Case studies in workflow: Three approaches

THE UNIVERSITY OF HULL

UNIVERSITY of VIRGINIA

SU LAIR Stanford University

FedoraCommons™

The Hydra Project

The University of Hull
The University of Virginia
Stanford University
Fedora Commons

Richard Green, Nathan Piazza, Lynn McRae

Tom Cramer, Tim Sigmon, Ross Wayland

Open Repositories 2009, Atlanta, GA
21st May 2009

“Lightweight workflow” is both an oxymoron and a continual aspiration of many stakeholders in the repository community

Introduction

- Hull, Virginia and Stanford, with Fedora Commons, are collaborating on the Hydra Project
  - Reusable application framework over Fedora to allow rapid deployment of repository-powered applications for wide variety of content types
- Workflow is integral and, to allow easy re-use and extensibility, methods for supporting workflow must be easily adaptable
- Three parallel workflow approaches

Hull, Hydra and BPEL

A short history

- Hull has been developing workflows using BPEL for the last four years
  - (Business Process Execution Language – an open standard)
- Used in conjunction with SOAP Web Services during the RepoMMan and REMAP projects
  - JISC-funded projects 2005-2007 & 2007-2009

Why BPEL?

- In 2005 Hull (and JISC) had an interest in using BPEL within a Service Oriented Architecture
- BPEL (then) available in an Open Source engine from (then) Active Endpoints
- Good fit with Fedora’s (then) SOAP Web Services interface (REST now available too)

Pros and Cons #1

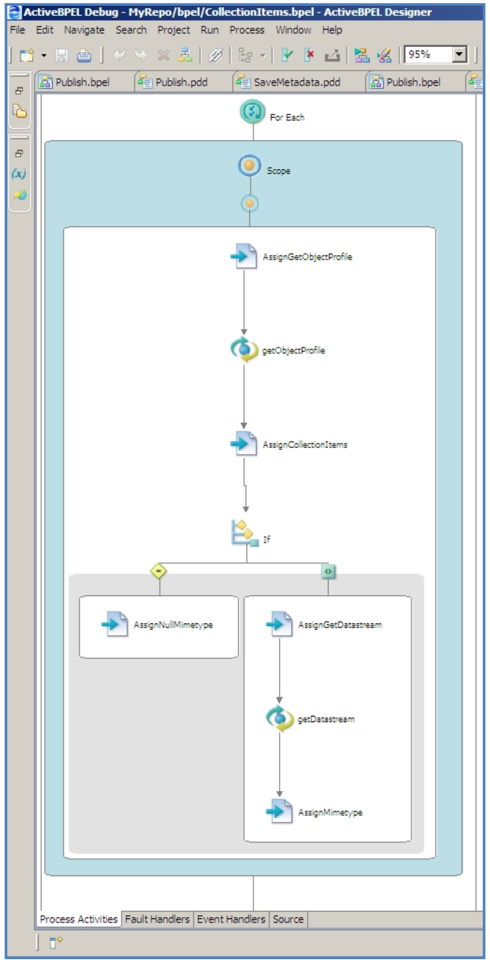
```
<bpel:forEach counterName="counter" parallel="no">
     <bpel:startCounterValue>1</bpel:startCounterValue>

<bpel:finalCounterValue>count($getCollectionItemsResponse/itemList/itemPID)
</bpel:finalCounterValue>
     <bpel:scope>
       <bpel:flow>
         <bpel:links>
           <bpel:link name="L1"/>
           <bpel:link name="L3"/>
           <bpel:link name="L2"/>
         </bpel:links>
         <bpel:assign name="AssignGetObjectProfile">
           <bpel:sources>
             <bpel:source linkName="L1"/>
           </bpel:sources>
           <bpel:copy>
             <bpel:from variable="getCollectionItemsResponse">
               <bpel:query>itemList[ $counter ]/itemPID</bpel:query>
             </bpel:from>
             <bpel:to variable="getObjectProfile">
               <bpel:query>pid</bpel:query>
             </bpel:to>
           </bpel:copy>
           <bpel:copy>
             <bpel:from>''</bpel:from>
             <bpel:to variable="getObjectProfile">
               <bpel:query>asOfDateTime</bpel:query>
             </bpel:to>
           </bpel:copy>
         </bpel:assign>
         <bpel:assign name="AssignCollectionItems">
           <bpel:targets>
             <bpel:target linkName="L2"/>
           </bpel:targets>
           <bpel:sources>
             <bpel:source linkName="L3"/>
           </bpel:sources>
           <bpel:copy>
          <bpel:from variable="getCollectionItemsResponse">
               <bpel:query>itemList[$counter]/itemPID</bpel:query>
             </bpel:from>
             <bpel:to variable="collectionItemsResponse">
               <bpel:query>
                  types:itemsRef[$counter]/types:objectPID
               </bpel:query>
             </bpel:to>
           </bpel:copy>
           <bpel:copy>
             <bpel:from variable="getCollectionItemsResponse">
               <bpel:query>itemList[$counter]/isCollection</bpel:query>
             </bpel:from>
             <bpel:to variable="collectionItemsResponse">
```

```
                 <bpel:query>
                    types:itemsRef[$counter]/types:isCollection
                 </bpel:query>
               </bpel:to>
             </bpel:copy>
             <bpel:copy>
               <bpel:from variable="getObjectProfileResponse">
                 <bpel:query>objectProfile/objLabel</bpel:query>
               </bpel:from>
               <bpel:to variable="collectionItemsResponse">
                 <bpel:query>types:itemsRef[$counter]/types:label</bpel:query>
               </bpel:to>
             </bpel:copy>
             <bpel:copy>
               <bpel:from variable="getObjectProfileResponse">
                 <bpel:query>objectProfile/objLastModDate</bpel:query>
               </bpel:from>
               <bpel:to variable="collectionItemsResponse“>
                  <bpel:query>
                    types:itemsRef[$counter]/types:lastModified
                 </bpel:query>
               </bpel:to>
             </bpel:copy>
           </bpel:assign>
           <bpel:invoke inputVariable="getObjectProfile"
name="getObjectProfile" operation="getObjectProfile"
outputVariable="getObjectProfileResponse" partnerLink="FedoraAccessLT">
             <bpel:targets>
               <bpel:target linkName="L1"/>
             </bpel:targets>
             <bpel:sources>
               <bpel:source linkName="L2"/>
             </bpel:sources>
           </bpel:invoke>
           <bpel:if>
             <bpel:targets>
               <bpel:target linkName="L3"/>
             </bpel:targets>
             <bpel:condition>
contains(string($getCollectionItemsResponse/itemList[$counter]/isCollection),
'true')
             </bpel:condition>
             <bpel:assign name="AssignNullMimetype">
               <bpel:copy>
                 <bpel:from>''</bpel:from>
                 <bpel:to variable="collectionItemsResponse">
                   <bpel:query>
                     types:itemsRef[$counter]/types:mimeType
                  </bpel:query>
                 </bpel:to>
               </bpel:copy>
             </bpel:assign>
             <bpel:else>
               <bpel:flow>
                 <bpel:links>
                   <bpel:link name="L4"/>
                   <bpel:link name="L5"/>
                 </bpel:links>
```

```
<bpel:assign name="AssignGetDatastream">
                  <bpel:sources>
                    <bpel:source linkName="L4"/>
                  </bpel:sources>
                  <bpel:copy>
                    <bpel:from variable="getCollectionItemsResponse">
                      <bpel:query>itemList[$counter]/itemPID</bpel:query>
                    </bpel:from>
                    <bpel:to variable="getDatastream">
                      <bpel:query>pid</bpel:query>
                    </bpel:to>
                  </bpel:copy>
                  <bpel:copy>
                    <bpel:from>
                      <bpel:literal>file</bpel:literal>
                    </bpel:from>
                    <bpel:to variable="getDatastream">
                      <bpel:query>dsID</bpel:query>
                    </bpel:to>
                  </bpel:copy>
                  <bpel:copy>
                    <bpel:from>''</bpel:from>
                    <bpel:to variable="getDatastream">
                      <bpel:query>asOfDateTime</bpel:query>
                    </bpel:to>
                  </bpel:copy>
                </bpel:assign>
                <bpel:invoke inputVariable="getDatastream"
name="getDatastream" operation="getDatastream"
outputVariable="getDatastreamResponse"
partnerLink="FedoraManagementLT">
                  <bpel:targets>
                    <bpel:target linkName="L4"/>
                  </bpel:targets>
                  <bpel:sources>
                    <bpel:source linkName="L5"/>
                  </bpel:sources>
                </bpel:invoke>
                <bpel:assign name="AssignMimetype">
                  <bpel:targets>
                    <bpel:target linkName="L5"/>
                  </bpel:targets>
                  <bpel:copy>
                    <bpel:from variable="getDatastreamResponse">
                      <bpel:query>datastream/MIMEType</bpel:query>
                    </bpel:from>
                    <bpel:to variable="collectionItemsResponse">
                      <bpel:query>
                        types:itemsRef[$counter]/types:mimeType
                     </bpel:query>
                    </bpel:to>
                  </bpel:copy>
                </bpel:assign>
              </bpel:flow>
            </bpel:else>
          </bpel:if>
        </bpel:flow>
      </bpel:scope>
    </bpel:forEach>
```

