SODALITE: Software-Defined Execution and Optimization of In-Silico Clinical Trials in HPC with SODALITE Platform

Kamil Tokmakov, Dr. Ralf Schneider
HLRS

The International Conference for High Performance Computing, Networking, Storage, and Analysis (2020)

17.11.2020

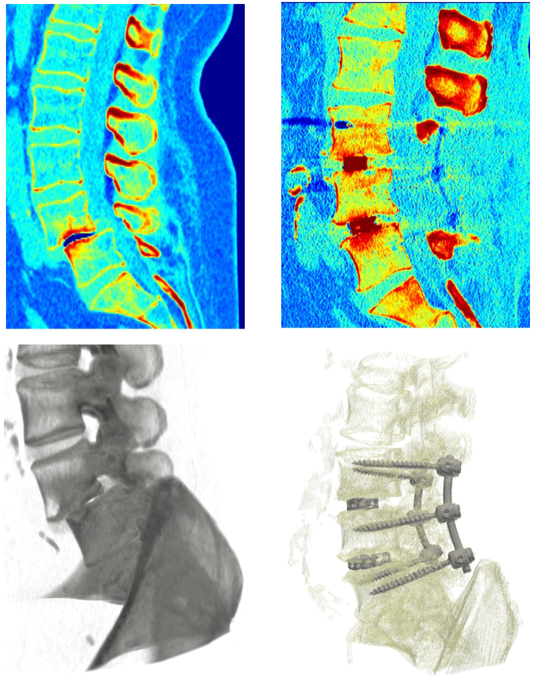
SC20

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 825480.

Medical Problem

Sodalite

- Some spinal conditions (e.g. disk displacement or prolapse) can only be treated ***operatively***. A common treatment is mono- or bisegmental ***fusion of the lumbar spine***.
- ***A screw-rod fixation bone implant system*** is used to fix parts of the lumbar spine.
- Biomechanical implant ***development*** is down to the present day ***done on empirical basis***
- Selection of type, size and placement position is done ***based on experience***
- ***Implant optimization is complicated***

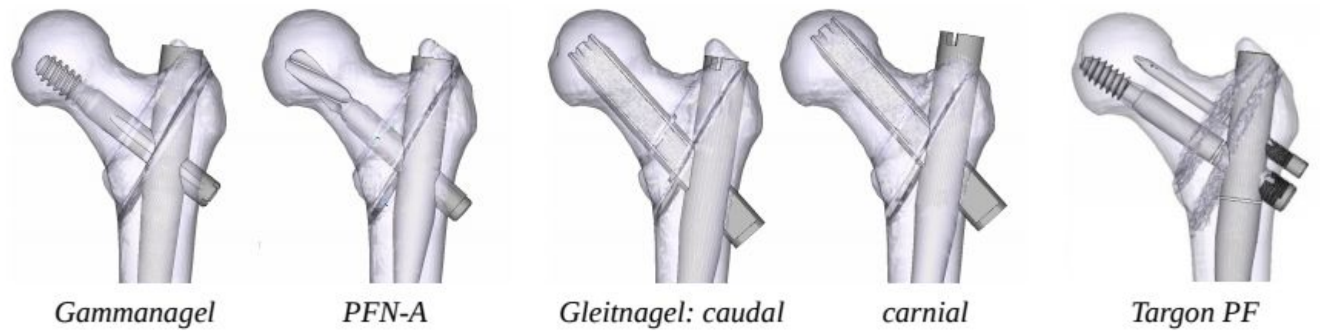


Before and after the fixation

Virtual Clinical Trials

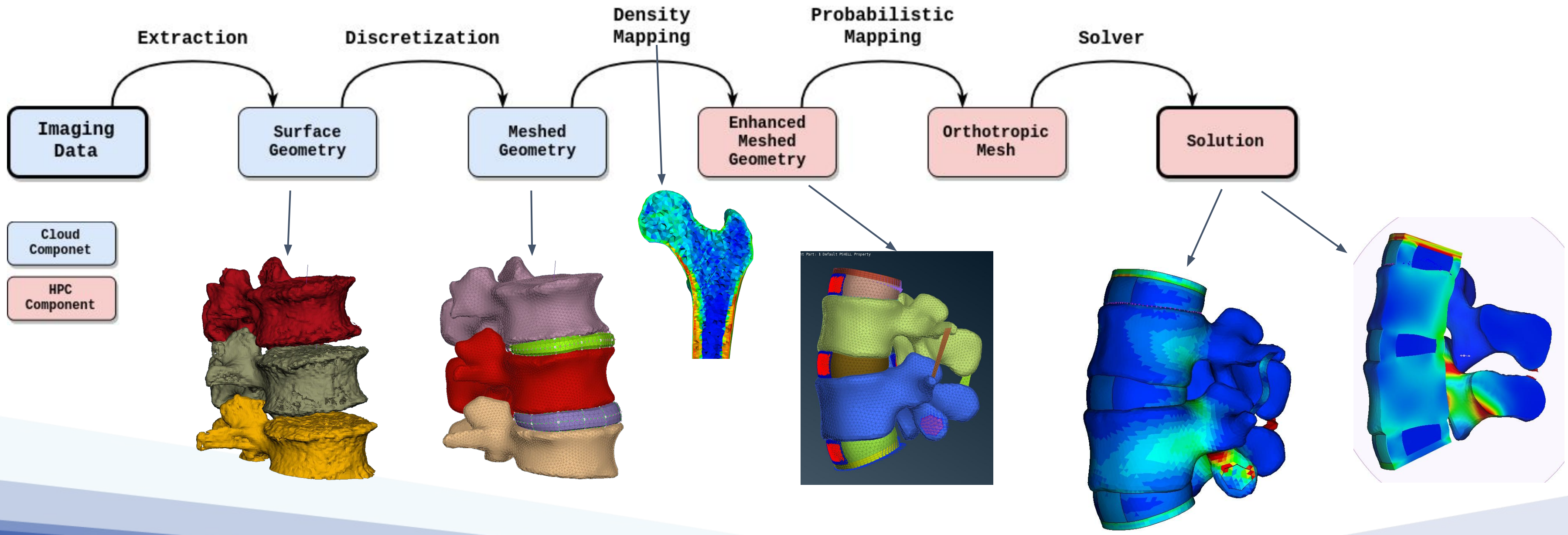
Sodalite

- Clinical trials with *“real”* patients are ***time-consuming and expensive***.
- Every patient is ***different and results can not be generalized***.
- **Virtual clinical trials reproduce clinical trials by means of simulation**.
  - Simulations are applied to virtual patient cohorts.
  - The UC represents research to advance this frontier.



