

# FEDERATED COMPUTING

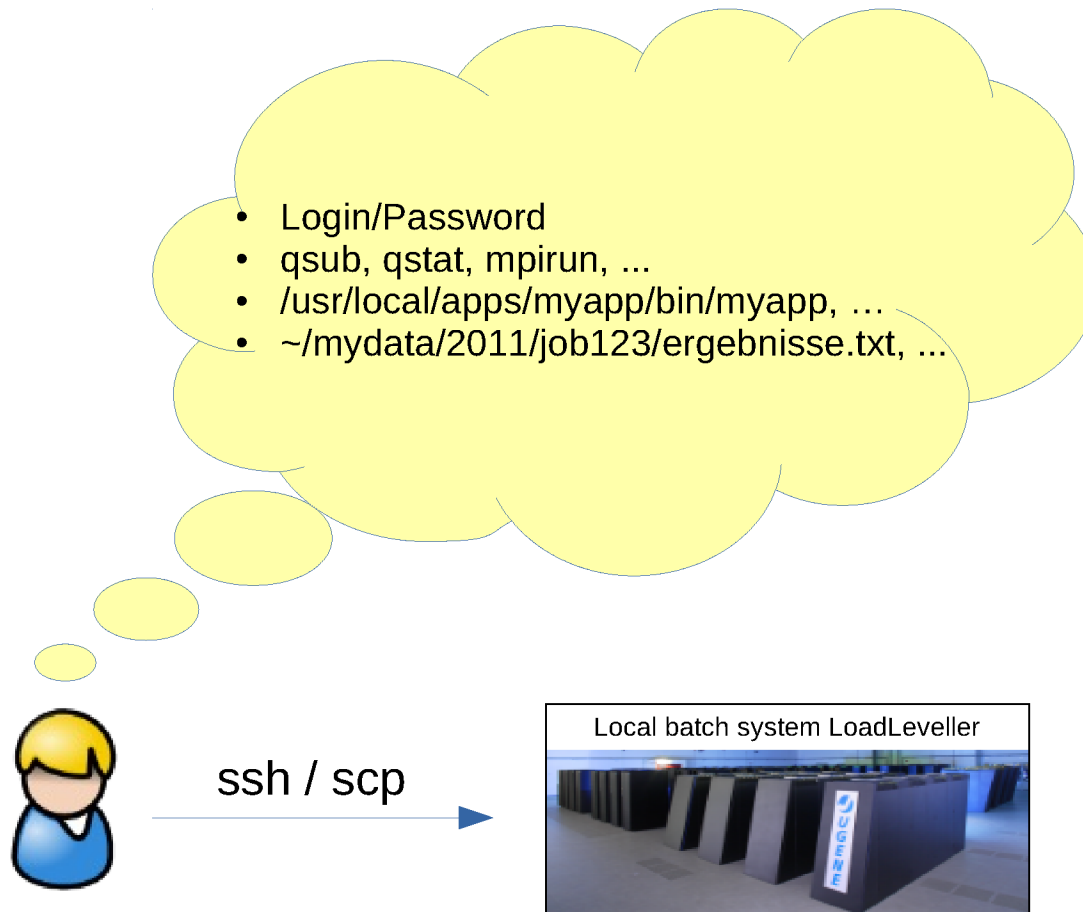
with UNICORE

FEBRUARY 20, 2019 | B. SCHULLER

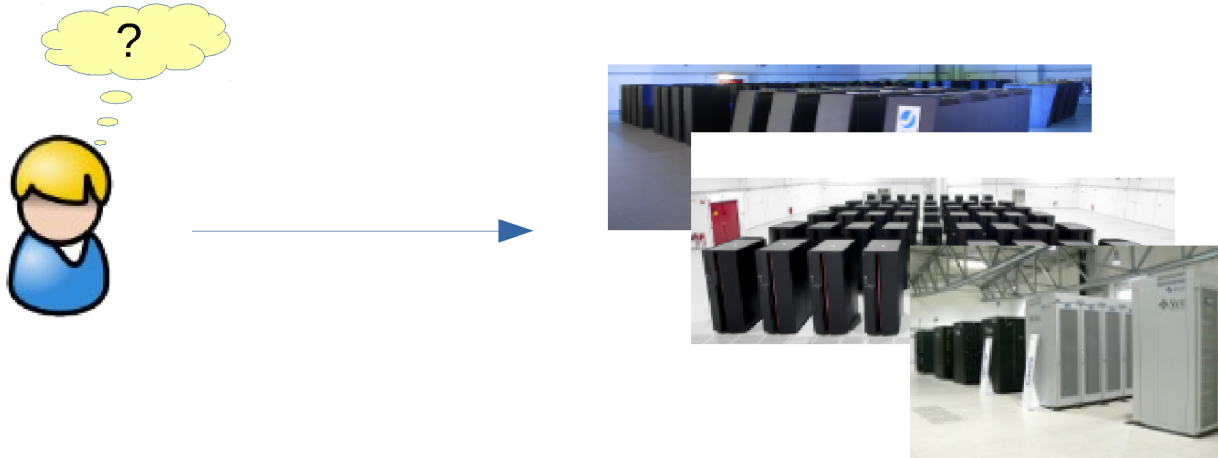
# OUTLINE

- Federating HPC with UNICORE and Unity
- Administrative concerns
- Use cases at JSC and their implementation
- Outlook on UNICORE 8

