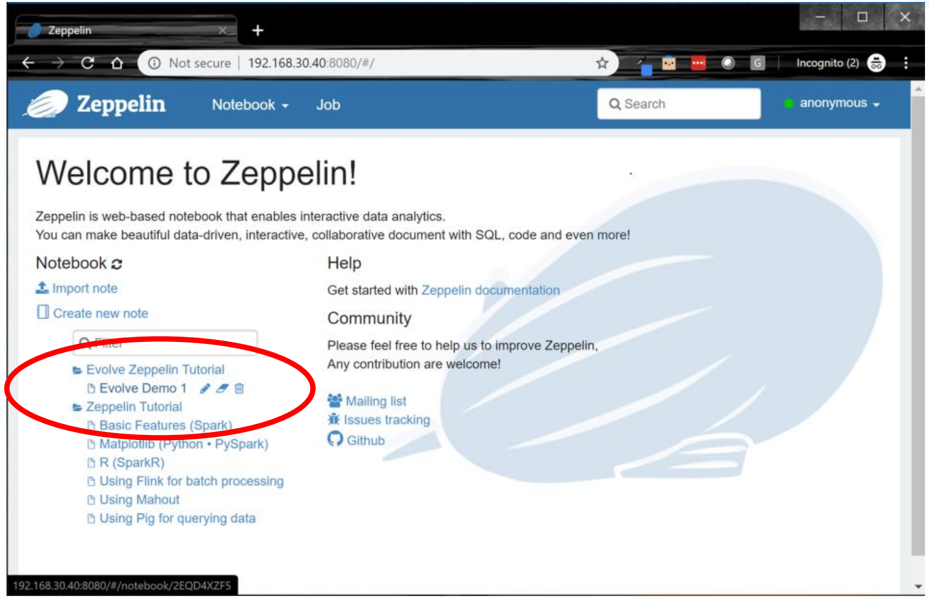
Integration of HPC Computation and Big-Data Frameworks

- Workflows can include HPC/MPI stages
- MPI stages can be invoked interactively from notebooks as well
- MPI tasks make use of containerized GPU, Infiniband support
- Evolve automatically creates containerized MPI “virtual clusters”
- Executes MPI under Slurm / K8s over shared resources with BD tasks



User support: Preconfigured workflows

- Dashboard Zeppelin service provides demo workflows to help users and Evolve use-case providers
- Zeppelin Tutorial folder contains Demo notebooks for workflows
- Available demo workflows:
  - Create a simple workflow
  - Create a workflow that includes a step with a custom python script
  - Create a workflow with a conditional step
  - Create two workflows with sensors
- “CookBooks” with instructions available as well



Welcome to Zeppelin!

Zeppelin is web-based notebook that enables interactive data analytics.
You can make beautiful data-driven, interactive, collaborative document with SQL, code and even more!

Notebook

- Import note
- Create new note
- Evolve Zeppelin Tutorial
  - Evolve Demo 1
- Zeppelin Tutorial
  - Basic Features (Spark)
  - Matplotlib (Python • PySpark)
  - R (SparkR)
  - Using Flink for batch processing
  - Using Mahout
  - Using Pig for querying data