Workflow

Sodalite



Requirements

Sodalite

A scientific workflow, composed of multiple integrated components, with ***efficient data processing*** over ***heterogeneous infrastructure***.
Efficient development and failure management. During a development cycle, a failed simulation should be ***debugged*** and ***restarted*** from the failed component, ***not running the whole chain*** again.
Data processing tasks are not finally defined and ***may change and get more complicated*** as the methodology of clinical virtual trials evolves: e.g. new data analytics component are likely to be introduced.
Efficient uncertainty quantification (currently done ***manually***), which is not only needed in this special case but is widely sought after nowadays.
Evaluation in terms of ***execution time/cost/power*** over ***various infrastructures and computing centers***.

Deployment problem

Sodalite

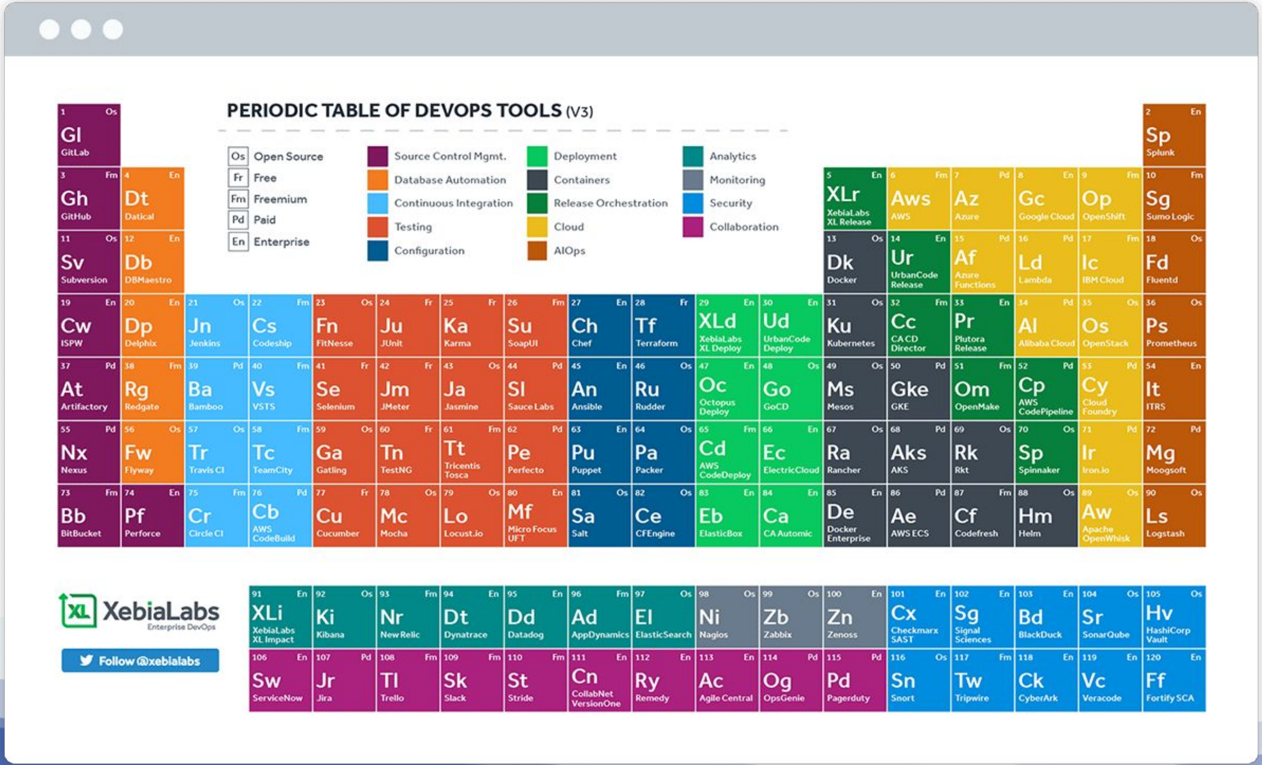
Current methodology of in-silico clinical trials in biomechanical simulations is **not productive**:

- Requires **effectiveness** in deployment, management and adaption to **different** IT-infrastructures (SC, Cloud, HW Heterogeneity)
- Requires **ease-of-use** for end users (medical device manufacturers or medical research institutes) and **reduced effort** of the developers.