# MOTIVATION

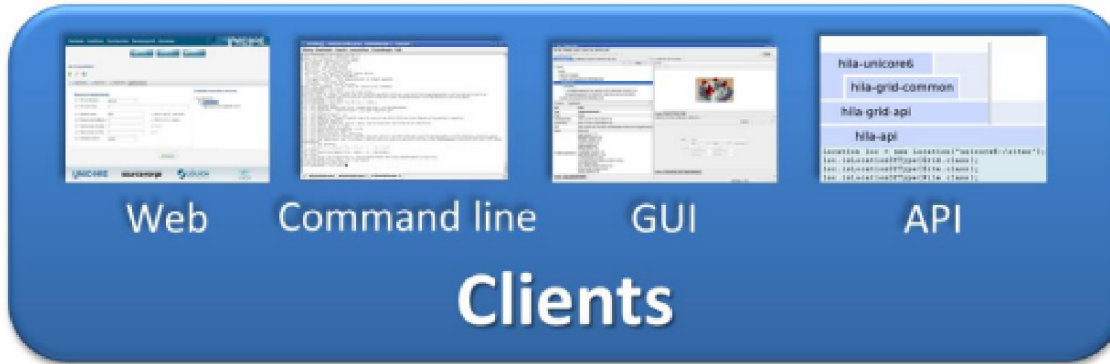


# MOTIVATION



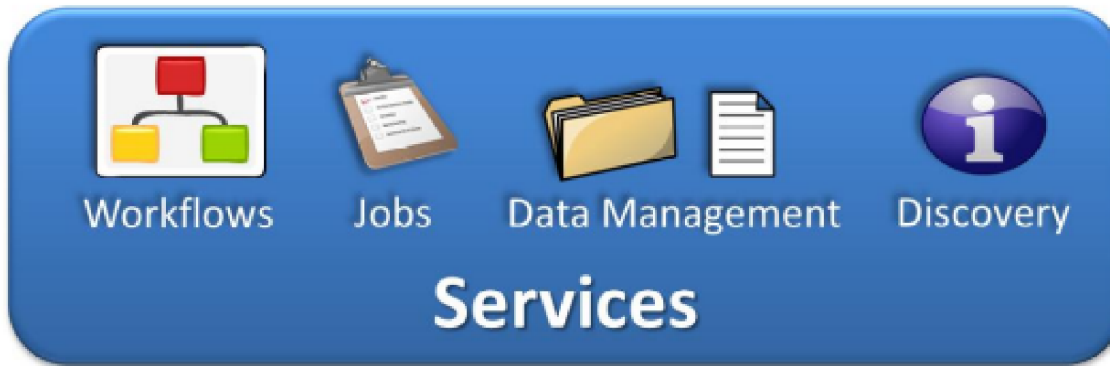
How can I...

- ... use multiple, heterogeneous systems seamlessly?
- ... manage my job input data and results?
- ... across systems? Workflows?
- ... integrate HPC into custom applications and portals?



Web Command line GUI API

## Clients



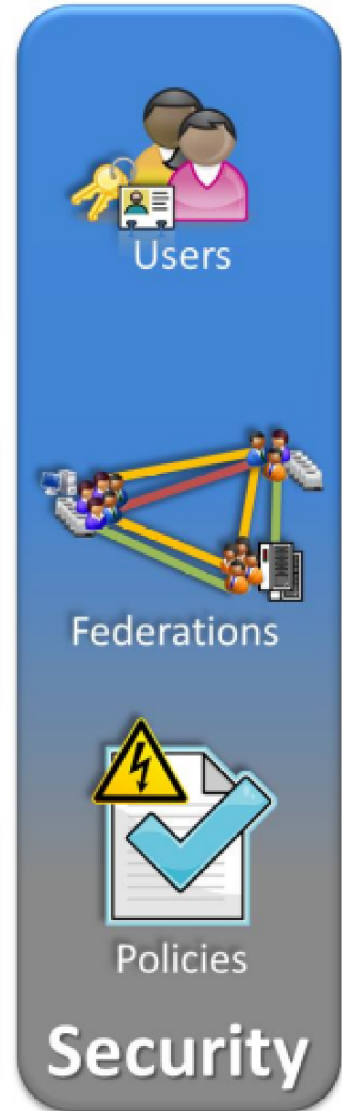
Workflows Jobs Data Management Discovery

## Services



Compute Storage

## Resources

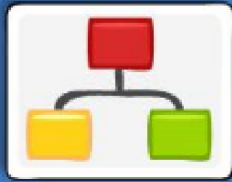


Users Federations Policies

## Security

# UNICORE KEY FEATURES

- A set of software components for integration of HPC into federated environments
- Federated authentication, local authorization, site keeps full control
- REST APIs for jobs, data, workflows
- Simplifies HPC use for non-experts (Application concept, abstract resource model, workflow templates, ...)
- Flexible and non-intrusive
- Independence of the OS and batch system



Workflows



Jobs



Data Management



Discovery

## Services

- Workflow enactment
- Task execution
- Job submission
- Job management
- Reservations
- File systems
- Storage access
- Metadata
- File transfers
- Service registry
- Resource brokering

# JOB SUBMISSION AND MANAGEMENT



→  
Job submission

→  
Access to  
working directory

→  
Abort/Restart  
jobs



- Cluster managers  
(Slurm, LSF, ...)

