Digital Library Storage using iRODS Data Grids

Mark Hedges, Tobias Blanke
Centre for e-Research, King's College London
Arts and Humanities Data Service
Arts and Humanities e-Science Support Centre

Adil Hasan
Rutherford Appleton Laboratory
Science and Technology Facilities Council

Overview

- Background: AHDS and Centre for e-Research
- Background: data deluge and broader data challenge
- Digital libraries and e-research infrastructures
- Digital libraries and data grids (SRB/iRODS)

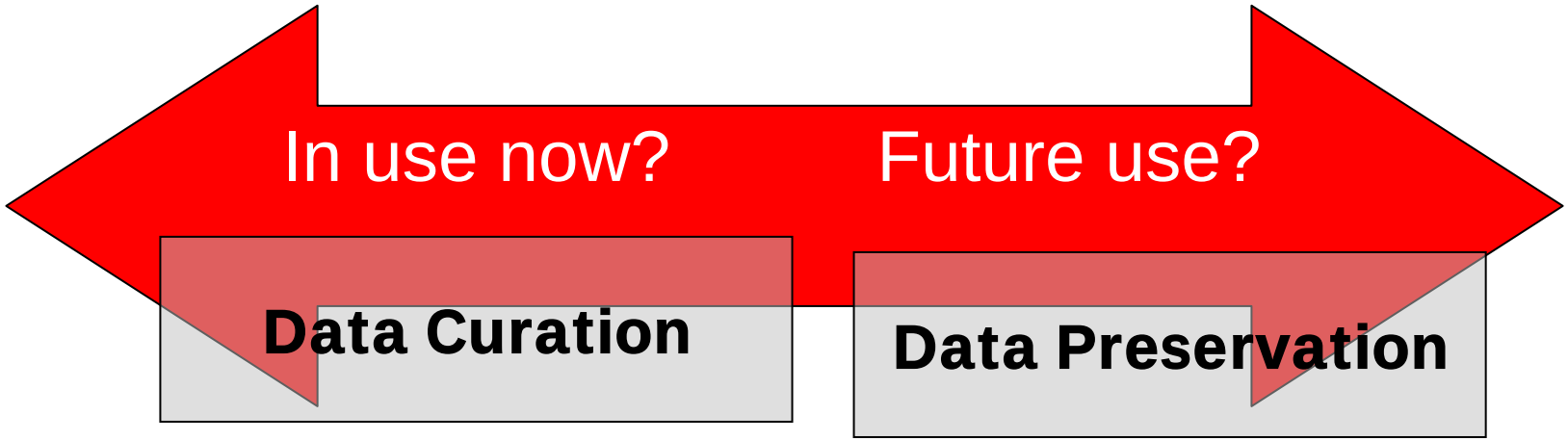
What is/was the AHDS

- Arts and Humanities Data Service
- Established 1996, funded until 2008
- Distributed structure
- Mission: to collect, preserve and distribute digital resources produced by and for arts and humanities research (mainly in the UK)

What is CeRch?

- Centre for e-Research at King's College London
- Established 2007
- Incorporates staff and expertise of AHDS and other groups such as AHeSSC
- Continuity, but change of focus

