LSDMA

JÜLICH FORSCHUNGSZENTRUM

Data-driven processing

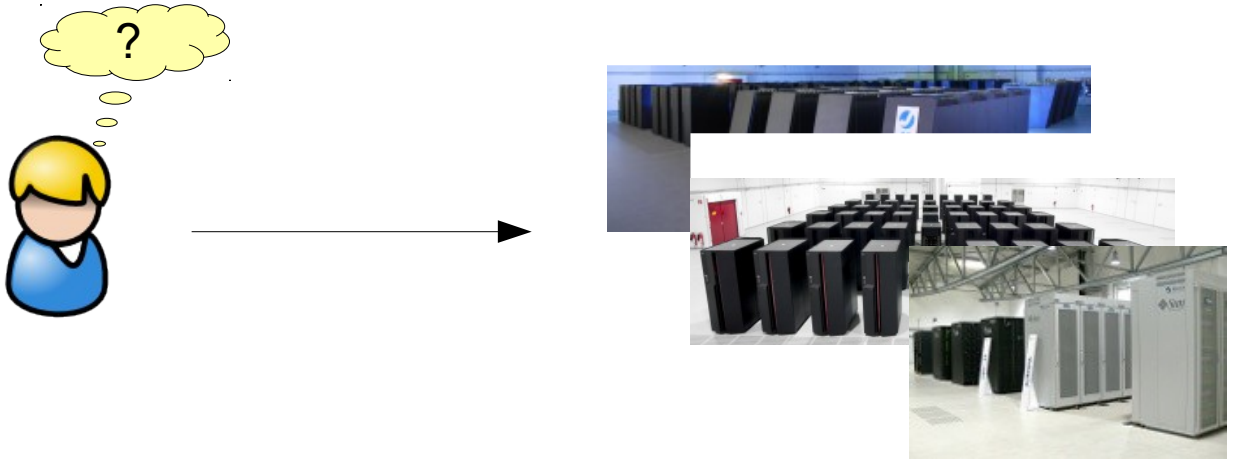
Bernd Schuller

Federated Systems and Data division, JSC Forschungszentrum Jülich GmbH

1 June 2015, LSDMA topical meeting „Data-intensive computing“

Outline

- UNICORE overview
- Data oriented processing
- Outlook



How can I ...

- … use multiple, heterogeneous systems seamlessly,
- … manage my jobs
- … manage input data and results? Metadata?
- … across systems? Workflows?