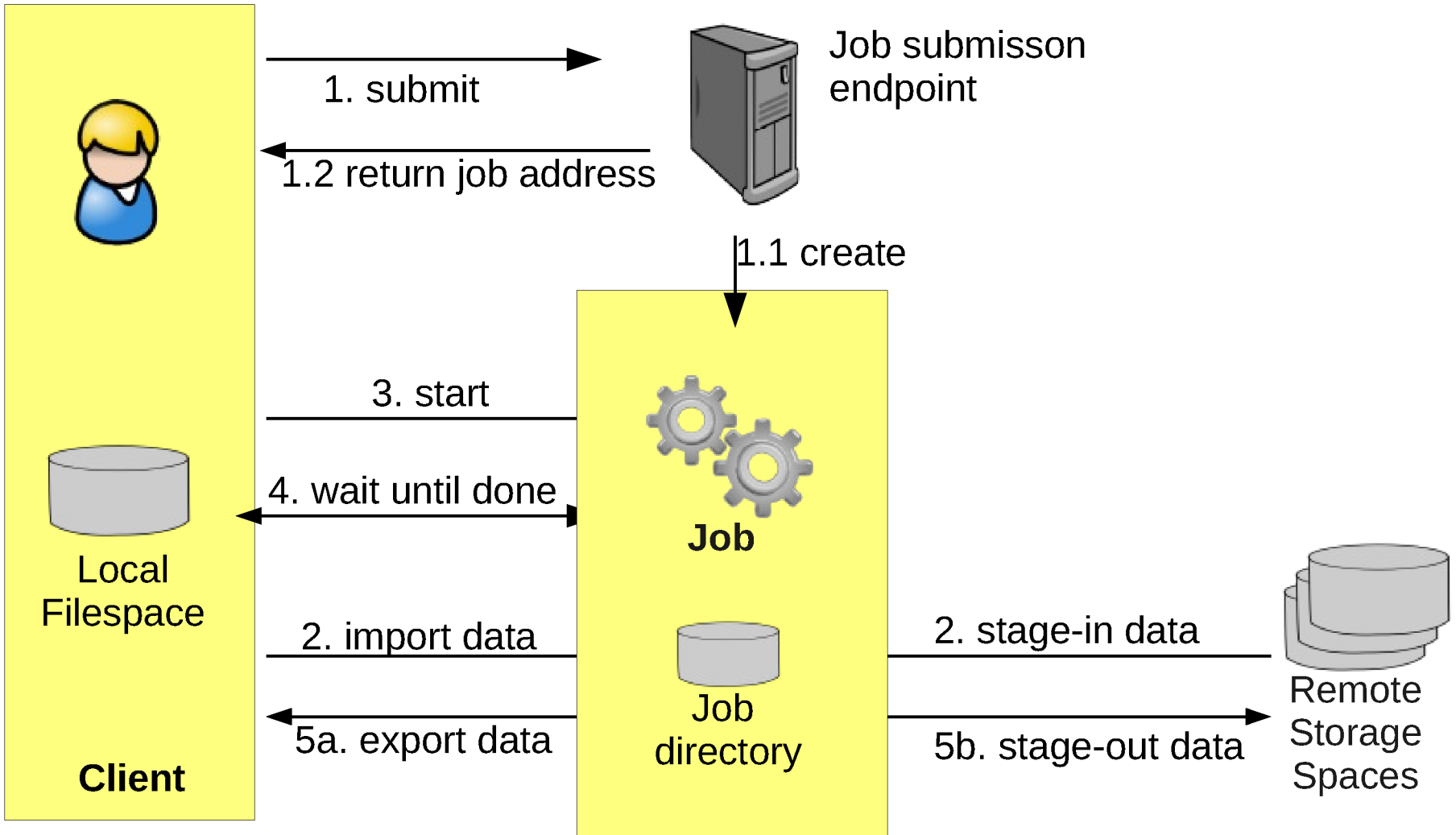
- YARN



- SSH tunnels



# JOB EXECUTION



# UNICORE'S "APPLICATION" CONCEPT

To setup a simulation run you need to ...

- ... manage working directory, code, input params
- ... create and submit batch scheduler script

```
#@job_name      = slns_demo
#...
#@bg_size       = 32
#@wall_clock_limit = 00:10:00

module load python3/3.4.2
export TMPDIR=$WORK/tmp
export PYTHONPATH=/homeb/slns/slns007/local/opt/...

runjob --ranks-per-node 1 --exp-env ... : /bgsys/.../python3 microcircuit.py
```

# UNICORE'S "APPLICATION" CONCEPT

- Admin (or power user) defines UNICORE application

```
<idb:IDBApplication>

  <idb:ApplicationName>NEST</idb:ApplicationName>

  <jsdl:POSIXApplication>
    <jsdl:Executable>runjob --ranks-per-node 1 --exp-env ... : .../python3</jsdl:Executable>
    <jsdl:Argument Type="filename">$NESTCODE?</jsdl:Argument>
    <jsdl:Argument Type="filename"># $PARAMETERS?</jsdl:Argument>
  </jsdl:POSIXApplication>

  <idb:PreCommand>#@environment = COPY_ALL</idb:PreCommand>
  <idb:PreCommand>module load python3/3.4.2</idb:PreCommand>
  <idb:PreCommand>export TMPDIR=$WORK/tmp</idb:PreCommand>
  <idb:PreCommand>export PYTHONPATH=/usr/local/...:$PYTHONPATH</idb:PreCommand>

  <idb:PostCommand>find -name *gdf | xargs zip output.zip</idb:PostCommand>

</idb:IDBApplication>
```