Research data management



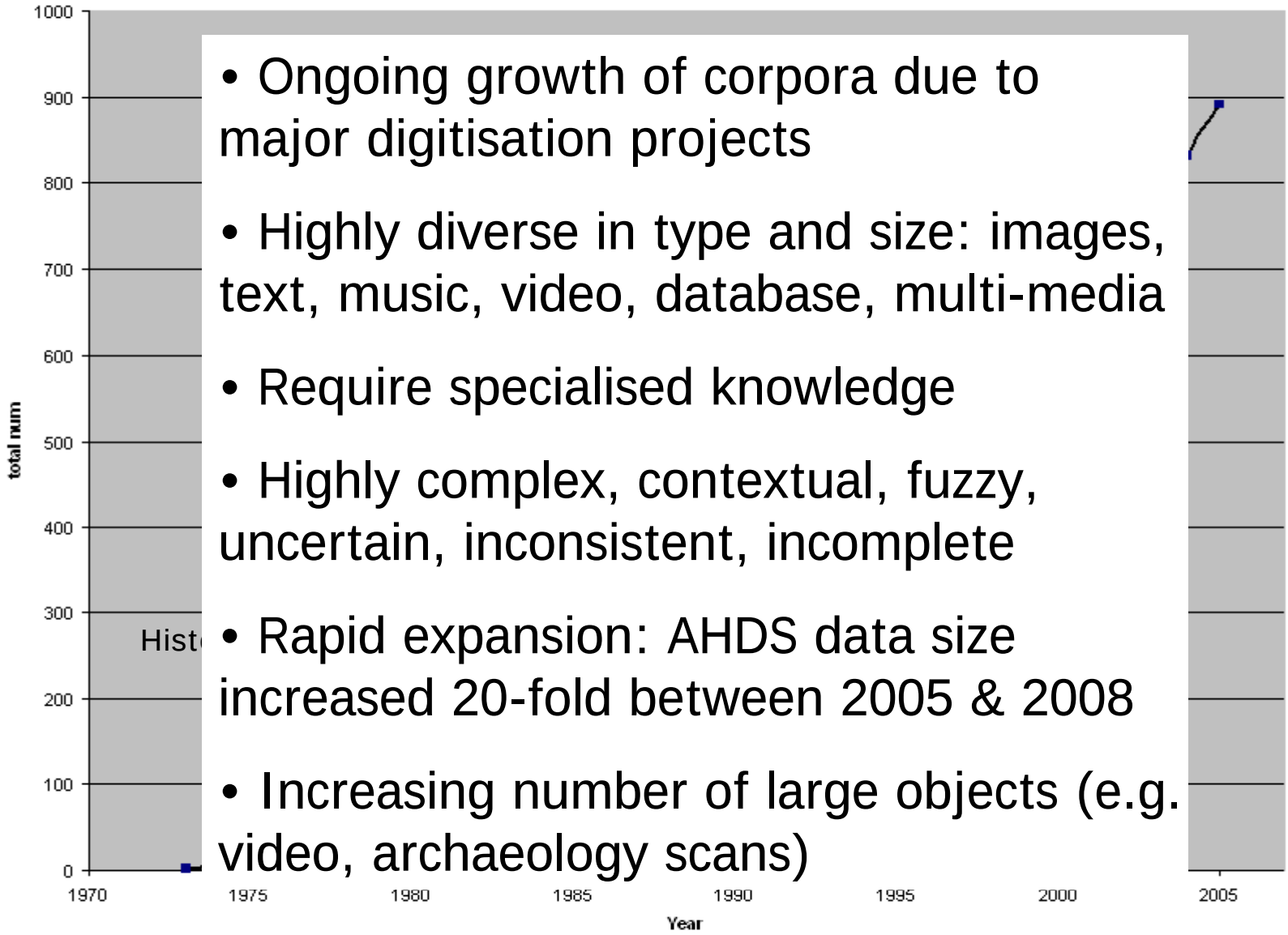
- **Curation:**

The activity of managing and promoting the use of data from its creation to ensure it is fit-for-purpose and remains available for discovery and re-use.

- **Preservation:**

An archiving activity in which data are maintained over time so they can still be accessed and understood through changes in technology

Data Challenge in the Humanities



- Ongoing growth of corpora due to major digitisation projects
- Highly diverse in type and size: images, text, music, video, database, multi-media
- Require specialised knowledge
- Highly complex, contextual, fuzzy, uncertain, inconsistent, incomplete
- Rapid expansion: AHDS data size increased 20-fold between 2005 & 2008
- Increasing number of large objects (e.g. video, archaeology scans)

Digital library systems

- Fedora Commons (at AHDS/CeRch)
- Supports digital resources that are diverse and structurally complex
- Flexible metadata management
- Disseminator framework supporting more complex and application specific processing of digital resources
- Not a stand-alone DL, but a component of an integrated research infrastructure