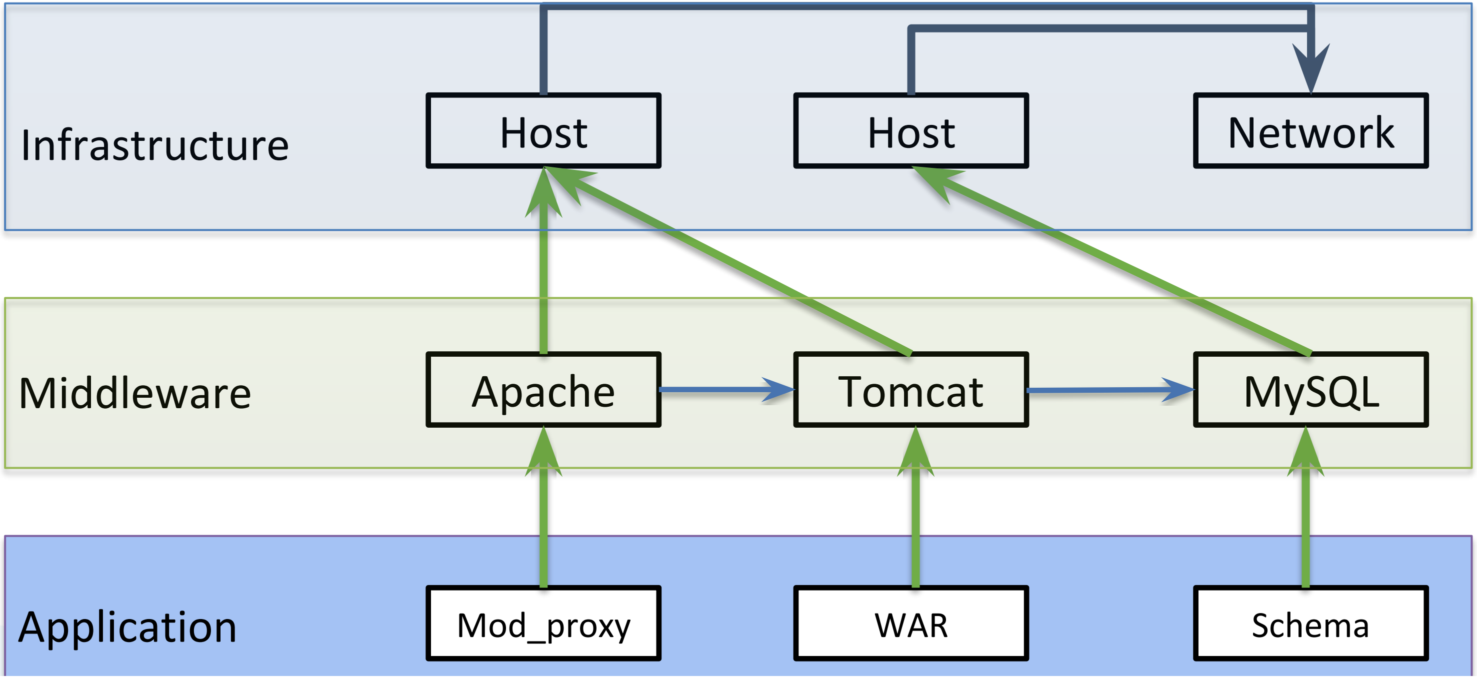
→ DevOps practices shall be adopted: IaC-based abstraction, flexibility, portability, reduced cost and effort

The realm of DevOps tools

Sodalite



Towards standard Infrastructure-as-Code (IaC)



→ <u>OASIS TOSCA</u> **(Topology and Orchestration Specification for Cloud Applications) standard:** quite complex (steep learning curve), no optimisation

SODALITE Vision

SODALITE provides tools to enable **simpler** and **faster development of IaC** and **deployment and execution of heterogeneous apps** in HPC, Cloud & SW defined computing environments.

Particular focus of SODALITE is on performance, quality, and manageability of the applications on the underlying infrastructures.

What SODALITE offers - a selection

Sodalite

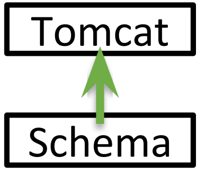
- Smart modeling
- Design-time application optimization
- Automated resource discovery (out-of-scope)
- Runtime optimization and control (out-of-scope)

Smart modeling

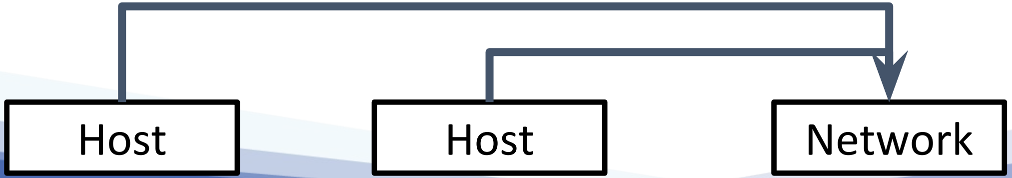
Sodalite

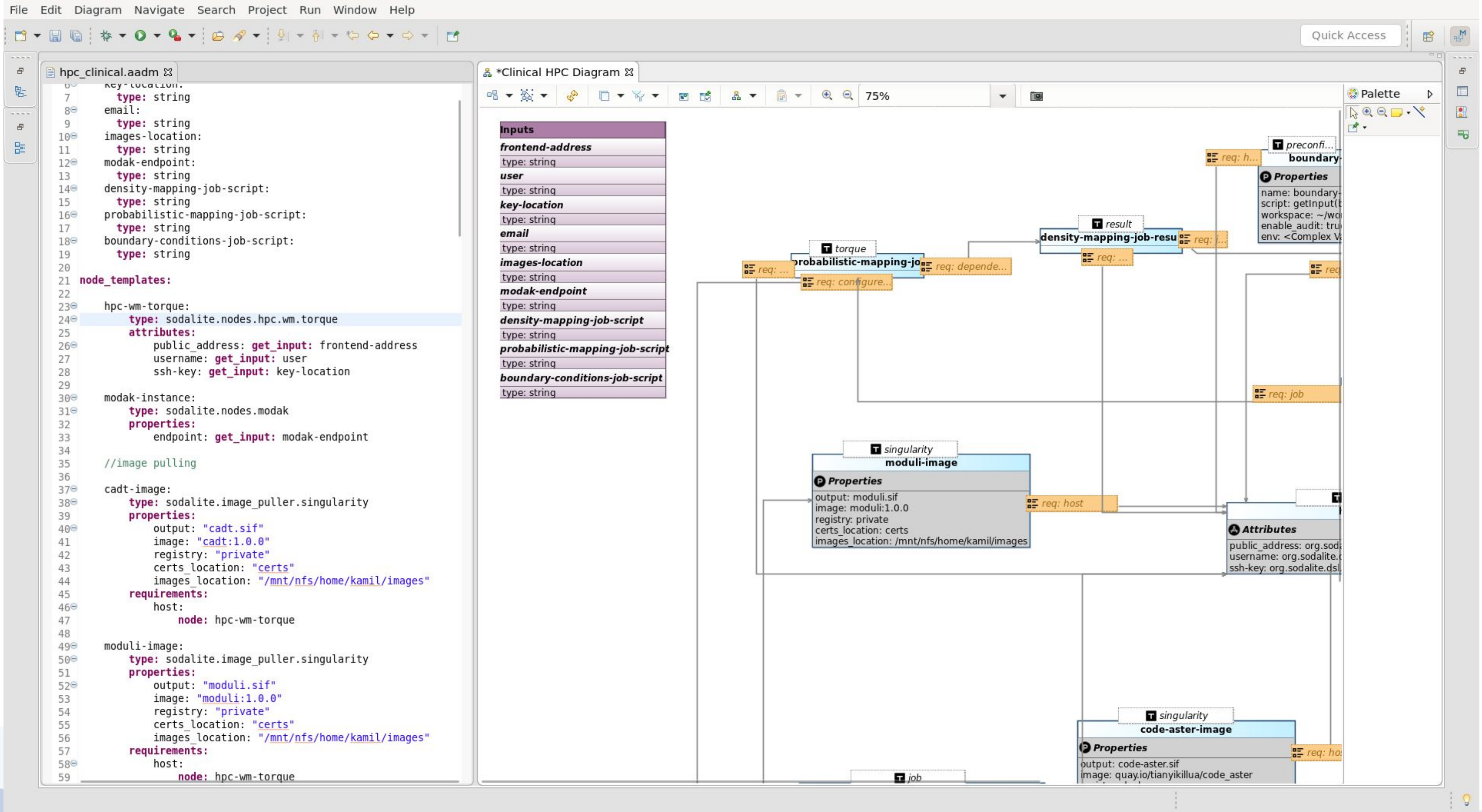
- Smart creation of deployment models through a textual and graphical DSL
- Editing is supported by an ontology-based reasoning mechanism that
  - Checks the semantic validity of a model
    - E.g., it signals a problem if a requirement of a source node is not satisfied by a capability of the target node
  - Enables the development of decision making tools, e.g.:
    - context-aware assistance of user at design-time
    - model enrichment taking into account domain knowledge