Pros and Cons #2

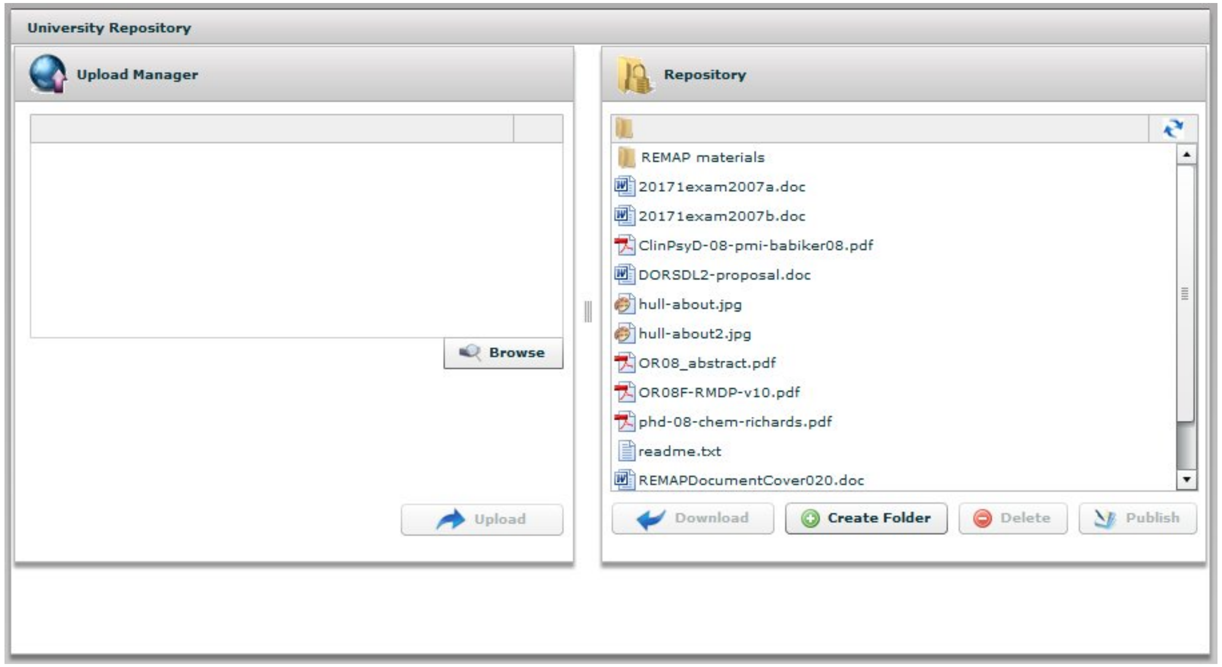


- Cons: verbose, fiddly, syntactically demanding, soul destroying, ….
- Pros (given a good graphical design interface): powerful, flexible, relatively quick to “write”, test and edit…
- Each node in the tree is an ‘activity’ (for each, assign, get, if, etc) for which you provide the parameters
    - Note: the ‘for each’ loop depicted here results in the code on the previous slide

The REMAP tool

- The REMAP tool (son of RepoMMan) uses BPEL-orchestrated Web Services to allow a user to interact with the institutional repository
- Each component Web Service can be used and re-used in multiple contexts given appropriate granularity