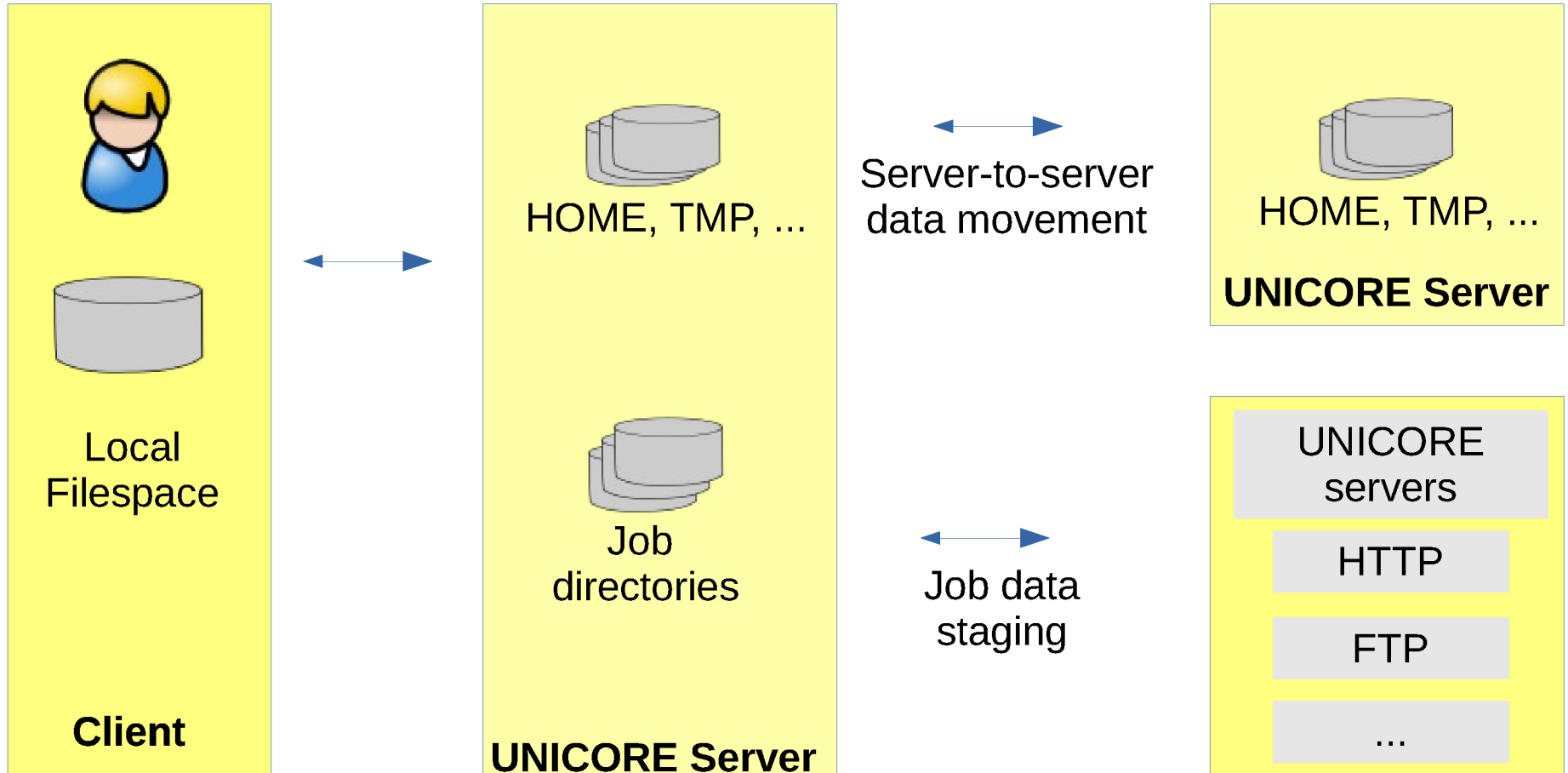
# UNICORE'S "APPLICATION" CONCEPT

Complexity is now hidden!

- Users need only provide *relevant* parameters
- ... jobs can even be 100% the same on different compute clusters

```
{  
  ApplicationName: NEST,  
  
  Parameters: [  
    NESTCODE: microcircuit.py, PARAMETERS: parameters.py, ],  
  
  Imports: [ ... ],  
  
  Resources: { Nodes: 32, Runtime: 1200 },  
}
```

# DATA AND STORAGE SERVICES



# DATA MANAGEMENT FEATURES

- Posix-like operations (ls, stat, mkdir, cp, rm, ...)
- File transfers (client-server, server-server)
- Metadata management
  - Schema-free, key-value
  - Indexed via Lucene, searchable
- Rule-based data processing
  - New files automatically trigger actions
  - Possible use cases: file registration, metadata extraction, compression, etc

# WORKFLOWS

- Built on single jobs
- Powerful constructs
  - Graphs
  - Loops
  - Conditions, variables
  - Hold/continue (with variable changes)
- Data management
  - Input / output / intermediate files

# WORKFLOW EDITOR

UNICORE Rich Client

File Edit Window Help

100%

UNICORE

Grid Browser

Grid

- VSGC Registry
  - Workflow engine at zam025s02.zam.kfa-juelich.de
    - VSGC
    - VSGC-2
    - Global storage at zam025s02.zam.kfa-juelich.de
    - Storage factory at zam025s02.zam.kfa-juelich.de

Key	Value
Name	Workflow engine at zam025s02.zam.kfa-juelich.de
State	Ready
CurrentTime	2010-03-15 12:16:39
Type	WorkflowFactory
Version	2.1.0-rc1
TerminationTime	2020-03-11 15:20:05
Dialects	http://www.chemomentum.org/workflow/simple, h

Tools

- Select
- Control Flow
- Applications
  - Blender v0.2
  - Generic v1.0
  - POV-Ray v3.51
- Structures
  - If-Statement
  - While-Loop
  - Group
- Variables
  - Declaration
  - Modifier