Schema requirement is not compatible with Tomcat capability



Two interacting hosts must be connected through a network



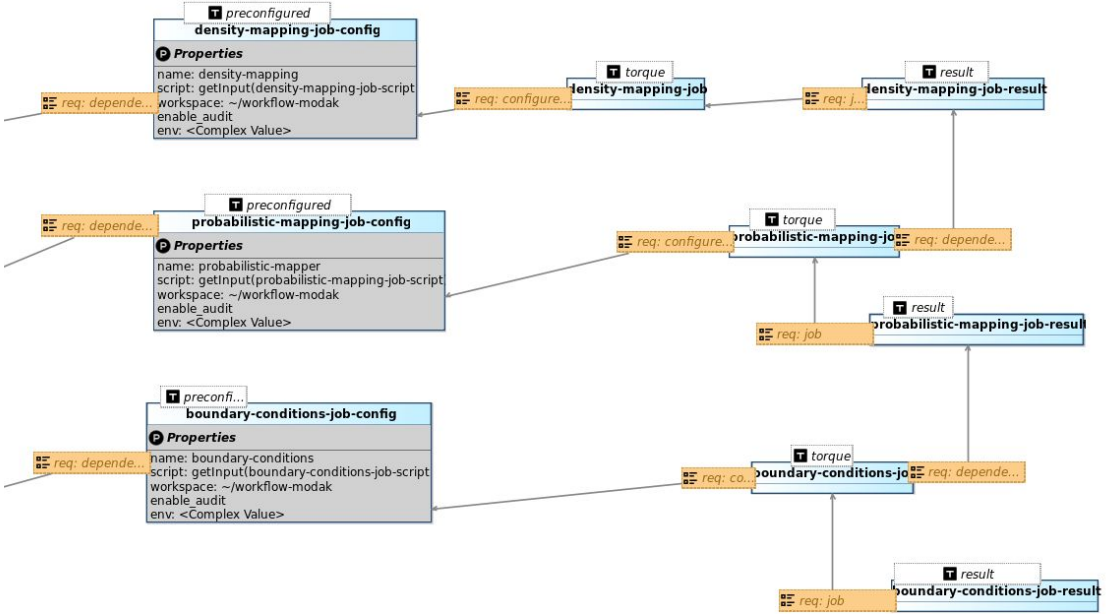


IDE demo: https://www.youtube.com/watch?v=8YC11JFSWC4

Workflow modeling

Sodalite

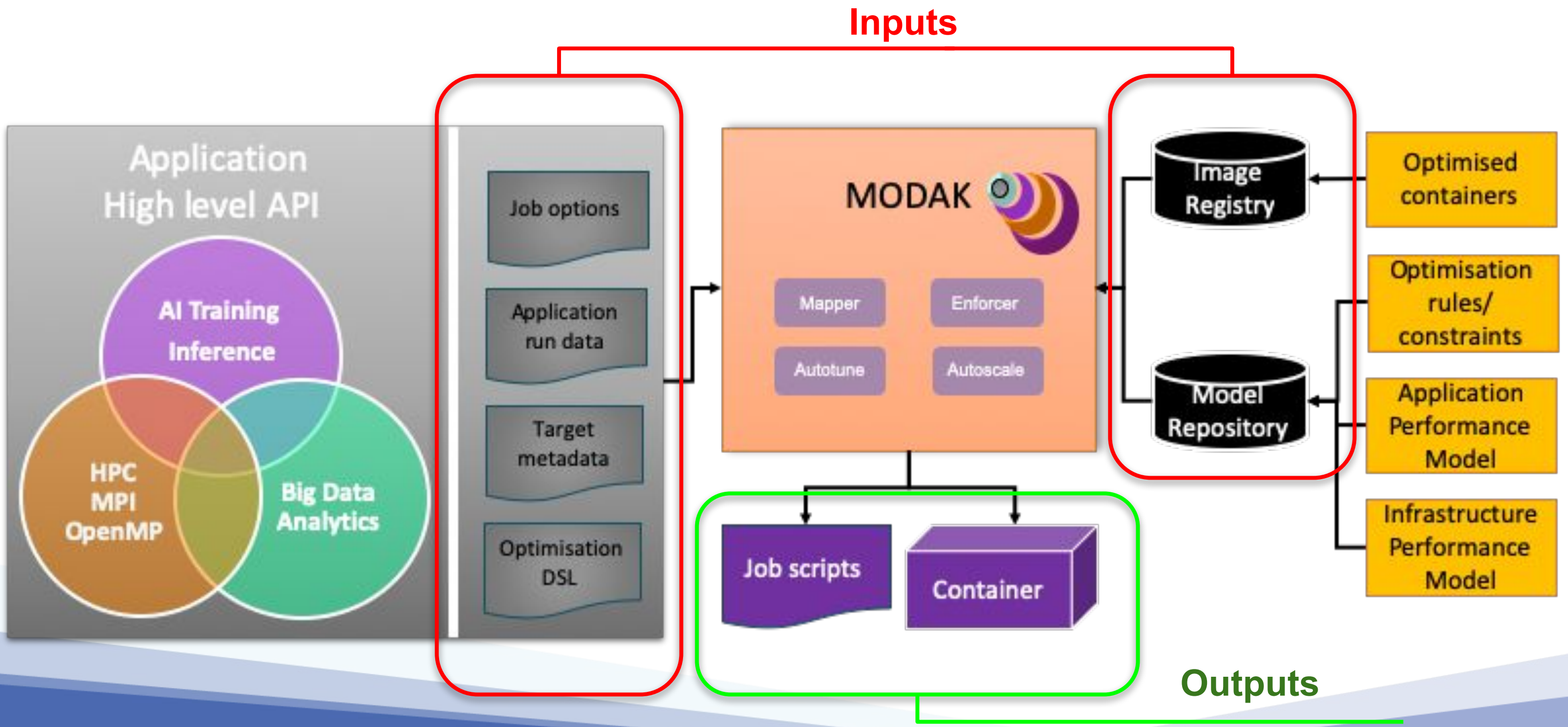
```yaml
density-mapping-job-config:
  type: sodalite.nodes.hpc.job.torque.preconfigured
  properties:
    name: "density-mapping"
    script: get_input: density-mapping-script
    workspace: "~/workflow"
    env:
      SINGULARITY_DIR: "/home/kamil/images"
  requirements:
    host:
      node: hpc-wm-torque

density-mapping-job:
  type: sodalite.nodes.hpc.job.torque
  requirements:
    host:
      node: hpc-wm-torque
    configured_job:
      node: density-mapping-job-config

probabilistic-mapping-job:
  type: sodalite.nodes.hpc.job.torque
  requirements:
    host:
      node: hpc-wm-torque
    configured_job:
      node: probabilistic-mapping-job-config
    dependency:
      node: density-mapping-job-result
```



Full description can be found in IDE GitHub here

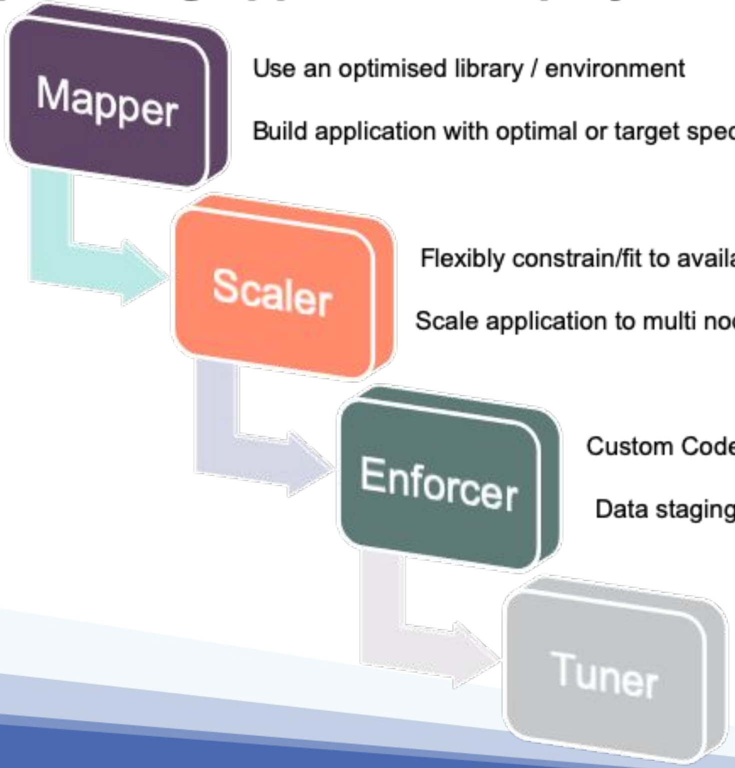
Application optimization

Support to design time application optimization for HPC



MODAK Components

Optimising application deployment



Mapper
- Use an optimised library / environment
- Build application with optimal or target specific compiler (flags/settings)

Scaler
- Flexibly constrain/fit to available resources
- Scale application to multi nodes/GPUs

Enforcer
- Custom Code changes or optimisation settings
- Data staging in storage / memory

Tuner