REMAP #2

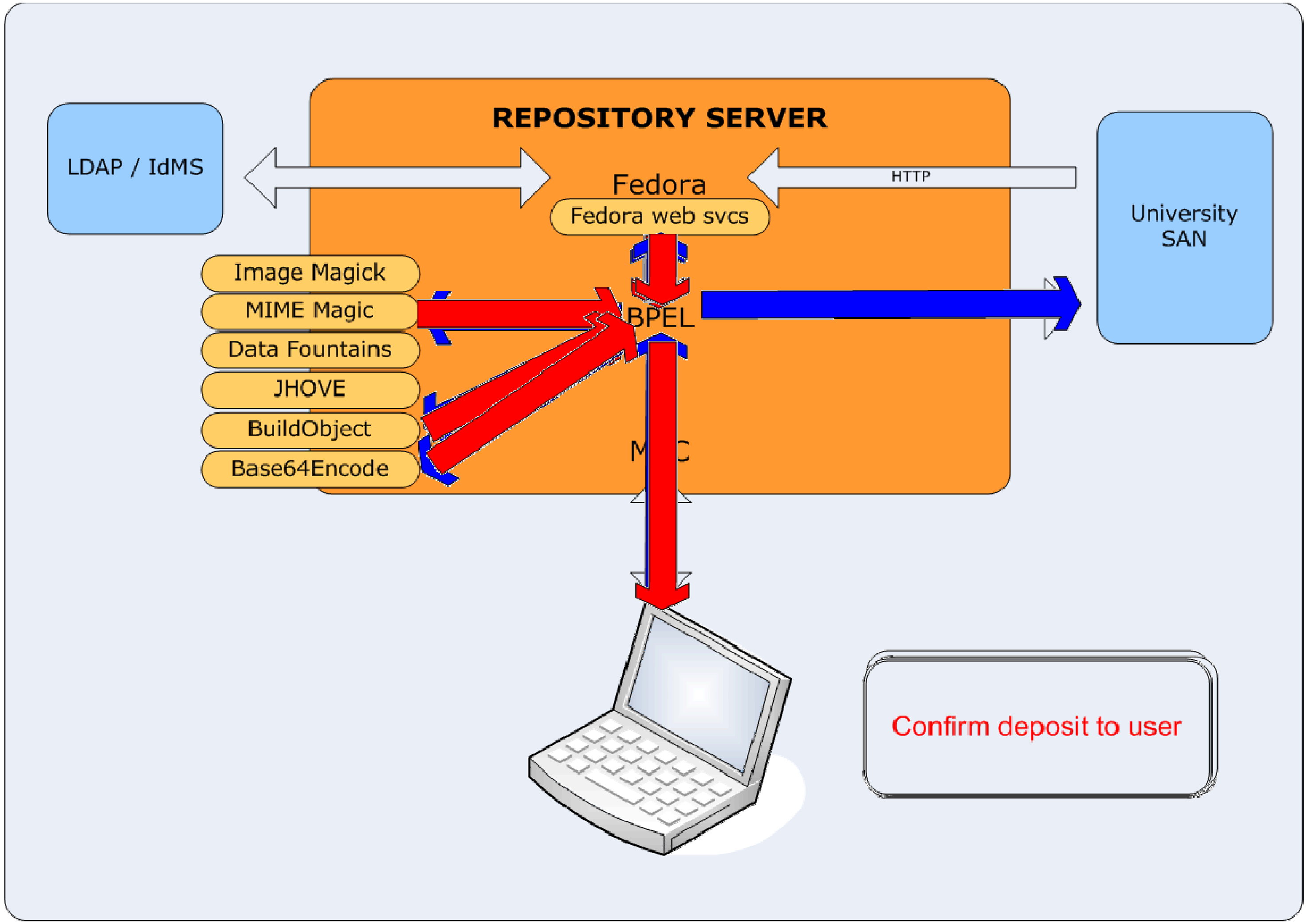


REMAP materials
20171exam2007a.doc
20171exam2007b.doc
ClinPsyD-08-pmi-babiker08.pdf
DORSDL2-proposal.doc
hull-about.jpg
hull-about2.jpg
OR08_abstract.pdf
OR08F-RMDP-v10.pdf
phd-08-chem-richards.pdf
readme.txt
REMAPDocumentCover020.doc

- Consider a user copying a file from their computer to their private repository space

- They browse their computer at the left and upload the file to their repository space, represented at the right. Lots of stages (Web Services) involved ‘under the lid’