Start

Script1

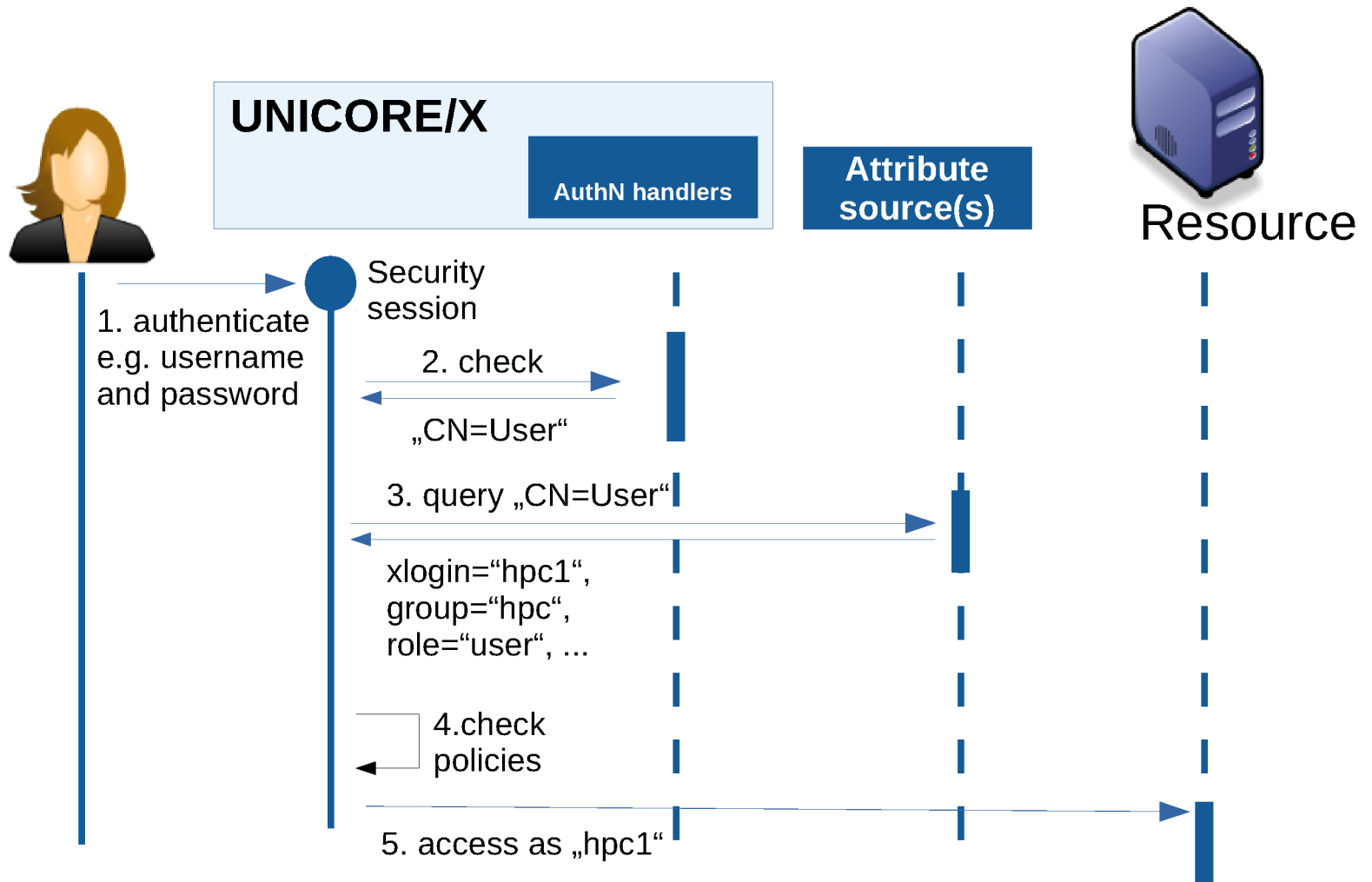
Script2

# SECURITY, AUTHENTICATION, ETC.

# USER AUTHENTICATION

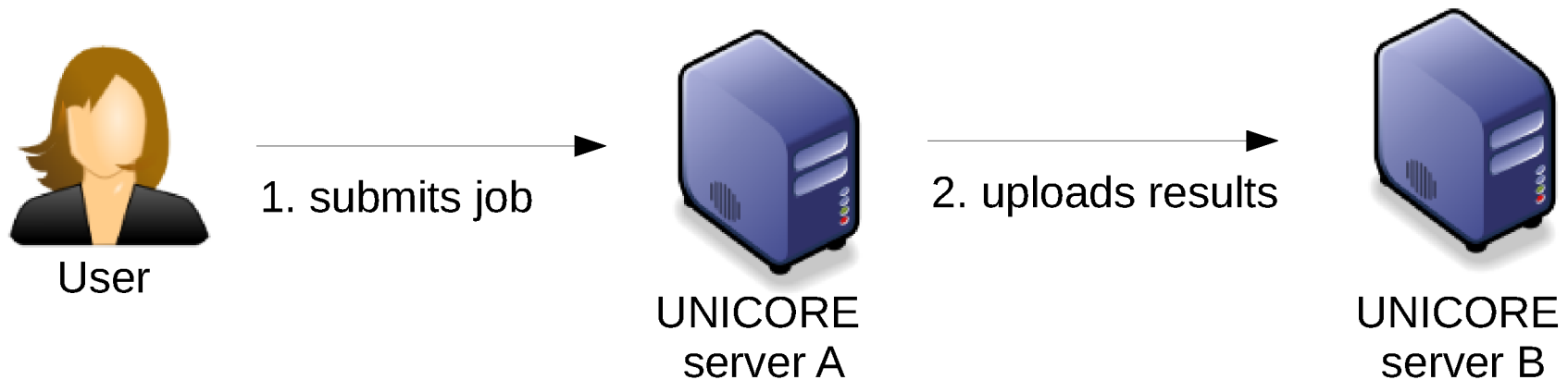
- Via Unity
  - Username/password (PAM, LDAP, Unity internal)
  - OAuth
  - SAML IdPs
  
- X509 certificates (better not)

# AUTHN / AUTHZ FLOW



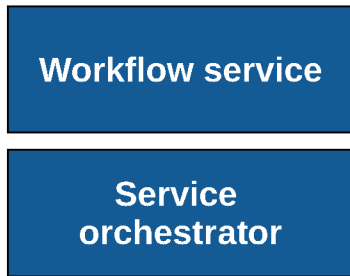
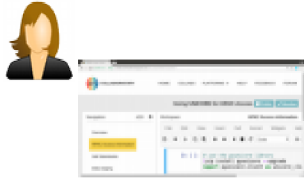
# DELEGATION

- Allows services to work on behalf of the user
- Without storing and passing actual user credentials
- SAML-based (SOAP/WS)

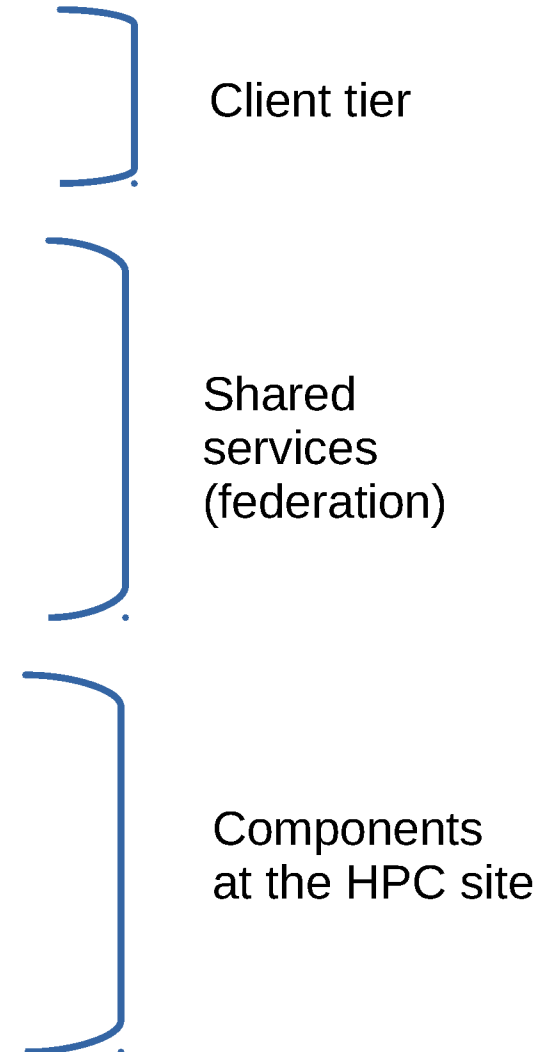


# DEPLOYMENT

# UNICORE COMPONENTS



Local RMS (e.g. Slurm, LSF, etc.)