- Autotune application parameters
- Autotune application build

MODAK API example

Response:

optimised container image
+
job script


```json
{
  "job":{
    "job_options": {
      "job_name": "solver",
      "wall_time_limit": "1:00:00 ",
      "node_count": 2,
      "core_count": 40,
      "process_count_per_node": 40,
      "standard_output_file": "file.out",
      "standard_error_file": "file.err",
      "combine_stdout_stderr": true,
      "request_event_notification": "abe",
      "email_address": "tokmakov@hlrs.de"
    },
    "application": {
      "app_tag": "solver_clinicalUC",
      "app_type": "hpc",
      "executable": "${ASTER_ROOT}/14.4/bin/aster ",
      "arguments": "${ASTER_ROOT}/14.4/lib/aster/Execution/E_SUPERV.py -commandes fort.1 --num_job=1432 --memjeveux=8192.0 --tpmax=3600.0",
      "build": {
        "src": "https://www.code-aster.org/FICHIERS/aster-full-src-14.4.0-1.noarch.tar.gz",
        "build_command": "python3 setup.py install\n"
      }
    },
    "optimisation": {
      "enable_opt_build": true,
      "enable_autotuning":true,
      "app_type":"hpc",
      "opt_build": {
        "cpu_type": "x86",
        "acc_type": ""
      },
      "autotuning": {
        "tuner": "cresta",
        "input": "dsl text"
      },
      "app_type-hpc": {
        "config":{
          "parallelisation":"mpi"
        },
        "data":{
          "location": "L1L2_NonLinear_prepared.tar.gz"
        },
        "parallelisation-mpi": {
          "library": "mpich",
          "version": "3.14",
          "scaling_efficiency": 0.75
        }
      }
    }
  }
}
```