LSDMA

JÜLICH FORSCHUNGSZENTRUM

UNICORE

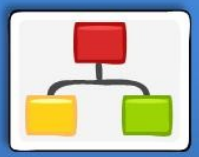

Web


Command line


GUI

API

Clients


Workflows


Jobs


Data Management


Discovery

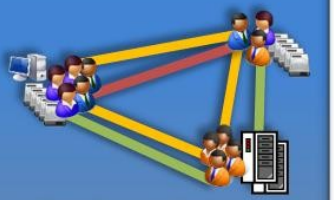
Services


Compute


Storage

Resources


Users

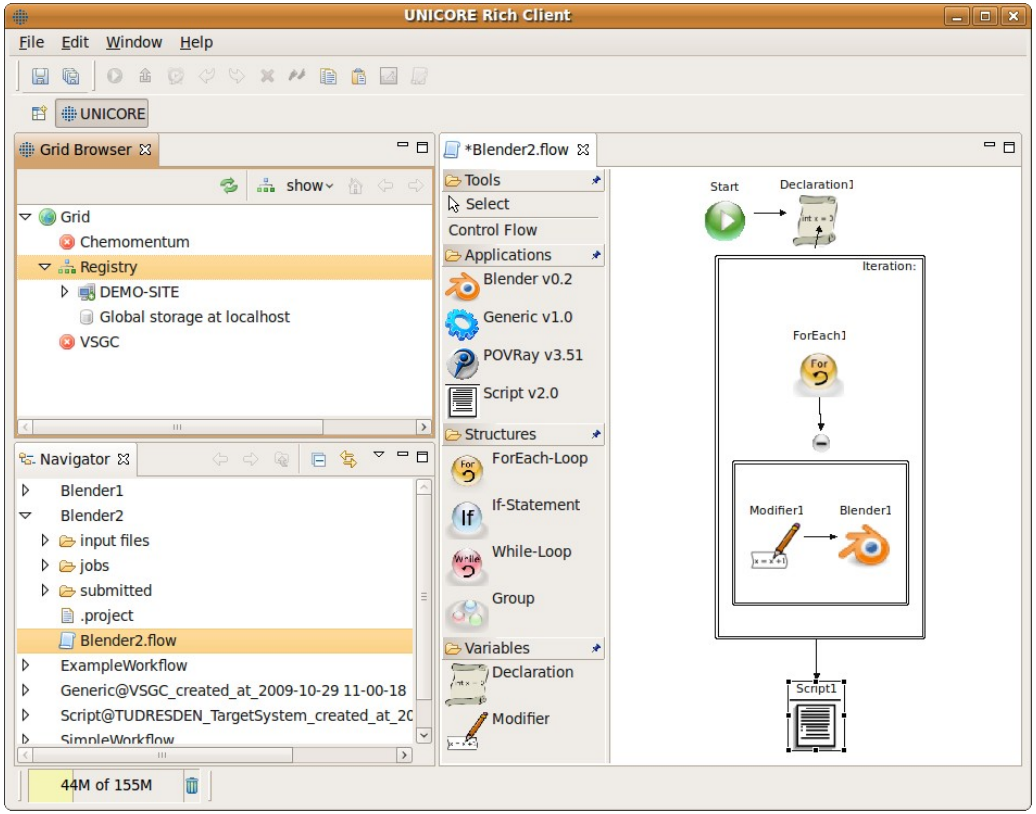

Federations


Policies

Security

Traditional ways of data processing using UNICORE

- Single jobs
- Workflow system
- End-user clients
  URC / UCC / Portal
  or REST APIs



Single jobs

- Batch job oriented
  - Data stage-in
  - Execution
  - Data stage-out
- End-user must ...
  - Setup job definition
  - Select site
  - Upload input data
  - Submit

Workflow system

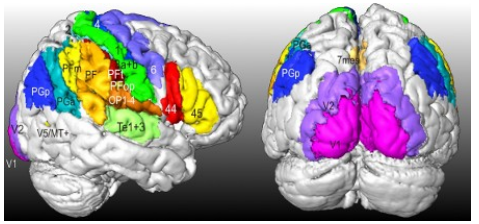
- Sequences / Graphs / Control
- Based on single jobs
- End-user client tasks
  - Setup workflow definition
  - Upload input data
  - Submit
- Pros
  - Easy automation of complex processes
  - Control constructs available
  - Low load on client side
- Cons
  - High overhead on servers
  - Data staging can be a limiting factor
  - Direct user interaction needed

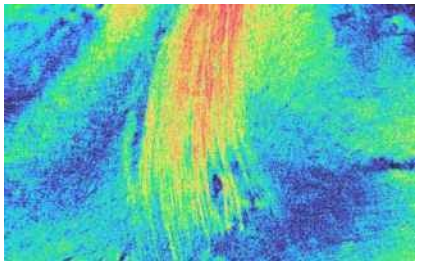
Idea: „data-oriented“ processing

- As opposed to „job-oriented“
- Driven purely by data
- No end-user involvement required (apart from setup)
- Kind of like a „cron job“

Example use case: High-throughput brain scans – the „Data Lifecycle Lab Health“ at Jülich

- Goal is to create a 3D brain atlas
- Brain section scans (ex vivo) (~2000 slices, 500GB per slice)
- MRT scans (in vivo)
- Post-processing: image registration, calibration, segmentation, etc
- Image processing (incl. HPC)
- Raw data often re-processed (new algorithms, new software versions)