REMAP #3



REPOSITORY SERVER

LDAP / IdMS

Fedora

Fedora web svcs

HTTP

University SAN

Image Magick

MIME Magic

Data Fountains

JHOVE

BuildObject

Base64Encode

BPEL

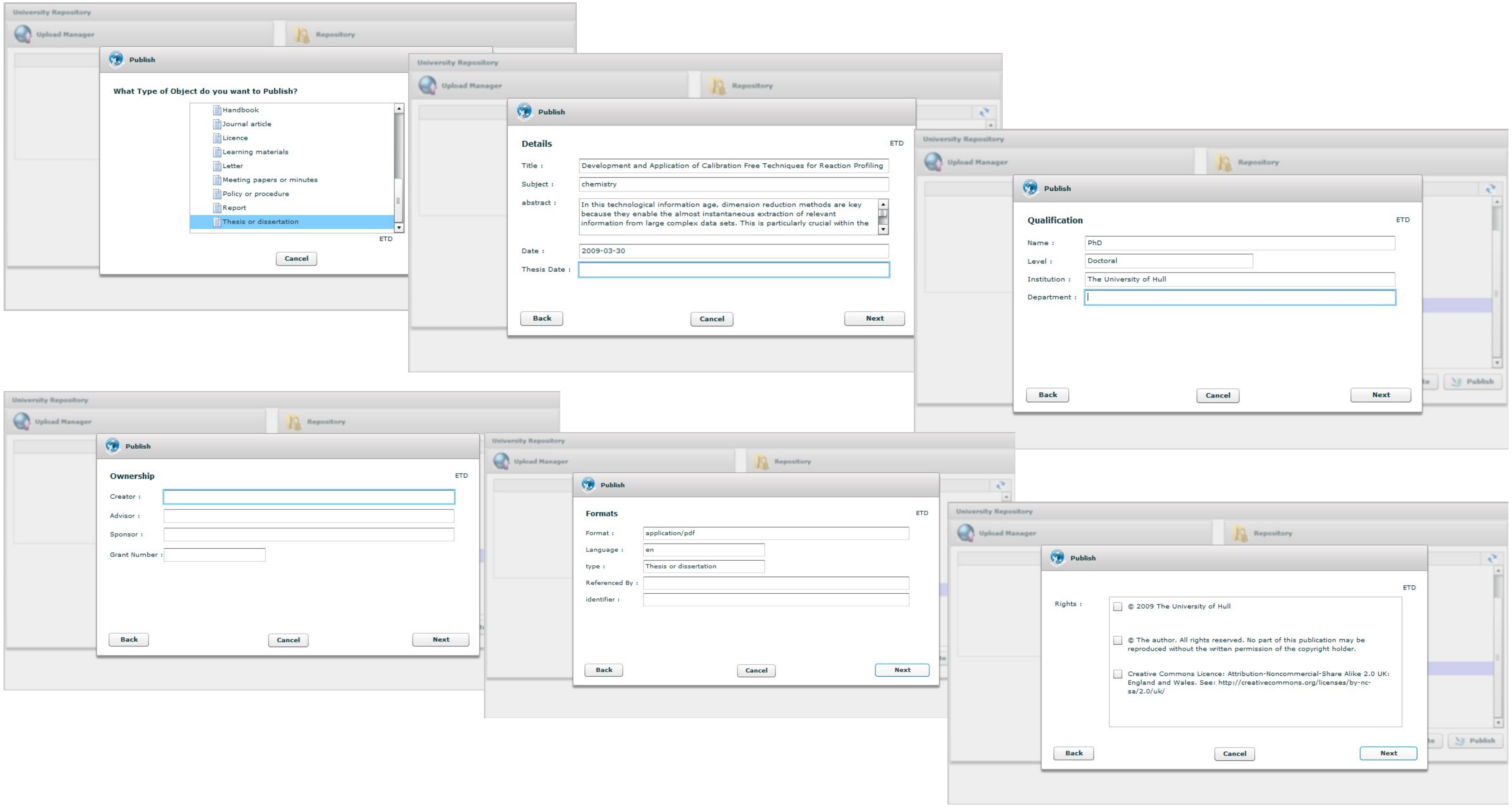
M C

Confirm deposit to user

REMAP #4

- The user can (optionally) publish a file to the institutional repository. The tool provides a context sensitive wizard.
- The process is moderated through an accession queue.
- Take the example of a thesis (ETD)

Publishing “my” thesis



What Type of Object do you want to Publish?

- Handbook
- Journal article
- Licence
- Learning materials
- Letter
- Meeting papers or minutes
- Policy or procedure
- Report
- Thesis or dissertation

ETD

Details ETD

Title : Development and Application of Calibration Free Techniques for Reaction Profiling

Subject : chemistry

abstract : In this technological information age, dimension reduction methods are key because they enable the almost instantaneous extraction of relevant information from large complex data sets. This is particularly crucial within the

Date : 2009-03-30

Thesis Date :

Qualification ETD

Name : PhD

Level : Doctoral

Institution : The University of Hull

Department :

Ownership ETD

Creator :

Advisor :

Sponsor :

Grant Number :

Formats ETD

Format : application/pdf

Language : en

type : Thesis or dissertation

Referenced By :

identifier :

ETD

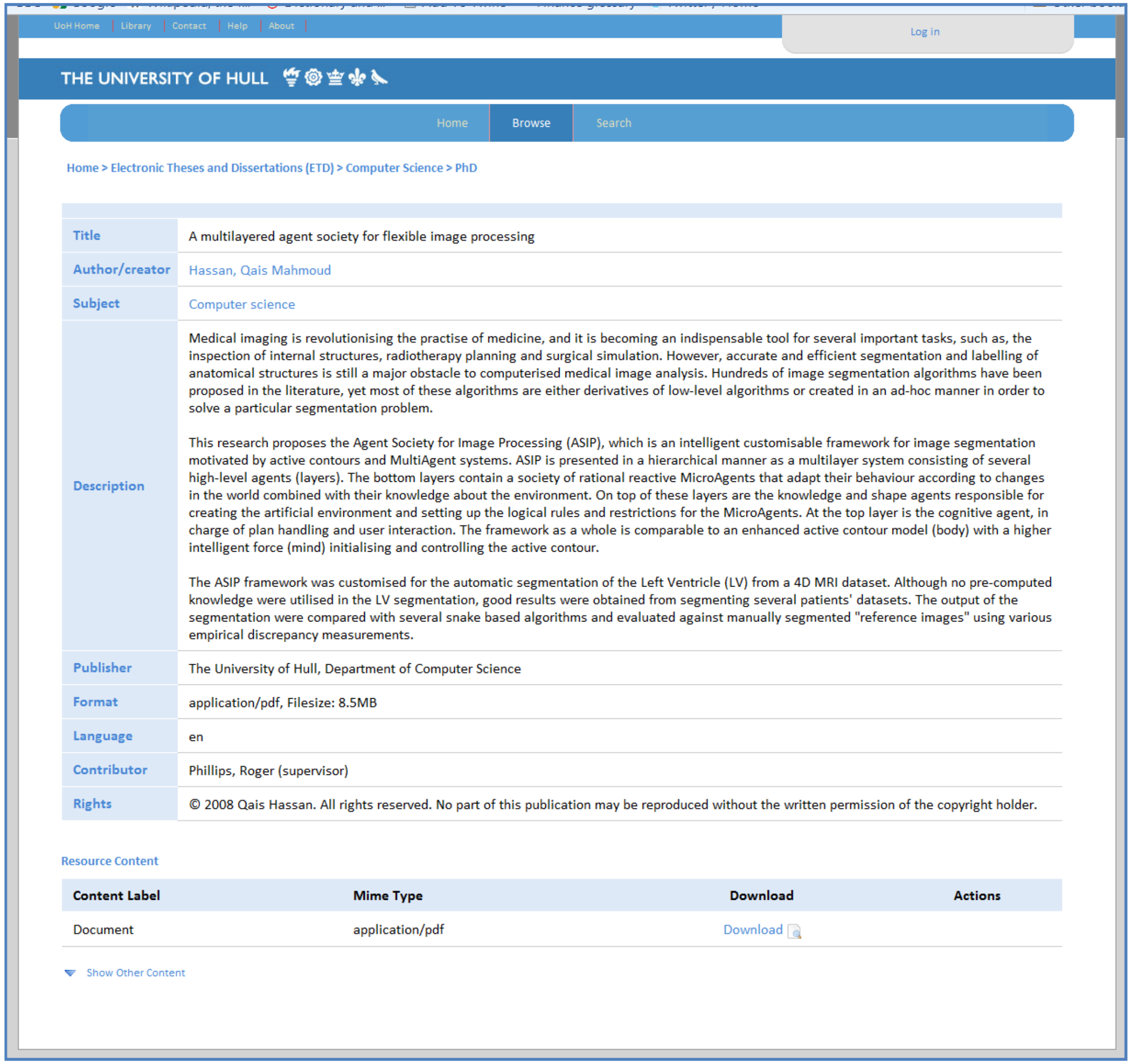
Rights :

☐ © 2009 The University of Hull

☐ © The author. All rights reserved. No part of this publication may be reproduced without the written permission of the copyright holder.