Help

Get started with Zeppelin documentation

Community

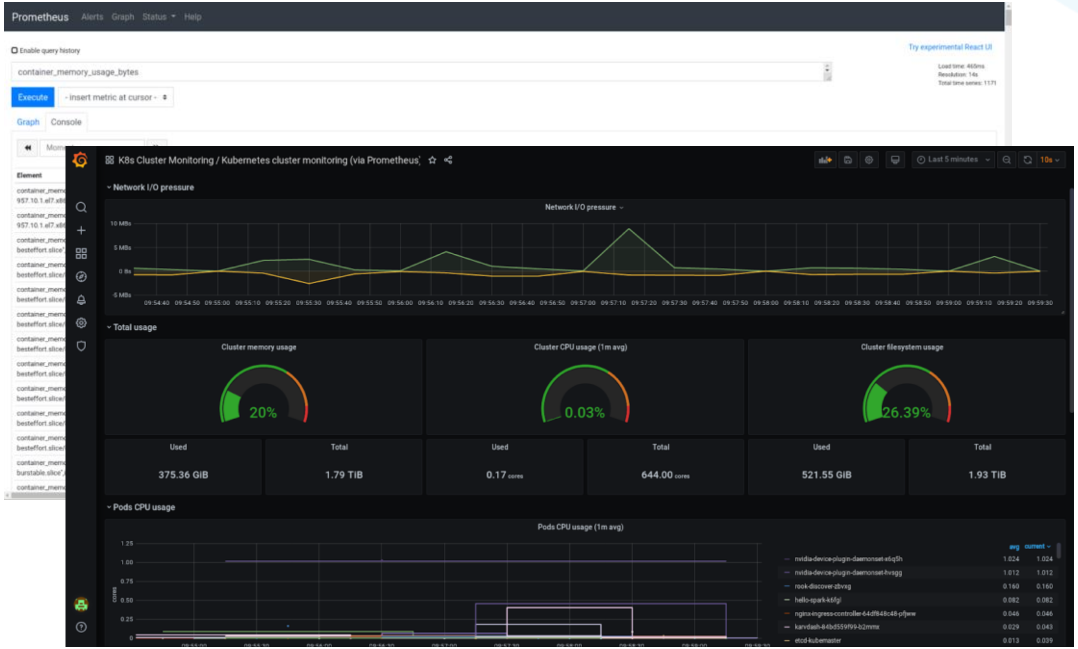
Please feel free to help us to improve Zeppelin,
Any contribution are welcome!

- Mailing list
- Issues tracking
- Github

192.168.30.40:8080/#/notebook/2EQD4XZF5

“Vertical” Monitoring

- Hardware resources
- Software stack
- Storage



container_memory_usage_bytes

Load time: 465ms
Resolution: 14s
Total time series: 1171

K8s Cluster Monitoring / Kubernetes cluster monitoring (via Prometheus)

	Used	Total
Cluster memory usage (20%)	375.36 GiB	1.79 TiB
Cluster CPU usage (1m avg) (0.03%)	0.17 cores	644.00 cores
Cluster filesystem usage (26.39%)	521.55 GiB	1.93 TiB

	avg	current
nvidia-device-plugin-daemonset-x6q5h	1.024	1.024
nvidia-device-plugin-daemonset-hvsgg	1.012	1.012
rook-discover-zbvxg	0.160	0.160
hello-spark-k6fgl	0.082	0.082
nginx-ingress-controller-64df848c48-pfjww	0.046	0.046
karvdash-84bd559f99-b2mmx	0.029	0.043
etcd-kubemaster	0.013	0.039

Outline

- Front-end: Decouple users from the platform
- Back-end: Resource use and efficiency
- Summary and Reflections

Overview

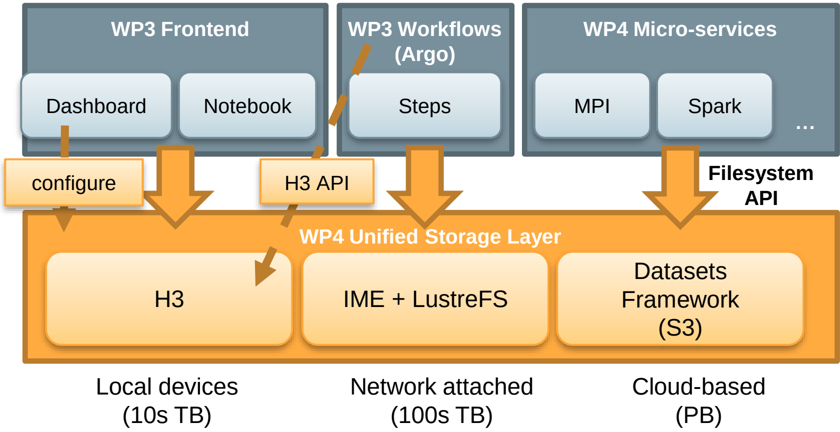
1. **End-to-end integration of all micro-services with the HPC platform**
    - MPI, Kafka, Spark Structured Streaming, Apache Spark, TensorFlow/keras, Visualization
2. **Enhanced storage features**
    - Data access path → H3 (FORTH)
    - Dataset abstraction for lifecycle management → DLF (IBM)
2. **Resource management**
    - Resource adaptation → Skynet/Genisys resource allocators
    - Locality based scheduling → DLF colocation of tasks and cache pods