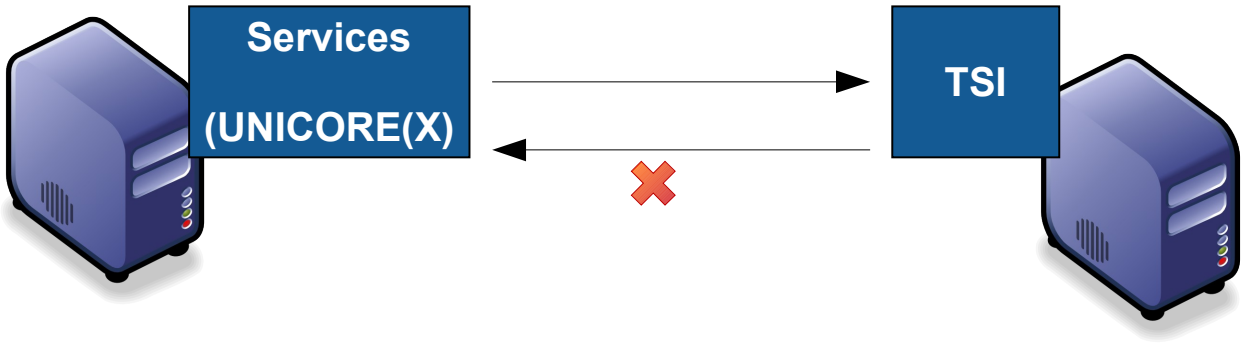


Basic UNICORE architecture - I

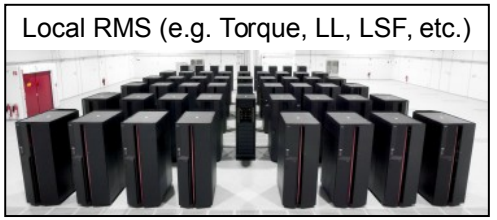
- User centric
- Everything is „owned“ by a user (submission services, jobs, storages, file transfers ...)
- Fully compatible with Unix file permissions
- UNICORE never operates as a „superuser“

Basic UNICORE architecture - II

- Services / logic lives on the UNICORE/X server
- File system and batch system accessed via TSI agent
- TSI accessed via request/response
- No file system notifications possible with current TSI



Cluster access node



UNICORE Storage Management Service

Client

mkdir, ls, rm, stat, ...