Job parameters - converts into PBS or Slurm job script parameters

Parameters of how to build and execute application

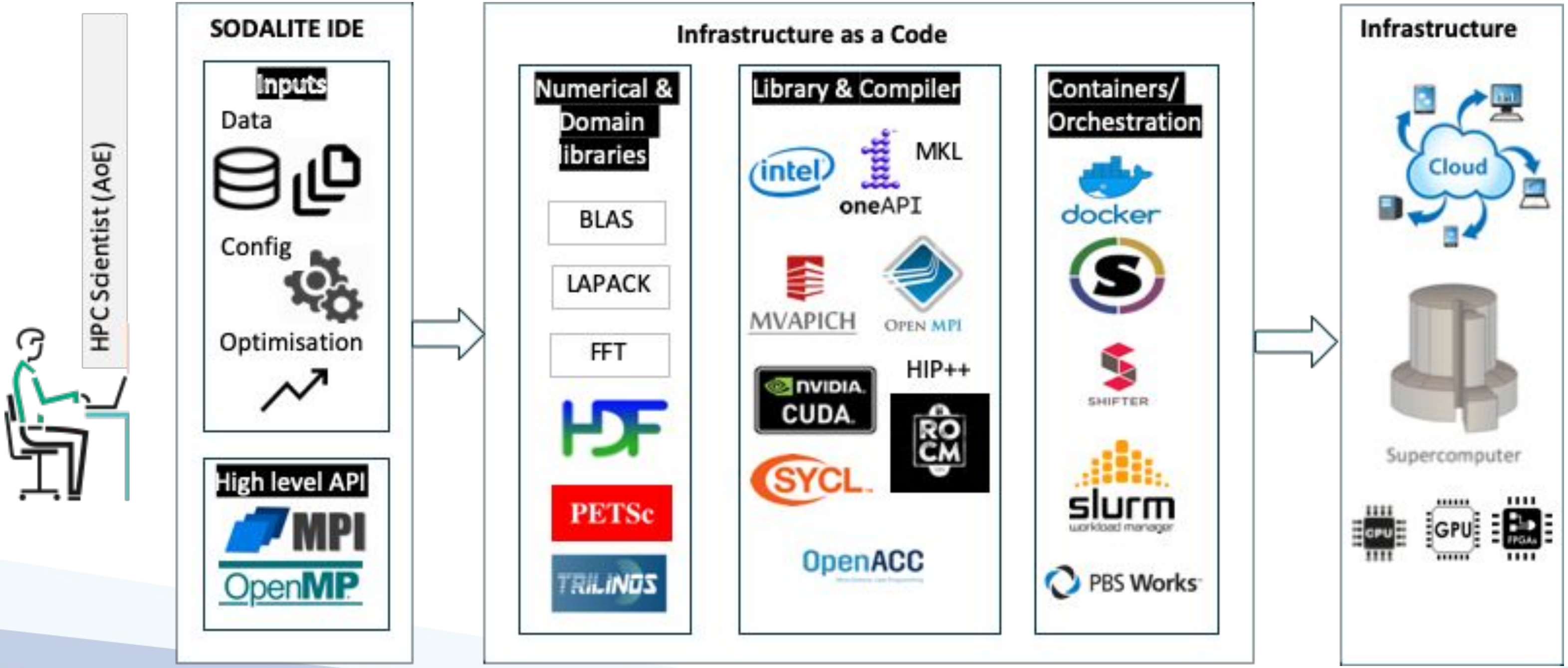
Optimisation parameters - specifying e.g. application type "hpc", parallelisation, CPU architecture, autotuning