☐ Creative Commons Licence: Attribution-Noncommercial-Share Alike 2.0 UK: England and Wales. See: http://creativecommons.org/licenses/by-nc-sa/2.0/uk/

ETD in the repository



Home > Electronic Theses and Dissertations (ETD) > Computer Science > PhD

Title	A multilayered agent society for flexible image processing
Author/creator	Hassan, Qais Mahmoud
Subject	Computer science
Description	Medical imaging is revolutionising the practise of medicine, and it is becoming an indispensable tool for several important tasks, such as, the inspection of internal structures, radiotherapy planning and surgical simulation. However, accurate and efficient segmentation and labelling of anatomical structures is still a major obstacle to computerised medical image analysis. Hundreds of image segmentation algorithms have been proposed in the literature, yet most of these algorithms are either derivatives of low-level algorithms or created in an ad-hoc manner in order to solve a particular segmentation problem. This research proposes the Agent Society for Image Processing (ASIP), which is an intelligent customisable framework for image segmentation motivated by active contours and MultiAgent systems. ASIP is presented in a hierarchical manner as a multilayer system consisting of several high-level agents (layers). The bottom layers contain a society of rational reactive MicroAgents that adapt their behaviour according to changes in the world combined with their knowledge about the environment. On top of these layers are the knowledge and shape agents responsible for creating the artificial environment and setting up the logical rules and restrictions for the MicroAgents. At the top layer is the cognitive agent, in charge of plan handling and user interaction. The framework as a whole is comparable to an enhanced active contour model (body) with a higher intelligent force (mind) initialising and controlling the active contour. The ASIP framework was customised for the automatic segmentation of the Left Ventricle (LV) from a 4D MRI dataset. Although no pre-computed knowledge were utilised in the LV segmentation, good results were obtained from segmenting several patients' datasets. The output of the segmentation were compared with several snake based algorithms and evaluated against manually segmented "reference images" using various empirical discrepancy measurements.
Publisher	The University of Hull, Department of Computer Science
Format	application/pdf, Filesize: 8.5MB
Language	en
Contributor	Phillips, Roger (supervisor)
Rights	© 2008 Qais Hassan. All rights reserved. No part of this publication may be reproduced without the written permission of the copyright holder.

Resource Content

Content Label	Mime Type	Download	Actions
Document	application/pdf	Download	

Show Other Content

- Repository object has been given RMDP tags to help management and potential preservation

- Metadata conversion has taken place – all BPEL and Web Services

Workflow

- This has described one workflow. Hydra will allow non-expert users to
  - Reconfigure existing workflows
  - Build other workflows (Templates?) using a ‘Lego set’ of Web Services provided
  - Choice of orchestration method
- Hull pursuing BPEL for now although the Active Endpoints Open Source BPEL engine is no longer being developed by Active VOS

Virginia and Hydra: Community-Driven Workflow and Staying RESTful

Background

- Virginia has less developed workflow implementations than Hull or Stanford
- Still in the process of learning exactly what our workflow needs are/will be
- A culture of RESTfulness (Blacklight)
- The “Million Manuscript March” as key usecase
- A decentralized community

SOA Workflow Assumptions

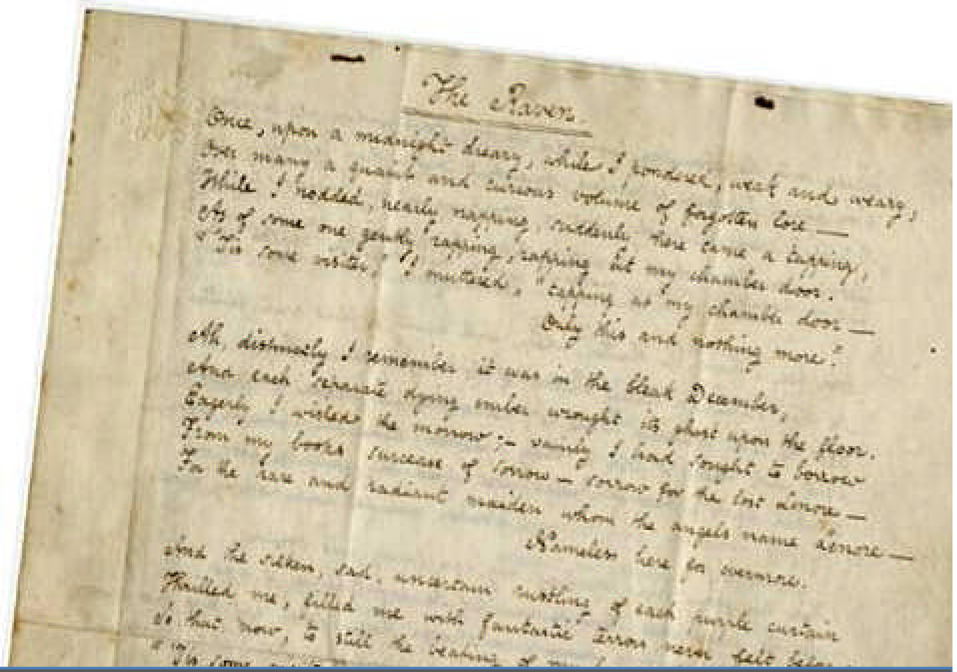
- Hierarchical (business) management structure with top-down power to mandate IT policy
- Widespread and consistent programmer skillsets
- Many distributed machines/systems
- Use cases with complex business procedures requiring formal signoff/many human hands

By Contrast: UVa's Situation

- Distributed management structure: cooperation between academic and administrative units on IT initiatives is a fresh proposition every day
- Variable programmer skillsets from unit to unit and department to department
- A handful of systems everyone wants to talk to
- Use cases where a handful of people are doing the same informal tasks every day, focused mostly on their own needs

Usecase-Driven Workflow: Manuscript Digitization

- Many small projects in one library
- Many different text processing procedures and metadata schemes
- A genuine "community" of users, focused primarily on internal project needs
- Few manuscript-processing procedures exposed as web services



The Raven

Once, upon a midnight dreary, while I pondered, weak and weary,
Over many a quaint and curious volume of forgotten lore —
While I nodded, nearly napping, suddenly there came a tapping,
As of some one gently rapping, rapping at my chamber door.
"'Tis some visiter," I muttered, "tapping at my chamber door —
Only this and nothing more."

