

Put functionality

Augmenting interoperability across scholarly repositories

20/21 April 2006

Rachel Heery,
UKOLN, University of Bath



UKOLN is supported by: **JISC** 
MUSEUMS LIBRARIES ARCHIVES



www.ukoln.ac.uk

UKOLN a centre of expertise in digital information management

My perspective

- JISC Digital Repositories Programme
 - 20+ repository projects
- JISC Framework
 - Defining reference models and services
- Deposit API meeting and ongoing
 - Intention to find light-weight solution to assist populating repositories within timescales of JISC programme
- NB submit, deposit, post, put...similarities but subtle differences

Put – User requirements

- To enable users to populate repositories effectively
- To capture content from desktop applications, experimental equipment (smart labs), learning content development tools etc
- To enable repositories to exchange data in predictable manner
- To hide complexity from end-user
- To be compatible with follow-on added value services layered on repository content

Put - Scenarios

- Author 'saves' a report from a desktop authoring system to the institutional repository
- An institutional repository submits a learning object to a national learning materials repository
- Experimental data output from spectrometer is 'saved as' a file and a file containing metadata on operational parameters is also generated. A data capture service is invoked and the files pertaining to the experiment are deposited, along with the necessary metadata, in the laboratory repository.

See <http://www.ukoln.ac.uk/repositories/digirep/>

Functional requirements

- Must be generic enough to be offered by wide range of heterogeneous repositories
 - scholarly publications, data, learning objects, images, etc.
- Must accept submission of different digital object types in consistent way:
 - data and/or metadata possibly in the form of complex objects
- Data may be of different types and metadata may be of different formats
- Data may be large scale (scientific datasets)
- Repositories may have different policy rules
- Successful deposit should initiate ingest process for local storage in repository:
 - may be automated and/or involve manual intervention (metadata extraction, quality assurance, identity and provenance processing etc)