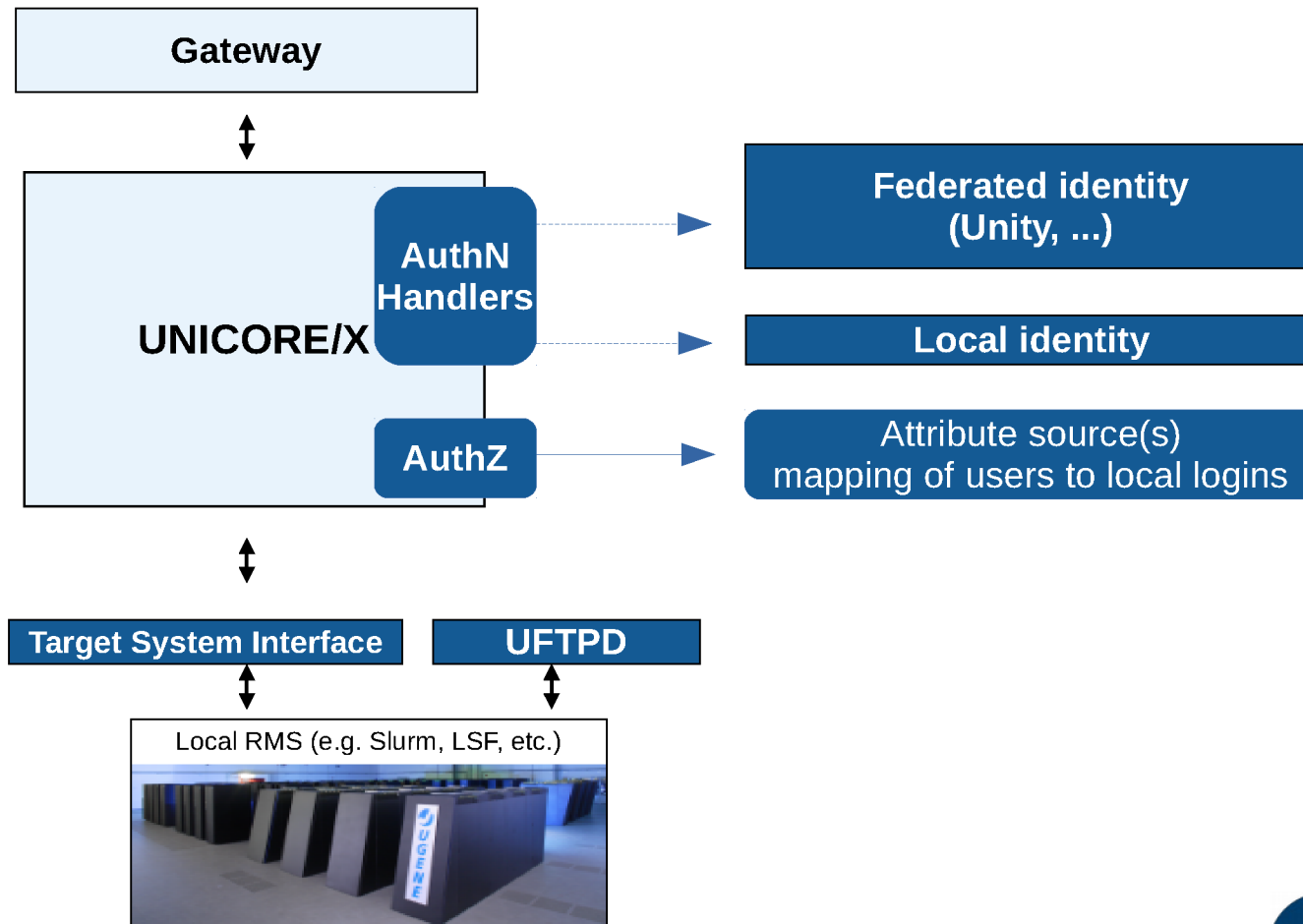
Client tier

Shared services (federation)

Components at the HPC site

# UNICORE COMPONENTS

- Access to a single cluster



# COMPONENT OVERVIEW / DEPLOYMENT

- **Gateway (optional)**

- „Firewall entry point“ for all client requests
- Gateway address must be accessible for clients and for UNICORE/X server – all other components can run in the intranet
- Used to access multiple servers „behind“ it (multiple UNICORE/X, Registry, Workflow, ...)

- **UNICORE/X**

- Main server, connects to all other components
- One UNICORE/X per resource (i.e. cluster + associated file systems) is required
- Requires memory and processing power

# COMPONENT OVERVIEW / DEPLOYMENT

- **TSI (target system interface)**
  - Needs to be installed on the resource (e.g. cluster login node)
  - Needs access to file systems, resource management system (submit jobs, check queues etc)
  - Needs to run as root for multi-user operation
  
- **UFTPD (UNICORE file transfer protocol daemon) (OPTIONAL)**
  - High-performance file transfer daemon
  - Similar requirements as TSI: runs as root on a cluster access node
  - UFTPD „listen“ port must be accessible from the internet, firewall must be configured for FTP)

# COMPONENT OVERVIEW / DEPLOYMENT

- **XUADB („Extended UNICORE user database“) (OPTIONAL)**
  - Default attribute source
  - Maps Grid users to local Unix accounts and roles (user, admin, ...)
  - Can be used by multiple services at a site (multiple UNICORE/X, Registry, Workflow etc)
  
- **Registry (OPTIONAL)**
  - Useful when more than one resource are to be accessed via UNICORE
    - At a single site
    - In a larger, multi-site installation
  - Very much like UNICORE/X in terms of configuration (except it does not connect to a TSI or UFTPD)

# COMPONENT OVERVIEW / DEPLOYMENT

- **Workflow server (OPTIONAL)**
  - Workflow submission and management interfaces
  - Workflow enactment
  - Very similar to a UNICORE/X server
  
- **Service Orchestrator (OPTIONAL)**
  - Resource broker
  - Executes single workflow tasks (jobs)
  - Very similar to a UNICORE/X server

# COMPONENT OVERVIEW / DEPLOYMENT

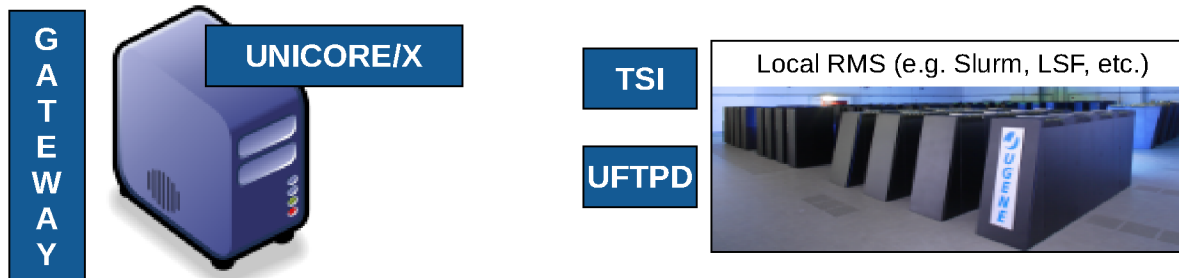
- **Unity (federated identity) (OPTIONAL)**
  - Maps authentication options to Grid users
    - SAML, username/password, OAuth, ...
    - Integration with third-party AAI systems (e.g. EduGAIN)
  - Can be used by all UNICORE/X-like services in the Grid
- **Must:** configure Unity public key as „trusted SAML issuer certificate“ in all UNICORE/X servers
- **Optional:** REST API authentication in UNICORE/X