Ah, distinctly I remember it was in the bleak December,
And each separate dying ember wrought its ghost upon the floor.
Eagerly I wished the morrow; — vainly I had sought to borrow
From my books surcease of sorrow — sorrow for the lost Lenore —
For the rare and radiant maiden whom the angels name Lenore —
Nameless here for evermore.

And the silken, sad, uncertain rustling of each purple curtain
Thrilled me, filled me with fantastic terrors never felt before;
So that now, to still the beating of
"'Tis some

REST: A Better Fit for UVa?

- What the web was designed to do: supporting communities with varying skillsets, timetables, and priorities in need of ad hoc publishing with a few low-level standards
- REST tries to preserve the webbiness of the web
- REST emphasizes system independence

A Little REST Partisanship

The relationship between REST and SOAP/WSDL is similar to that between XML and SGML. XML was prescriptive: "you must use Unicode." SGML was descriptive: "you may use any character set but you must declare it." XML: "you must use URIs for identifiers." SGML: "You may use any sort of identifier (filename, database key, etc.)."

SOAP advocates say: "We want to work with you. Tell us what you need added to SOAP/WSDL and we will add it." But actually what REST advocates want is not more but less.

- Paul Prescod, REST Advocate, author of “The XML Handbook” (http://www.prescod.net/rest/rest_vs_soap_overview/)

REST as Workflow Minimalism and Gradualism

- REST thus implies 'less is more' where workflow is concerned
- Having a workflow engine won't make your community agree to use a lot of application-level standards; you can only encourage them to grow towards that by dipping a toe into the services waters
- Specific usecases should drive complexity of implementation
- Start with GET POST PUT and DELETE, and see how it squares with the 80/20 rule
- See service exposure evangelism as a key part of what we do; the level of service exposure may justify more investment in SOA approaches down the line
- Entice project stakeholders to use centralized services where needs are common and development resources are available

REST Drawbacks

- Where business processes are both genuinely complex and distributed enough to justify a full-blown “web programming language”, SOA is a much better fit.
- There are almost no off-the-shelf REST workflow solutions. If what's needed is a turnkey application, SOA is a much better fit.
- REST by its very nature implies gradualism and community. If your goal is transform your centrally-managed institution into a web services juggernaut overnight, using a crack team of muscular programmers to expose everyone's data at once, SOA is the better fit.
- SOA is basically web-based remote invocation. If EJB and CORBA have been essential to your institution, REST may feel like an alien paradigm.

REST Workflow Beyond CRUD

- Create (Post), Read (Get), Update (Put), Delete (Delete) is the core paradigm of the REST philosophy, but that's not the end of the story
- Cookies have been used to add statefulness to the web for years
- WS-CDL (Web Services Choreography Description Language) exists to do more complex kinds of interoperability, but so far the need for that complexity in the REST community appears yet to emerge
- Distributed transactions seem like one of the commonest use cases requiring some of the messaging and execution control capabilities of a language like BPEL. But there is no reason you could not implement your own RESTful messaging mini-protocol to suit these needs.

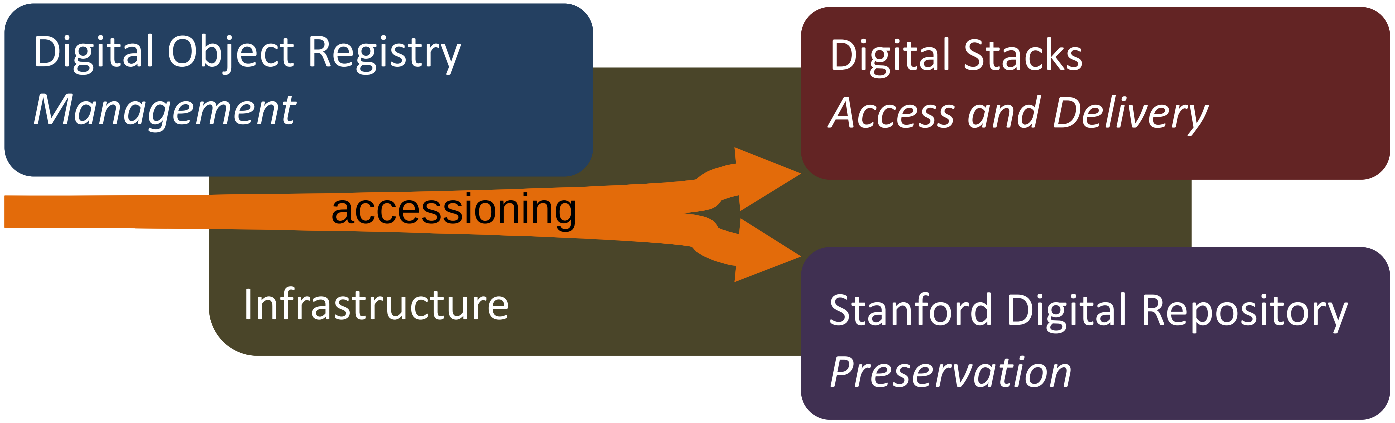
By and large, the ease and flexibility of CRUD for workflow is vastly underestimated. For an example, see: "How to GET a Cup of Coffee"

(http://www.infoq.com/articles/webber-rest-workflow)

Hydra and Stanford Workflow

Stanford Accessioning

- A Digital Object Registry (DOR) provides full object management from the moment an item is acquired
  - Built with Fedora
  - Support object deposit, conversion, metadata enrichment, derivatives, packaging, tracking, etc.
  - Prepares resources for *Access and Delivery* and *Preservation* environments



~~Workflow~~ WorkDo

- A classic need for workflow?
- The work required to "ready" a resource is described as a set of conditions that must be met -- *"get descriptive metadata"*, *"validate files"*, *"generate METS"*, etc.
- Wanted a simple, lightweight approach to getting objects prepped and assembled
- Focus should be on *what needs to be done*, not the process that gets you there