Resource Efficiency → Evolve Unified Storage Layer

- Unified storage layer: from data to datasets
- Workflow stages communicate via storage
  - Practically unlimited address space
- DLF: Data Lifecycle Framework
  - A dataset resolution service
  - Unified names for data as datasets
  - Visible everywhere, throughout task lifetime
  - Handles remote S3 datasets as well
- H3 data access layer
  - High-Volume Highly-Available High-Throughput (H3) service – Compatibility to S3
  - Unified access method, independent of devices
  - Transient workflow results go to fast local drives
  - Persistent end-results go to LustreFS through IME

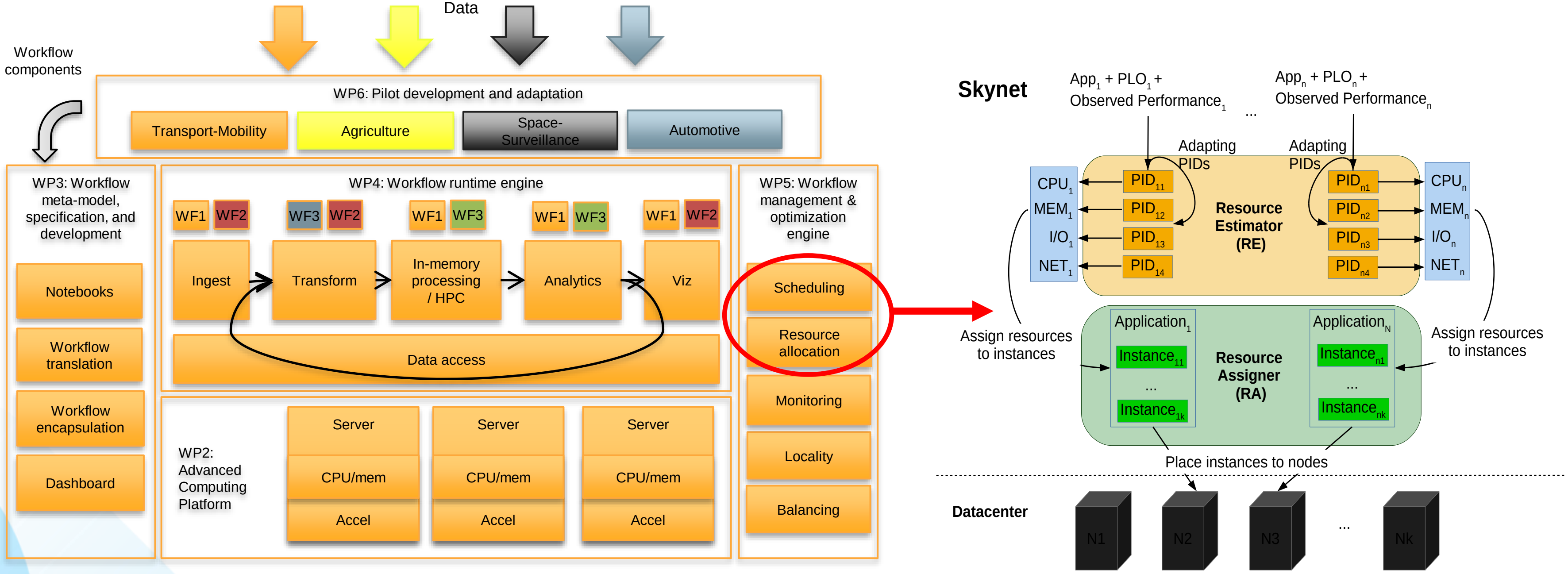
Workload Annotations

```
kind: Job
metadata

  labels:
    dataset.0.id: "my-dataset"
    dataset.0.useas: "mount"
...
```