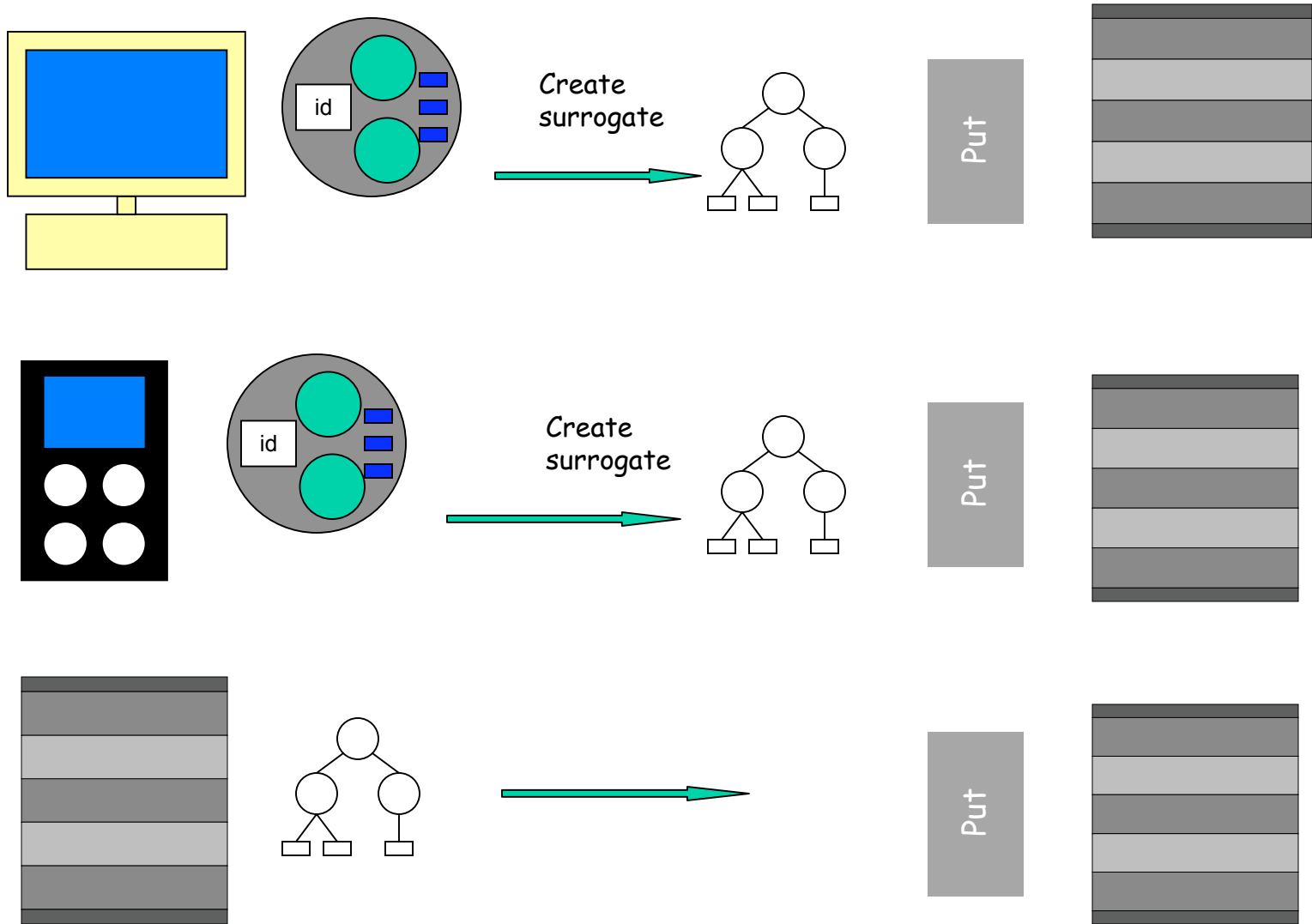
Put – abstract service definition

Put interface: a Repository interface that supports submission of one or more Surrogates into the Repository, thereby facilitating the addition of Digital Objects to the collection of the Repository.

Questions:

- How are Digital Objects submitted in first instance?
 - As ‘attachments’ to surrogate? As separate ‘deposit’ service? Embedded in surrogate?
- Are some digital objects sufficiently different to warrant a separate interface e.g. large scale datasets?

Put scenarios



Put – abstract interface definition

- data in
 - introspection request (“explain”)
 - put request with optional parameters (e.g.digital object ‘semantics’, metadata formats..)
- data out
 - introspection response (“repository policy info”)
 - receipt confirmation and digital object identifier

Next steps

- Specify abstract service
 - Information models and APIs must be developed in manner neutral to implementation binding
- Examine existing protocols and packaging formats to see how far they could implement the abstract service
- Evaluate and decide whether something new required (protocols and/or packaging formats)

Potential bindings for abstract service

Simple approach?

- HTTP with attachment??
- Use XOP to deal with transporting data efficiently??

Examine existing Protocols?

- SRW Update
- OKI OSID
- JSR 170 & 123
- Fedora Deposit API
- XOP
- Atom

Examine existing packaging formats?

- METS, MPEG DIDL, IMS Content Packaging

Issues

- How are Digital Objects submitted in first instance?
 - As ‘attachments’ to surrogate? Embedded in surrogate?
 - As separate ‘deposit’ or ‘obtain’ service?
- Is there requirement for packaging service associated with ‘put’?
 - At what stage is surrogate created/packaged’?
 - Is there requirement for a separate packaging service?
- Boundaries between put and ingest
 - What has already happened at point of ingest? regarding metadata and identifiers
- Data integrity
 - Is there requirement to get back (export) exact object that was deposited?
- Can look up of policy rules be done as a request to service registry?
 - How far is look up of policy rules automated?

Issues (2)

Can levels of interoperability be defined for deposit?

- Level 0: lowest common denominator e.g. no authentication, local identifiers assigned
- Level 1: added layer of constraint e.g. some authorisation, recognised identifier scheme
- Level 2: more complex authorisation process, single default identifier scheme, semantic typing of data etc