MODAK generated job script

Sodalite

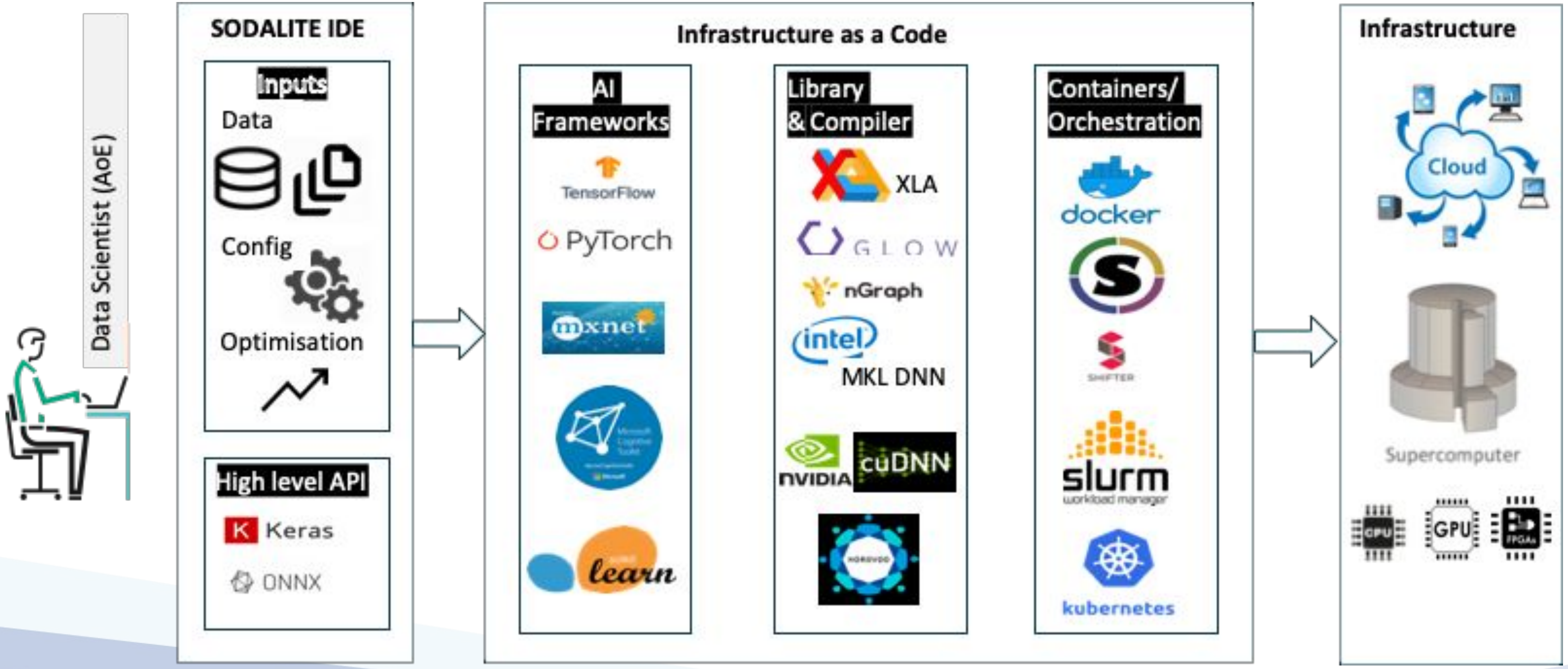
```
#PBS -S /bin/bash
## START OF HEADER ##
#PBS -N solver
#PBS -l walltime=1:00:00
#PBS -l nodes=2:ppn=40
#PBS -l procs=40
#PBS -o file.out
#PBS -e file.err
#PBS -j oe
#PBS -m abe
#PBS -M tokmakov@hlrs.de
## END OF HEADER ##
cd $PBS_O_WORKDIR
export PATH=$PBS_O_WORKDIR:$PATH

## START OF TUNER ##
file=solver_20201116190135_tune.sh
if [ -f $file ] ; then rm $file; fi
wget --no-check-certificate https://storage.googleapis.com/modak-bucket//modak/solver_20201116190135_tune.sh
chmod 755 solver_20201116190135_tune.sh

singularity exec $SINGULARITY_DIR/mpich_ub1804_cuda101_mpi314_gnugprof.sif solver_20201116190135_tune.sh
## END OF TUNER ##

wget --no-check-certificate https://www.code-aster.org/FICHIERS/aster-full-src-14.4.0-1.noarch.tar.gz

singularity exec $SINGULARITY_DIR/mpich_ub1804_cuda101_mpi314_gnugprof.sif  python3 setup.py install


export OMP_NUM_THREADS=1
mpirun -np 40 singularity exec $SINGULARITY_DIR/mpich_ub1804_cuda101_mpi314_gnugprof.sif  \
       ${ASTER_ROOT}/14.4/bin/aster  ${ASTER_ROOT}/14.4/lib/aster/Execution/E_SUPERV.py -commandes fort.1 --num_job=1432 --memjeveux=8192.0 --tpmax=3600.0
```

Optimisation abstraction for Traditional HPC (MPI)



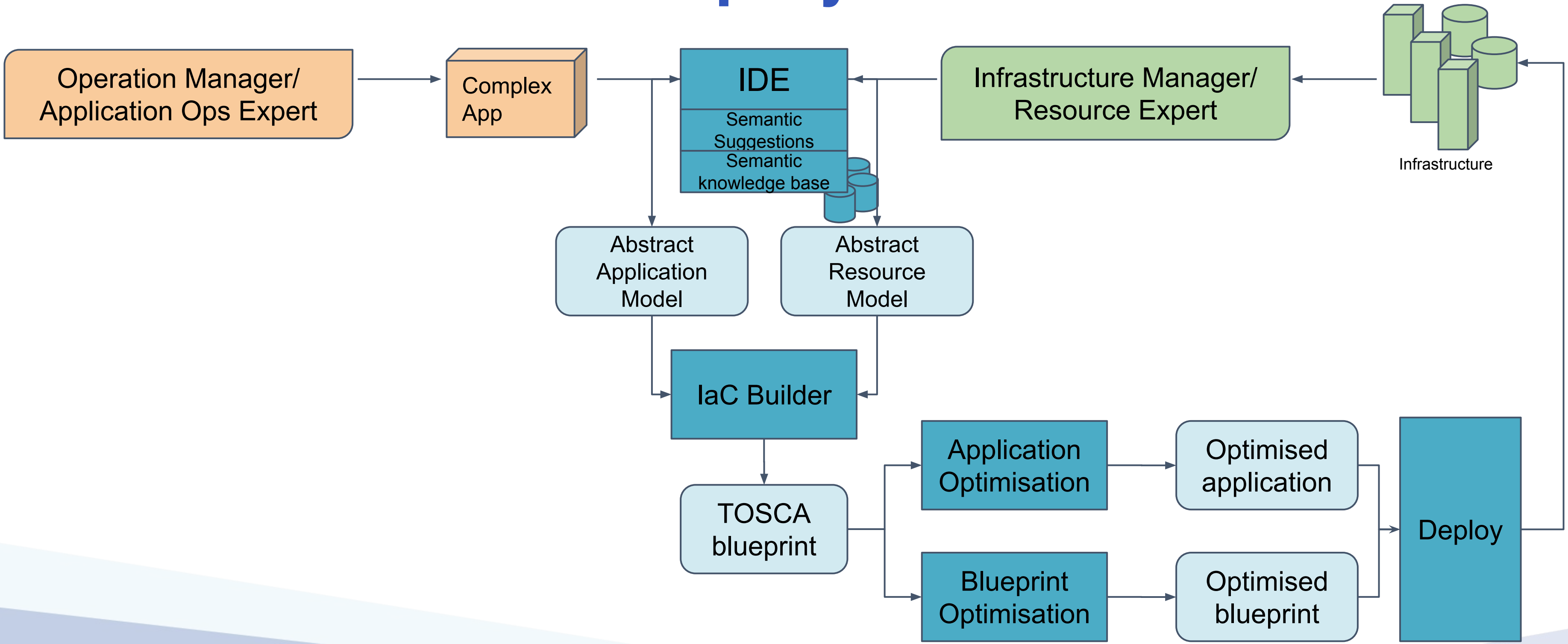
Optimisation abstraction for AI Training