Issues

- Focuses on support for structure/ complexity rather than storage issues
- Doesn't natively support distribution of data
- Performance limitations when processing large objects

Data Grids

- Storage Resource Broker (SRB), a widely-used data grid technology developed by the San Diego Super Computer Center
- Addresses storage issues for digital repository and preservation environments
- Provides uniform, searchable access to virtualised, distributed resources, so DL is insulated from:
  - physical location of data
  - types of storage
  - migrating to new hardware
- Scalable – as library grows, new resources can be added dynamically
- Auditing facilities

Limitations

- Not open source
- Not easy to exclude unwanted services
- Very effective for storage management, but not integrated with wider infrastructure.
- Not easy to integrate application-specific requirements (either change the core code, or implement in client, or use proxy commands)
- No built-in implementation of workflow (have to script this outside SRB, whether server or client side), or of asynchronous processing.
- Requires choreography between SRB admin and person running workflow.
- Relatively restricted support for metadata extension (Fedora supports but how to integrate)

iRODS

- The open source successor to SRB
- Provides similar data virtualisation
- Rule Engine allows data management policies to defined and realised as rules
- Policy virtualisation – insulation from how policies are implemented
- Execution of rules driven by events
- System level rules have great potential to ‘hide’ required data management operations from user/application level
- Event-condition-action model

What are rules? (1)

- Rules (or policies) are sets of operations that you want to impose on an object (file, user, resource, etc).
  - The operations are called “micro-services”
  - Each micro-service is a C-app that executes and does something (e.g. checksum data, convert a file from one format to another).
  - Micro-services are transactional (recovery operations created for each micro-service).
- In most cases you can define server-side workflow as a rule controlling a set of {micro-services, rules}.

What are rules? (2)

- Rule cast as {event: condition: action set: recovery set:}.
  - Can build rules of rules.
  - Allows you to model complex workflows.
- Supports execution of rules on most convenient resource (usually run on server connect to).
- Supports delayed execution of rules (i.e. “run this rule this evening”).
- Supports periodic execution of rules (i.e. “run this rule every evening”).

iRODS rules

The components of a rule definition are as follows:

actionDef | *condition* | *workflowChain* | *recoveryChain*

Where:

- *actionDef* identifies the action to be carried out
- *condition* is necessary condition for execution
- *workflowChain* is sequence of actions to be executed
- *recoveryChain* is corresponding sequence of recovery actions (to ensure consistent state).

Rule can be built up cumulatively from other rules.

Data passed into/within rules (via parameters/context).

Note: syntax may change in near future.

Example rule - preservation

Executed when an object has been ingested

acPostProcForPut | |
acCheckObjectIntegrity##
acAnalyseObject##
acNormaliseObject##
msiSysReplDataObj(PresRescGrp,all) |
nop##nop##nop##msiCleanUpReplicas