How WorkDo works

A workflow datastream in each object describes processing requirements and status

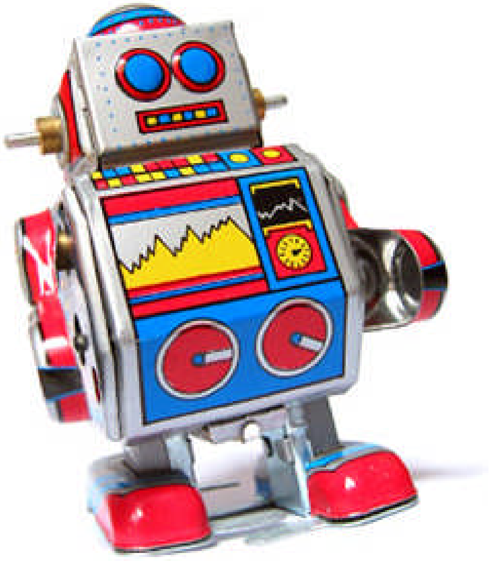
```
<workflow id=“googleScannedBookWF" status="active” …>
   <process name="register-object" status="completed” attempts="1" />
   <process name="desc-metadata" status="completed” attempts="1" />
   <process name="google-convert" status="completed” attempts="1" />
   <process name="google-download" status="exception”
         message="Item for barcode 0339518 not found" attempts="3" />
   <process name="create-pages" status="waiting” attempts="0" />
   <process name="ingest" status="waiting” attempts="0" />
   <process name="shelve" status="waiting” attempts="0" />
   <process name="cleanup" status="waiting” attempts="0" />
</workflow>
```

How WorkDo works

- Each condition = a task to be performed
  - Simple scripts for automated tasks
  - Web UI interactions for human tasks
- Tasks can often be done in parallel
- Simple pre-requisite conditions support dependencies between tasks, e.g.,
  - *“you can’t archive the object before the page files are processed”*
  - *”you can’t submit the Dissertation before the files are uploaded”*

Scripted tasks – Robots!

- A robot is a simple script assign to a task
- Autonomous, like robots on an assembly line
- A typical robot …
  - performs a task -- simplest robots mainly coordinate infrastructure service calls
  - creates or updates relevant DOR/Fedora objects and datastreams
  - updates workflow process status on completion of task



Workflow Services

Query workflow – a query for items with a *waiting* status yields “queues” of work to be done

GET https://dor.stanford.edu/workflow_queue?*[query]*

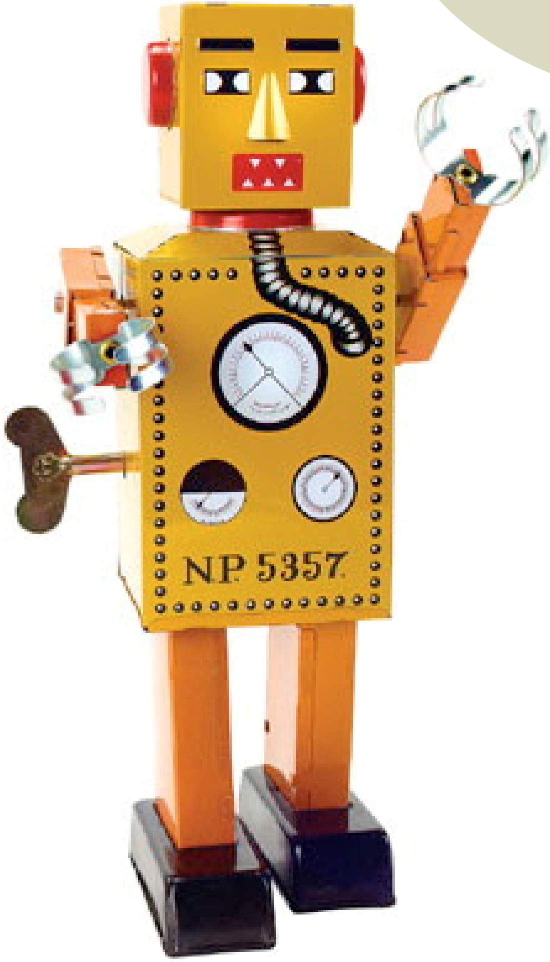
Initiate workflow – Adds workflow datastream to specified object

PUT https://dor.stanford.edu/objects/{id}/workflows/{workflow}

Update workflow – Updates status for a workflow step

PUT https://dor.stanford.edu/objects/{druid}/workflows/{workflow}/{process}

[show admin page here]



Working within the object

- Leverages data placed in the object itself:
  - The object itself can be asked about the status of workflow processes
  - Workflow state is indexed (SOLR) alongside other processing information
  - Provide ongoing management information about the flow of objects through the system
  - They can be exposed as facets in an administrative discovery environment

Cons

- It does not have all the capabilities of a fully featured workflow system, e.g.,
  - It is associated with specific set of objects so could not coordinate work across environments
  - Fits a certain sized "lifecycle" unit of work; not suited for controlling many small processes
  - It does not support very complex or highly dynamic workflows
- Need to evolve this solution as needed

Pros

- The integration of the workflow data with the object has been effective in satisfying the informational and processing needs of our digital resource management
- Lightweight? Does not require external rule or state engines, messaging, or separate process orchestration software
- Was quick and easy to implement
- Can evolve this solution only as needed
- We got robots!