Sodalite



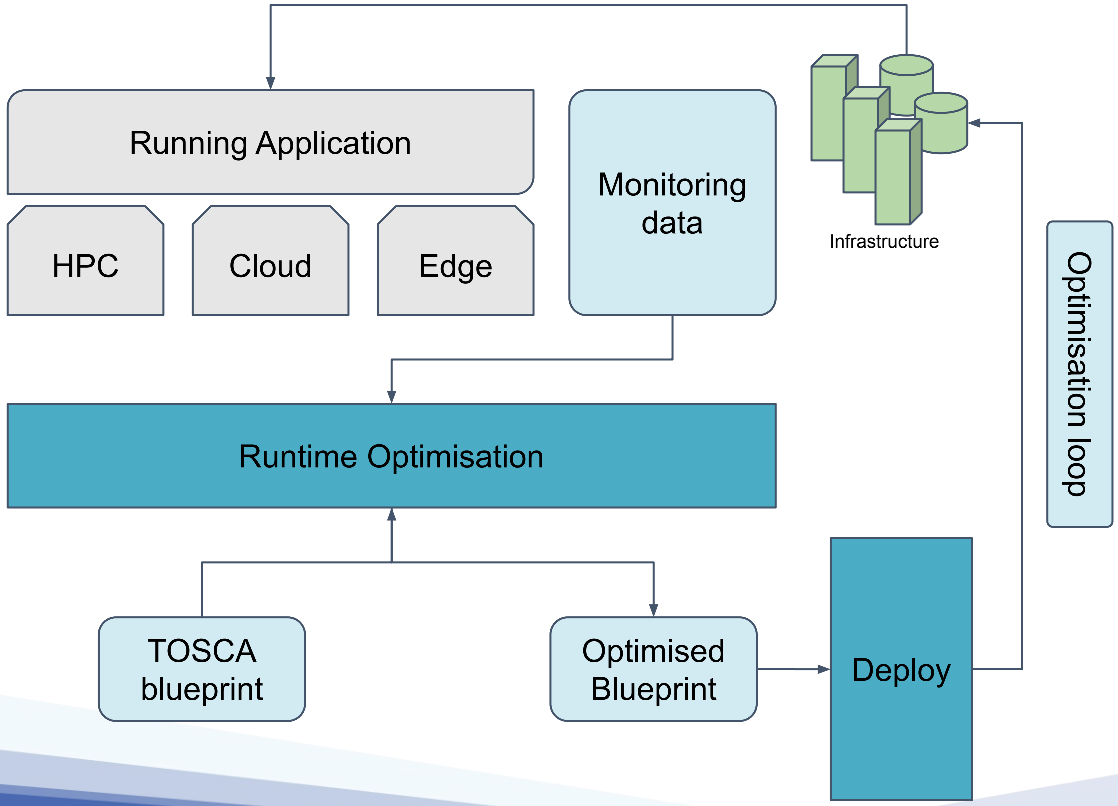
The SODALITE Deployment

Sodalite



The SODALITE Runtime

Sodalite



Performance Optimisation results

Sodalite

Optimisation results for AI training and HPC deployments with graph compilers and Singularity containers

- For **AI training**
  - **17%** speedup using custom built optimised containers
  - up to a **30%** speedup using graph compilers.
- For traditional **HPC** the work is ongoing (Solver optimisation)
  - Performance with singularity **containers** comparable to native build (up to **6%** speedup)

- Presented talk at *Supercomputing Frontiers*, *Warsaw (March 23 – 25, 2020)*
- Presented poster in *ISC-HPC (June 22 – 25, 2020), Frankfurt*
- Paper on *Optimising AI Training Deployments using Graph Compilers and Containers* accepted at 2020 IEEE High Performance Extreme Computing Conference (HPEC) 22 - 24 September 2020
- Submitted paper on *MODAK – an Optimiser for HPC and AI training deployments in software defined infrastructures* to The 2020 International Conference on High Performance Computing & Simulation (HPCS 2020)

Summary

Sodalite

SODALITE outcome already helps Virtual Clinical Trials in biomechanical simulations in moving the process towards production-like environments:

+ *Increase the **effectiveness** of component deployment*
  → *assisted via **IDE**, automated via **orchestrator***
+ *Ease the **adaptation** and **optimisation** for **different hardware/software platforms***
  → ***MODAK** and resource models provided abstraction*
+ ***Lower the efforts** for component **integration***
  → *incorporated components and dependencies (container images, data, artifacts) into the **workflow***
+ ***Lower the efforts** for **data management***
  → ***data management** as part of workflow*

Links

- Website - www.sodalite.eu
- GitHub Page - SODALITE-EU
- Docker Hub - sodaliteh2020
- YouTube Channel - Sodalite H2020
- Linkedin Page - sodalite-eu
- Twitter - @SODALITESW

Sodalite

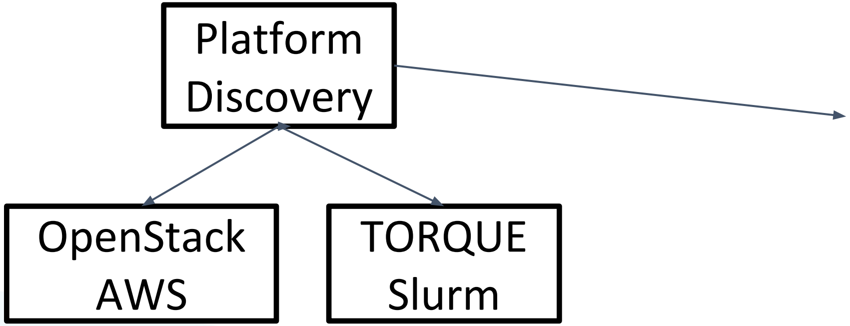
This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 825480.

Backup slides

Automated discovery of resources

Sodalite

- Automatic discovery and modeling of new infrastructural resources into IaC




```yaml
sodalite.nodes.hpc.resources.torque:
  derived_from: tosca.nodes.Compute
  properties:
    name:
      type: string
      default: hlrs_testbed
    total_gpus:
      type: integer
      default: 5
    total_nodes:
      type: integer
      default: 5
    total_cores:
      type: integer
      default: 200
```

Runtime deployment optimization

Sodalite