upload
download

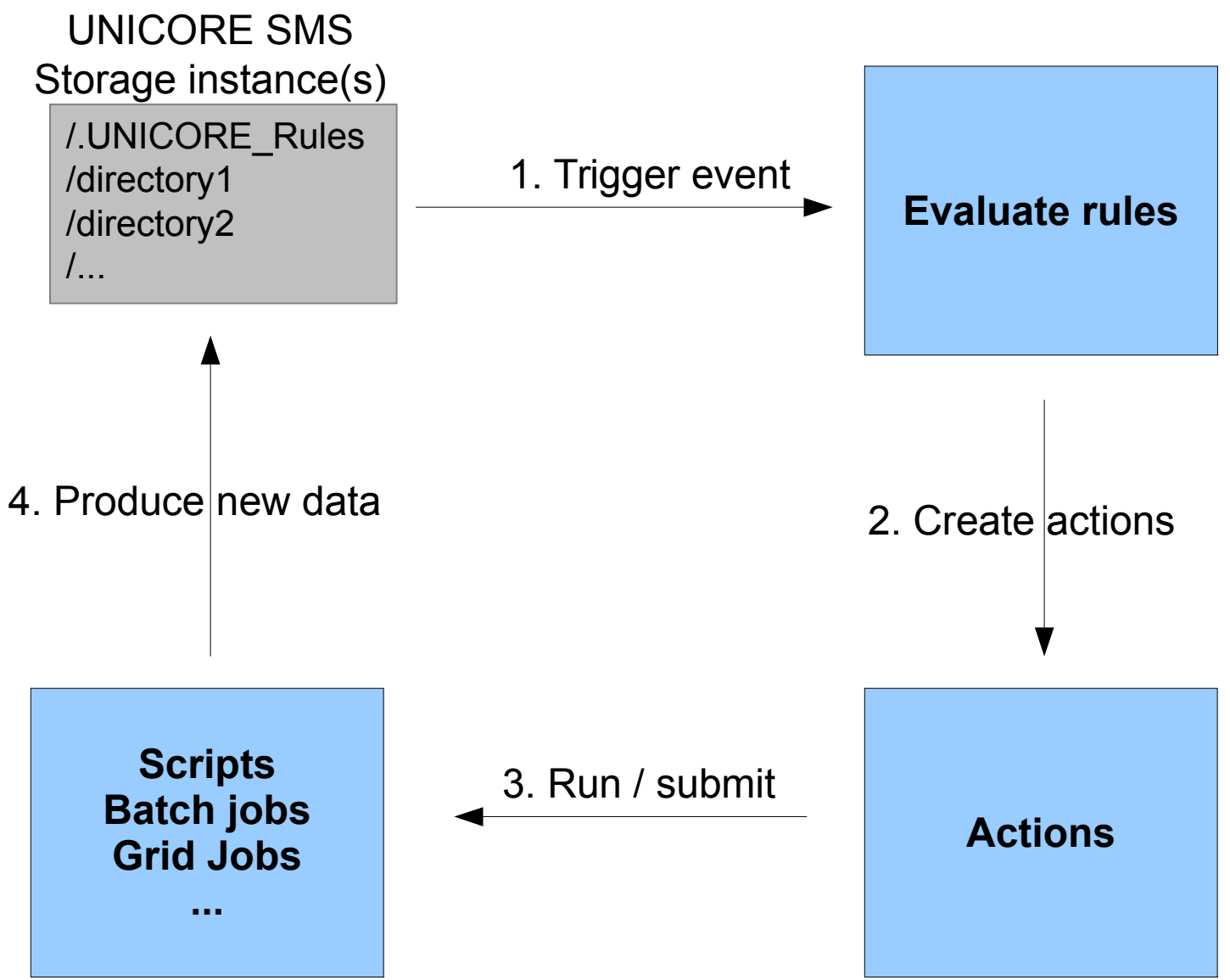
server-to-server copy

SMS

- File systems
- Apache HDFS

- S3

- iRODS (prototype)


UNICORE SMS
Storage instance(s)
/.UNICORE_Rules
/directory1
/directory2
/...
1. Trigger event
Evaluate rules
2. Create actions
Actions
3. Run / submit
Scripts
Batch jobs
Grid Jobs
...
4. Produce new data

Types of triggering events

- Periodic directory scan
  - Files can be written independently of UNICORE
  - Scan interval configurable
  - Directory include/exclude patterns
- (Explicit client invocation)
- (Finished file write(s))

Types of actions

- Local script
  - Executed via XNJS/TSI
  - TSI node (cluster login node)
- Local batch job
  - Executed via XNJS/TSI
  - Compute node(s)
  - UCC-like job description
- Metadata extraction
- TBD: Grid jobs, workflows, …?

Required setup by the end-user

- Create a storage (service instance) where triggering is enabled
  - „ucc connect“
- Configure (edit .UNICORE_Rules file)