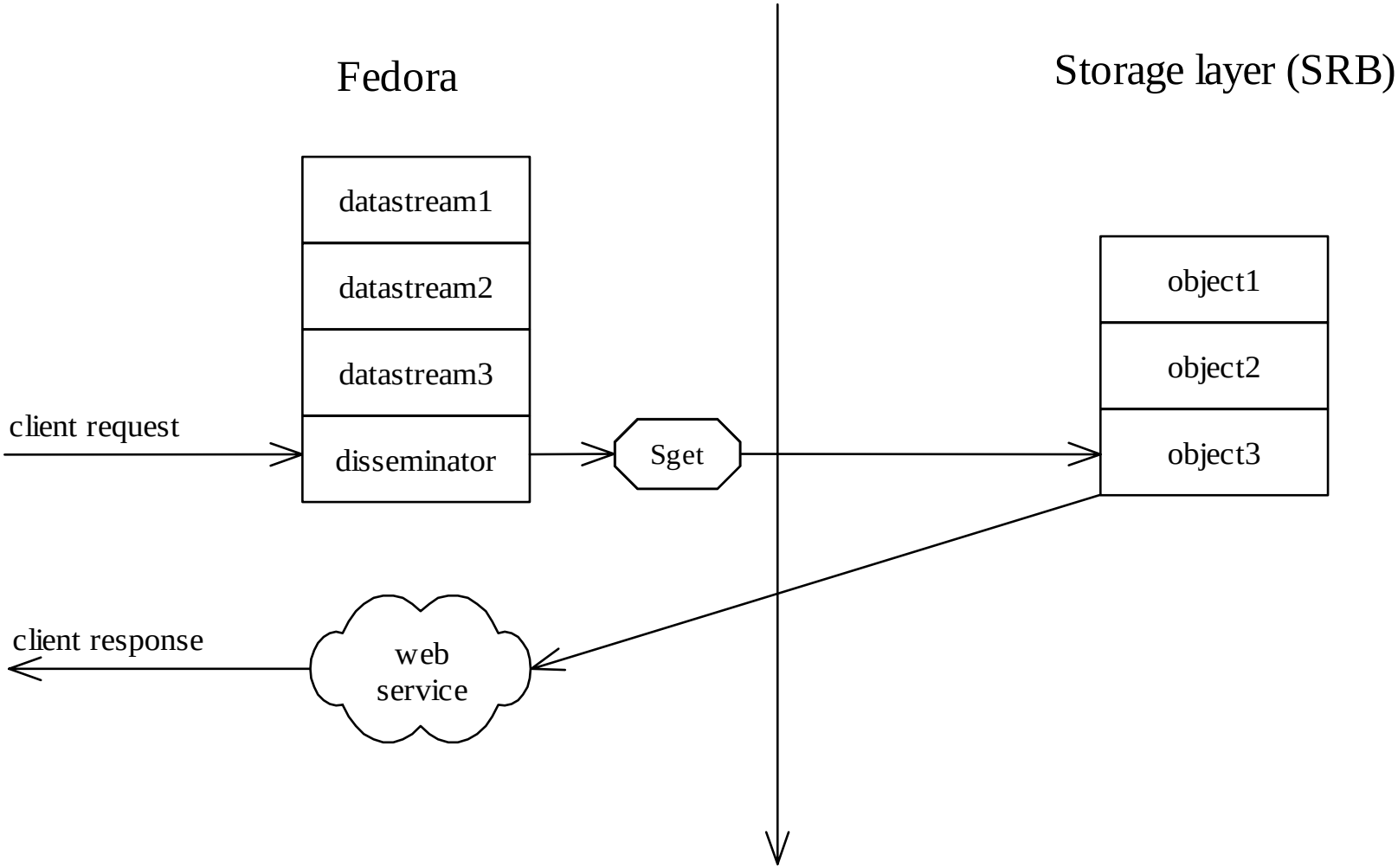
Example rule - application

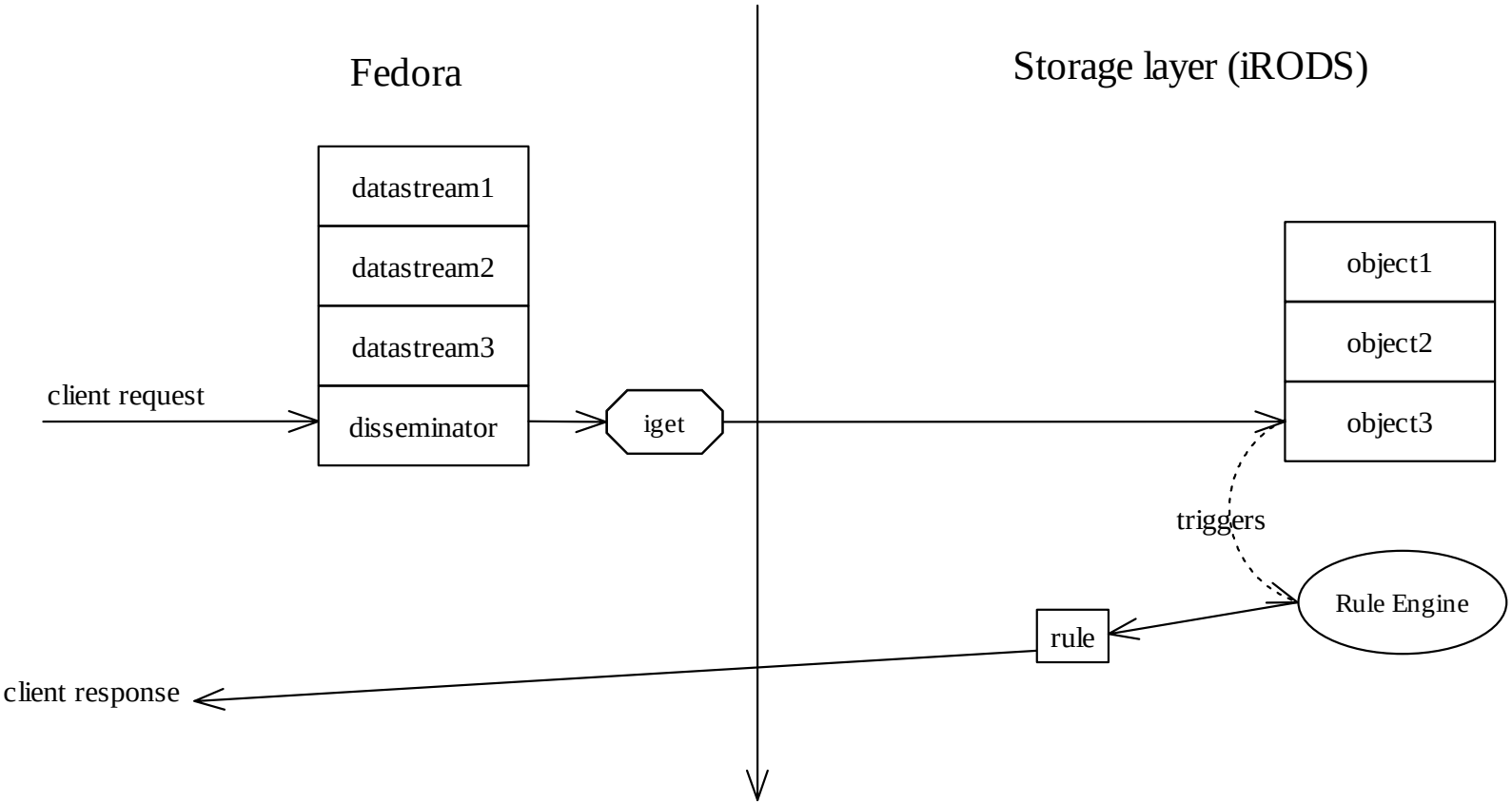
Executed when an object has been ingested

acPostProcForPut |
$format == "image/tiff" &&
$objectcategory="highResMS" |
msiCheckForJPEGTiling##
msiTiffToJPEGTiling##
msiValidateTiffToJPEGTiling |
nop##msiCleanUpJPEGTiling##
msiCleanUpJPEGTiling

Example

- Retrieving large objects for processing
- Retrieving entire object not always necessary, and can be inefficient
- Move the processing to the data
- Disseminators -> rules


Fedora
Storage layer (SRB)
datastream1
datastream2
datastream3
disseminator
client request
Sget
object1
object2
object3
client response
web
service


Fedora
Storage layer (iRODS)
datastream1
datastream2
datastream3
disseminator
client request
iget
object1
object2
object3
triggers
Rule Engine
rule
client response

Next steps/issues

- Prototypes -> production
- Developing more comprehensive set of rules for managing digital objects
- Jobs requiring data from multiple locations
- Dynamic deployment of jobs
- Virtual workspaces

Contacts

mark.hedges at kcl.ac.uk

tobias.blanke at kcl.ac.uk

a.hasan at rl.ac.uk