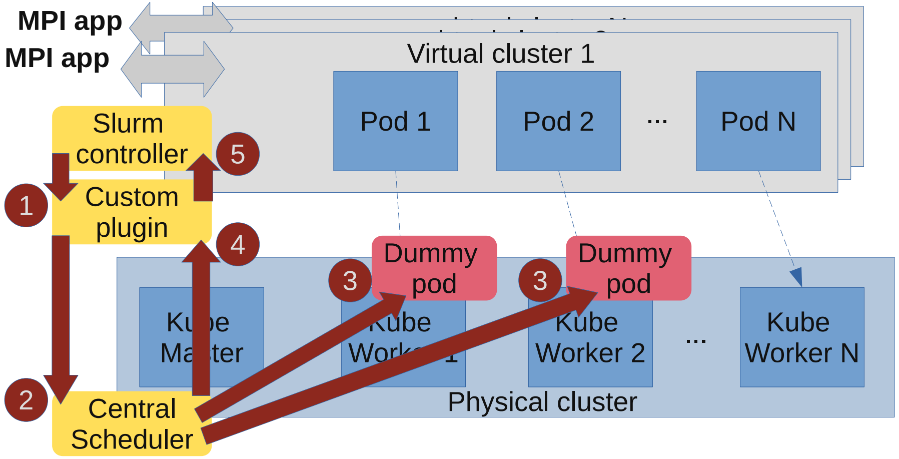
Resource Allocation

- Micro-services + Containers + Kubernetes
- Resource allocation and scheduling policies for Slurm as Kubernetes slave

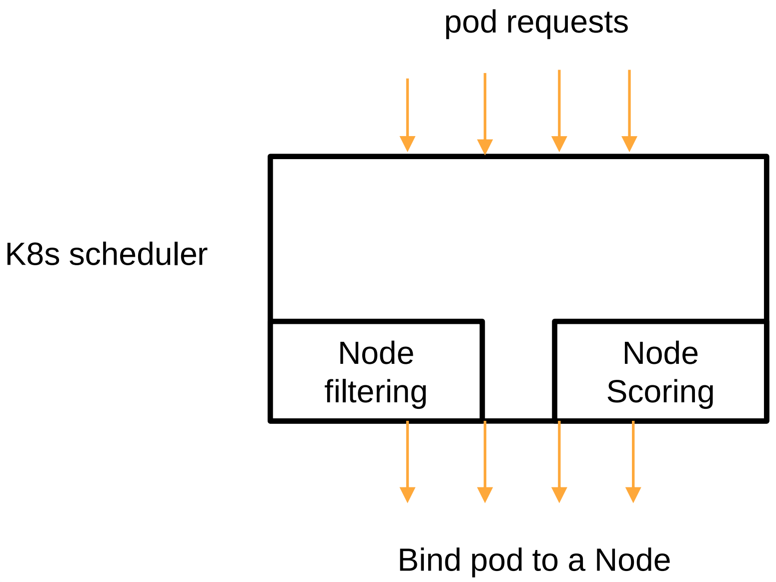


Colocation of Data Processing and MPI Tasks

- Kubernetes - Slurm co-operation
- Two possibilities
  - Slurm master
  - K8s master
- Evolve → K8s master
  - Run full MPI jobs with Slurm commands for strong compatibility
  - Slurm scripts are issued to K8s
  - K8s coordinates with Slurm for resource allocation
  - Slurm executes job script
- Evolve scheduler adjusts resources dynamically for BD tasks



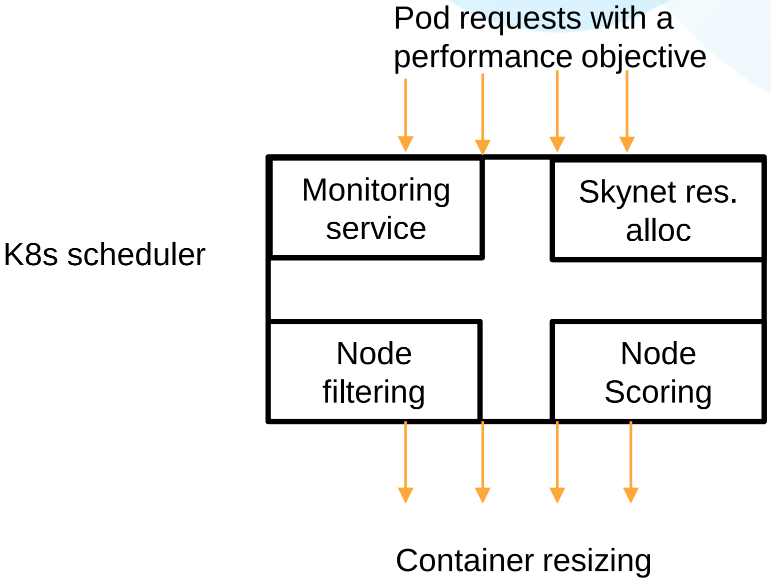
Automatic Resource Allocation on K8s



- Opportunity
  - Big Data workloads are elastic
- Difficulties
  - It needs to satisfy target performance
  - More tasks can fit in a server => interference problem
- In literature: ML either on profile runs or on historic traces
  - A lot of time spent for profiling runs
  - Programs constantly change performance behavior
  - Historic traces: Only 70% of workloads are recurrent (source: Morpheus paper from MSR)