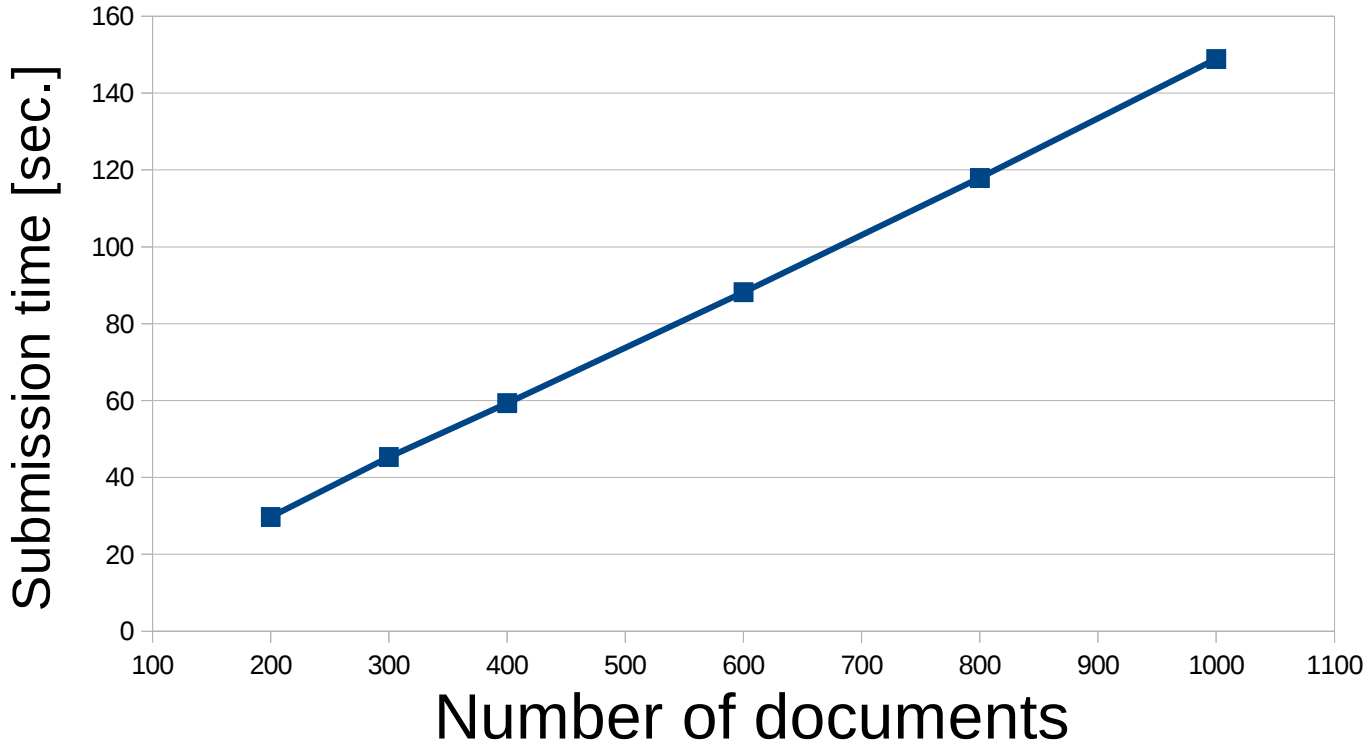
Example

- Goal: calculate checksums (md5) of PDF files in a certain directory using batch jobs
- Rule (job is in UCC syntax!)

```
Name: computeMD5Sum, Match: ".*\\.pdf",
Action: {
    Type: BATCH,
    Job:  {
      Executable: "/usr/bin/md5sum",
      Arguments: ["${UC_FILE_PATH}"],
      Exports: [
        {From: "stdout",
        To: "file://${UC_BASE_DIR}/checksums/${UC_FILE_NAME}.md5"},
      ],
    }
```

Example – some results

- Submission to XNJS
- Create uspace, start processing
- Running on localhost using nobatch TSI



→ **Performance limited by XNJS job acceptance/processing rate**

Outlook / issues to be solved

- Submission of Grid jobs and workflows
  - Security!! (long-running trust delegation required)
  - Need/want to deploy a Grid client (UCC) on the target system?
  - Submit from UNICORE/X?
- Using „shared“ storages
  - Used by more than one user (e.g. a project / Unix group)
  - Very typical setup in real-life
- Need more real-life testing → in progress!

Questions?

- Thanks
  - Jedrzej Rybicki for discussions on this topic