# PERFORMANCE NOTES

- More than one UNICORE service can be installed on a single machine
- Choose machines that can stand the expected load
- Especially UNICORE/X, Workflow, Service Orchestrator require memory and CPU (512 MB per component, 2 cores per component)
- UNICORE/X, Workflow, Service Orchestrator require storage for their state
  - File system (simple, but creates load)
  - MySQL server (reduces load)

# MINIMAL

- Single cluster accessed via UNICORE
- One Linux machine for Grid components
  - Gateway (optional), UNICORE/X
- Cluster frontend node
  - TSI, UFTPD (optional)



# STORAGES CONFIGURATION

- Configure directories that can be used as UNICORE storages
  - WORK, SCRATCH, HOME, ...
- „Storage factory“ service
  - Allows users to create storage instances dynamically
  - Think remote „mkdir“ on a shared space
  - Good for temporary data
  - Recommended if a workflow system is to be used
  - Should be installed only on systems with large and fast storage capacity

# FILE TRANSFER: UFTP

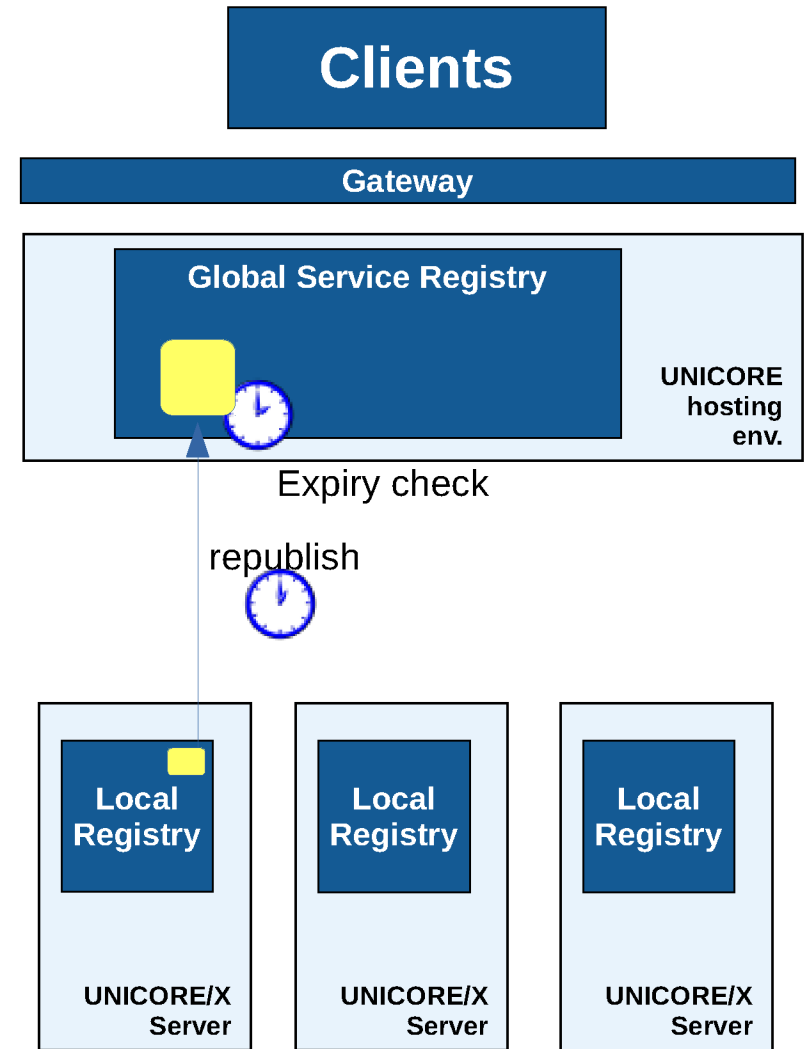
- Default file transfer through HTTPS: simple (zero config) and reasonably fast (some MB/sec.)
- High performance option: UFTP („UNICORE FTP“)
- Warmly recommended for environments / users who want to move lots of data
- Requires additional server (UFTPD) on a system with access to the filesystems (e.g. login node)
- Fast (typically many 10s of MB/sec) on single files
- UFTPD is a separate download (tgz, rpm, deb) and comes with detailed installation instructions

# ATTRIBUTE SOURCES

- Mapping of Grid identity (DN) to local attributes (role, local Unix login, local Unix groups)
- Currently three options exist
  - File: simple, easy to use, local to a UNICORE/X
  - XUADB: web service interface, can be used by multiple UNICORE/X servers at a UNICORE site, admin commandline tool
  - Unity: SAML web service interface, can be used by all sites in a UNICORE Grid, very powerful, but complex
- Attribute sources can be chained for maximum flexibility and administrator control
- Full details are given in the UNICORE/X manual

# REGISTRY

- Aggregates service information from multiple UNICORE sites
- An Entry contains at minimum:
  - Service address
  - Service type
  - Server DN
- Content is held up-to-date
  - Services register with local registry
  - Local registries push info to global registry
  - Entries in global registries expire if not re-published in time