Evolve Adjusts Resources at Runtime

- Implemented as a Kubernetes scheduler
- It expects users to enter a target performance
    - Today users ask for resources instead
- It monitors performance, adjusts resources to meet performance targets
- Adjustments based on a PID controller
    - Online model of performance for each application
    - Model provides benefits for each application with more or less resources
    - Considers multiple resource types (CPU, mem, I/O)
- Mixed HPC-BD workload runs on actual platform



Pod requests with a performance objective

K8s scheduler

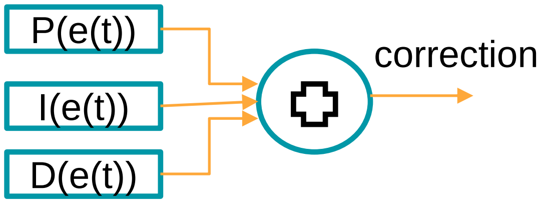
Monitoring service

Skynet res. alloc

Node filtering

Node Scoring

Container resizing



P(e(t))

I(e(t))

D(e(t))

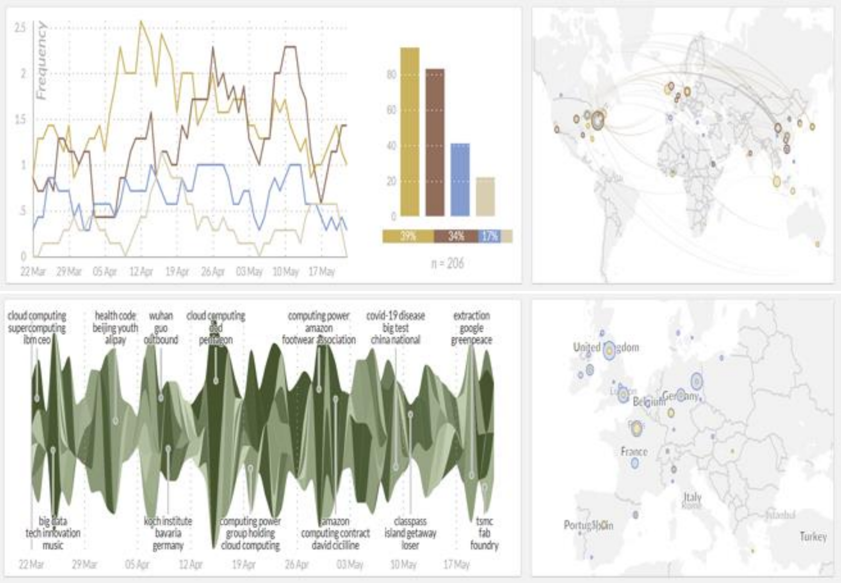
correction

Outline

- Front-end: Decouple users from the platform
- Back-end: Resource use and efficiency
- Summary and Reflections

Broad Applicability of Software Stack → Uses Cases in Multiple Domains

- Evolve workflows cover 7 use cases
  - **Automotive**
    - Engine Testing
    - Predictive maintenance
  - **Mobility**
    - Car-ride scheduling
  - **Earth Observation**
    - Change Detection
    - Maritime Surveillance
    - Agriculture
  - **Public Transportation**
    - Bus monitoring and optimization
- + 15 Proofs of Concept (PoCs)



Summary and Reflections

Road stoppers of existing technologies

- BD processing is slow
  - Data analytics frameworks are heavy
  - As data grows this is not sustainable
- HPC is not easy to use
  - Not enough flexibility due to vertical customization
  - Not all stages of HPC-based workflows are performance critical

In Evolve we have seen

- Performance boost to BD
  - >3x due to better storage + networking (for I/O critical stages)
  - >10x due to multi-core processing for compute-heavy tasks
- Productivity boost to HPC+BD hybrid pipelines
  - 10x-30x fewer lines of code (through the use of BD ecosystem)
  - More than 10x fewer lines of code (due to simpler workflow submission)

Evolve Software Stack

- Most components already open source
- Available as a "Cloud" version
  - No HPC hardware support
- Software stack being examined for deployment in specific applications
  - e.g. sat image processing
- Possibility for access to Evolve HW + SW platform at ATOS for third parties
- Going forward → Integration with the Edge is a main challenge
  - Transparency → Workflow management
  - Efficiency → Acceleration
  - Latency → Distributed state
  - Isolation & Protection → Multiple administrative domains
- **Try out the Evolve platform!**

Thank you !