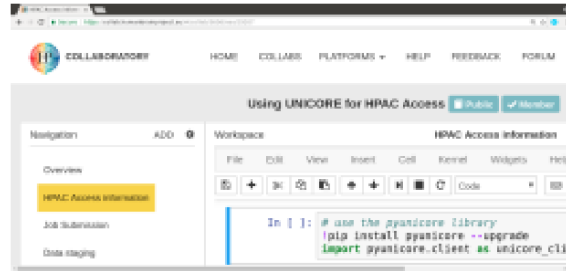


# UNICORE @ JSC

# UNICORE @ JSC



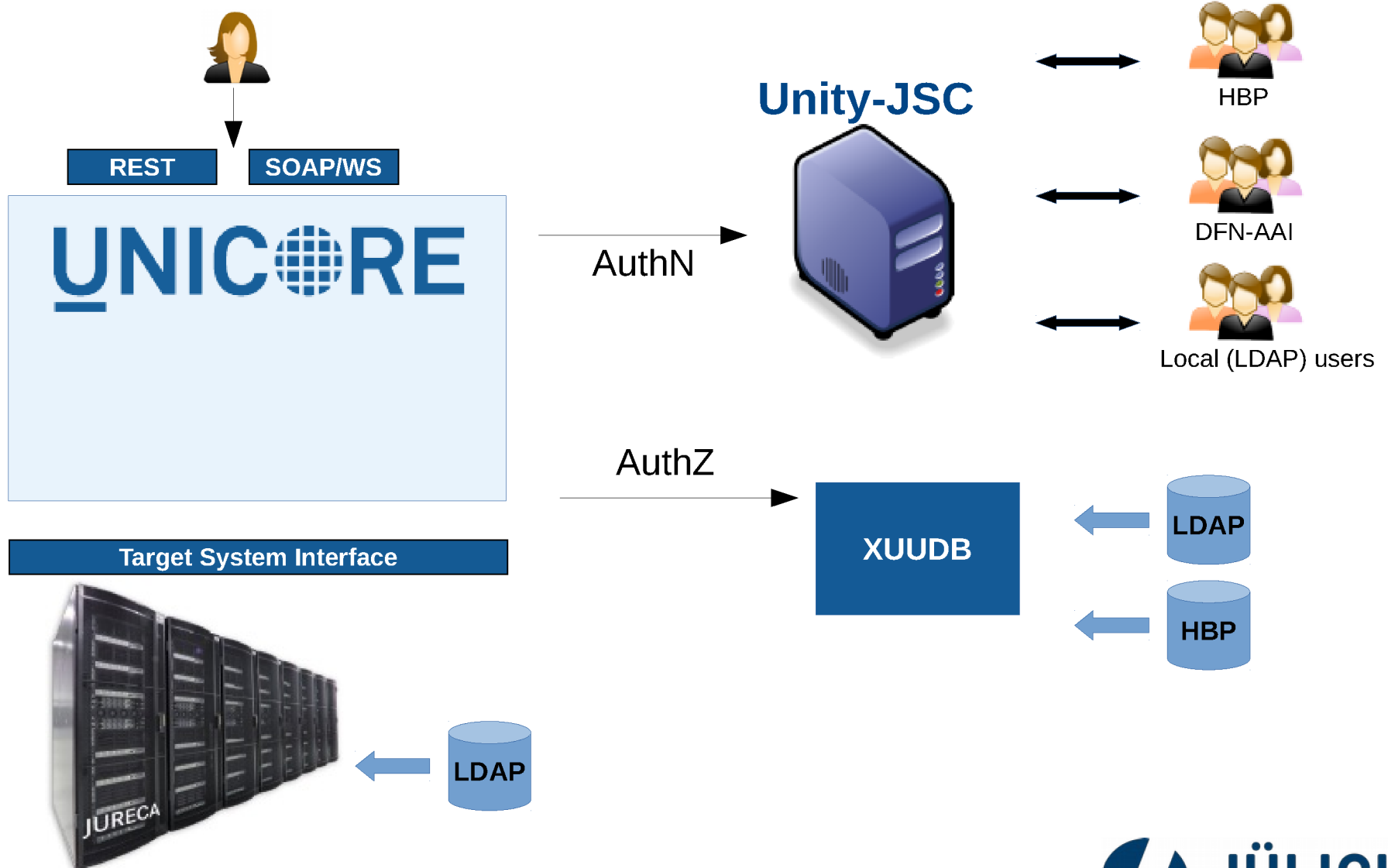
End-user clients  
(ucc, Rich Client, ...)



Human Brain Project



# UNICORE @ JSC



# UNICORE @ JSC

- Authentication via Unity
  - Multiple user communities served via different endpoints
  - E.g. HBP users vs Jupyter users
- Authorization: automated management of entries in UNICORE's user database
- JSC HPC LDAP / user DBs as primary information source

# UNICORE @ JSC

- One UNICORE/X per system (JURECA, JUWELS, ...)
  - Shared XUADB
  - Shared MySQL for persistence
- Multiple Registry instances (HBP, FZJ ...)
- Multiple workflow service instances (HBP, FZJ, ...)
- Single UFTPD instance accessing the (shared) GPFS

# EXAMPLE USE CASE: JUPYTERHUB

- Goal: secure and easy-to-use setup for accessing JSC's machines via Jupyterhub
- Authentication
  - Users log in with JSC LDAP username/password
- Authorization
  - XUADB User DB entries taken from JSC LDAP (automated)

# SUMMARY: UNICORE FOR FEDERATING HPC

- Main challenges
  - Open up HPC to non-traditional use cases
  - Access from community portals like HBP Collaboratory
  - Data sharing
- Solutions
  - UNICORE – compute, storage, workflows
  - Unity – federated identity management
  - UFTP – high-performance data transfer with sharing capabilities

# OUTLOOK: UNICORE 8

- Reduce complexity
- Strengthen core use cases
- REST API as the primary API

# REDUCE COMPLEXITY

- Remove some unused features
- Simplify config files
  - Remove internal details, simple property format
- Back-end resource definition and description
  - Introduce simpler format (JSON)
  - Auto-generate e.g. from Slurm information

# REDUCE COMPLEXITY

- Simplify workflow system
  - Drop “tracer” component
  - Join Workflow/Service components by default
- Out-of-the-box RESTful authentication without Unity
  - PAM module (username/password)
  - SSH keys support

# STRENGTHEN CORE USECASES

- Model current clusters
  - Heterogeneous, modular, many partitions and node types
- Improve abstract batch submission
  - extend “Application” concept
- Support low-level batch system interaction
  - Sometimes UNICORE is the only way for a user to access the system. We do not want to limit what the user can do.

# REST API AS PRIMARY API

- Full support for delegation
  - Using JWT (“JSON web tokens”) specification
- Dual SOAP/REST clients
- SOAP APIs will be dropped in UNICORE 9

# UNICORE 8 ROADMAP

- 7.x maintenance mode – further releases only as necessary
- First release of UNICORE 8 planned for Spring/Summer 2019