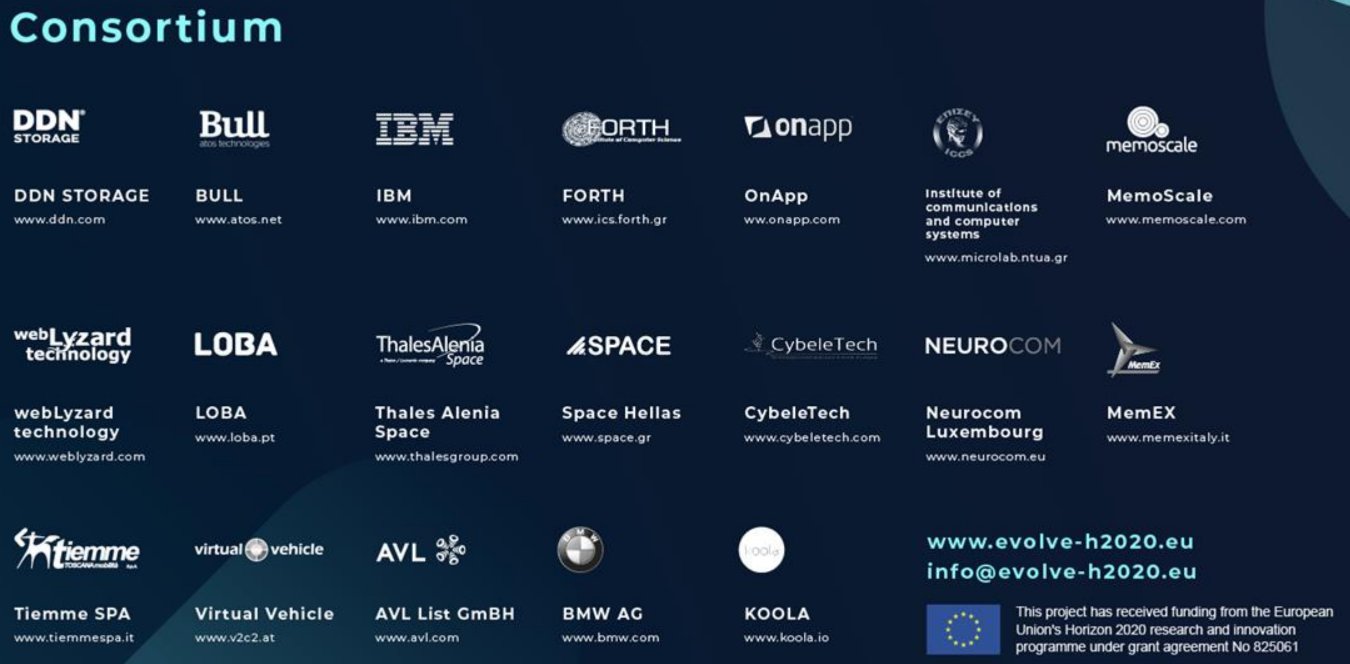
Questions: Now or via email

Angelos Bilas <bilas@ics.forth.gr>

EVOLVE Contact Info: info@evolve-h2020.eu

Project coordinator: Jean-Thomas, Acquaviva, DDN, jtacquaviva@ddn.com

Scientific director: Angelos Bilas, FORTH, bilas@ics.forth.gr



Consortium

DDN STORAGE
www.ddn.com

BULL
www.atos.net

IBM
www.ibm.com

FORTH
www.ics.forth.gr

OnApp
ww.onapp.com

Institute of communications and computer systems
www.microlab.ntua.gr

MemoScale
www.memoscale.com

webLyzard technology
www.weblyzard.com

LOBA
www.loba.pt

Thales Alenia Space
www.thalesgroup.com

Space Hellas
www.space.gr

CybeleTech
www.cybeletech.com

Neurocom Luxembourg
www.neurocom.eu

MemEX
www.memexitaly.it

Tiemme SPA
www.tiemmespa.it

Virtual Vehicle
www.v2c2.at

AVL List GmBH
www.avl.com

BMW AG
www.bmw.com

KOOLA
www.koola.io

www.evolve-h2020.eu
info@evolve-h2020.eu

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 825061