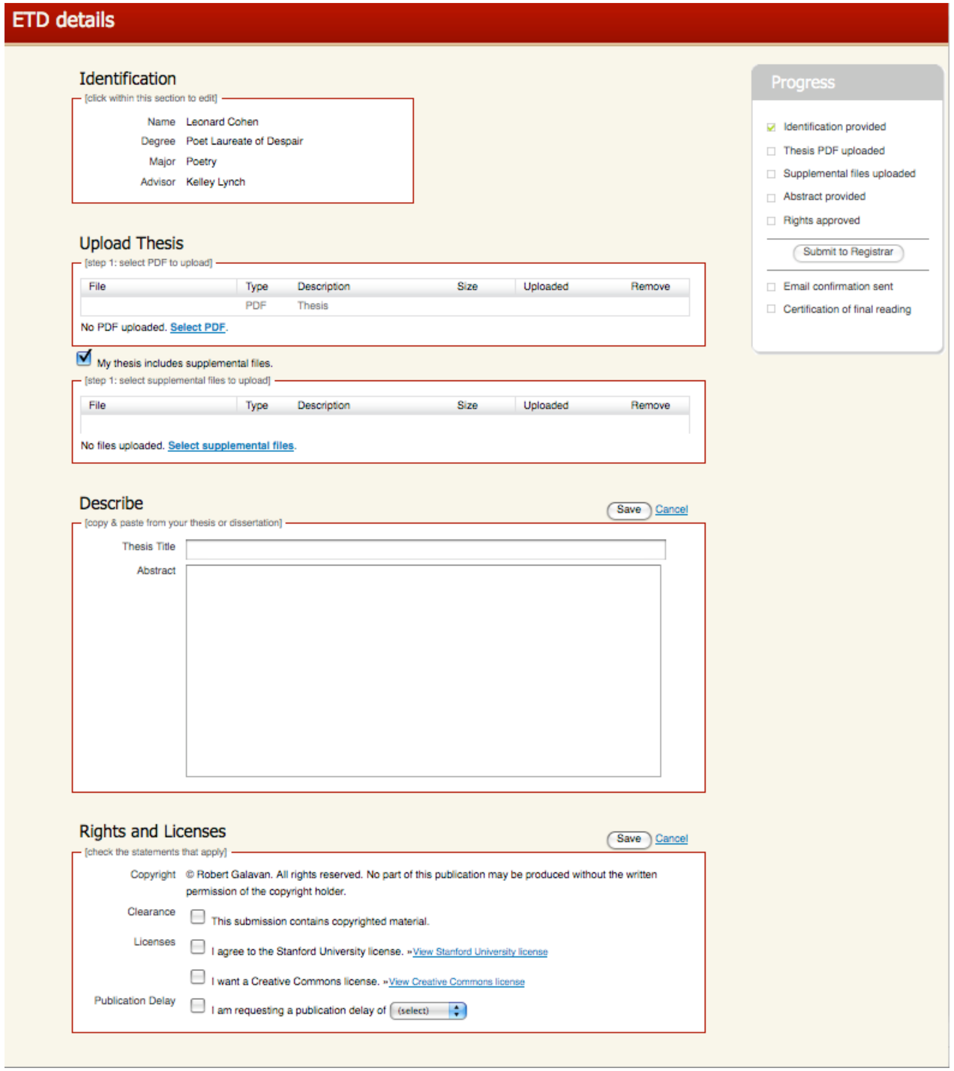
ETD Submission

Who/What

Upload

Describe

Rights



ETD details

Identification

[click within this section to edit]

Name	Leonard Cohen
Degree	Poet Laureate of Despair
Major	Poetry
Advisor	Kelley Lynch

Upload Thesis

[step 1: select PDF to upload]

File	Type	Description	Size	Uploaded	Remove
	PDF	Thesis			

No PDF uploaded. Select PDF.

☑ My thesis includes supplemental files.

[step 1: select supplemental files to upload]

File	Type	Description	Size	Uploaded	Remove

No files uploaded. Select supplemental files.

Describe

[copy & paste from your thesis or dissertation]

Thesis Title

Abstract

Rights and Licenses

[check the statements that apply]

Copyright © Robert Galavan. All rights reserved. No part of this publication may be produced without the written permission of the copyright holder.

Clearance □ This submission contains copyrighted material.

Licenses □ I agree to the Stanford University license. »View Stanford University license

□ I want a Creative Commons license. »View Creative Commons license

Publication Delay □ I am requesting a publication delay of (select)

Progress

☑ Identification provided

□ Thesis PDF uploaded

□ Supplemental files uploaded

□ Abstract provided

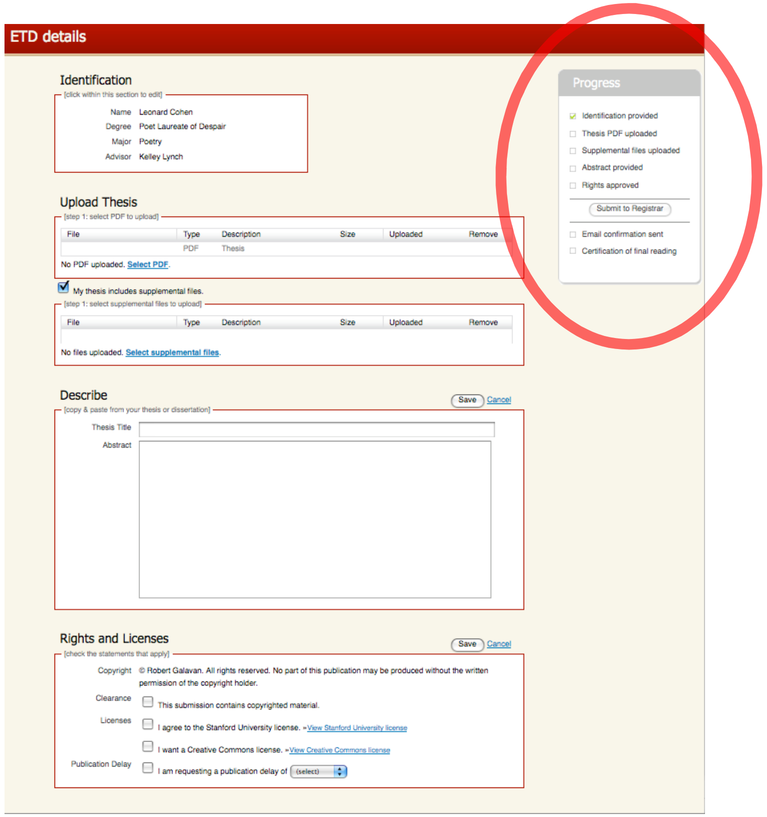
□ Rights approved

□ Email confirmation sent

□ Certification of final reading

Workflow

ETD Workflow



ETD details

Identification

[click within this section to edit]

Name	Leonard Cohen
Degree	Poet Laureate of Despair
Major	Poetry
Advisor	Kelley Lynch

Upload Thesis

[step 1: select PDF to upload]

File	Type	Description	Size	Uploaded	Remove
	PDF	Thesis			

No PDF uploaded. Select PDF.

☑ My thesis includes supplemental files.

[step 1: select supplemental files to upload]

File	Type	Description	Size	Uploaded	Remove

No files uploaded. Select supplemental files.

Describe

Save Cancel

[copy & paste from your thesis or dissertation]

Thesis Title

Abstract

Rights and Licenses

Save Cancel

[check the statements that apply]

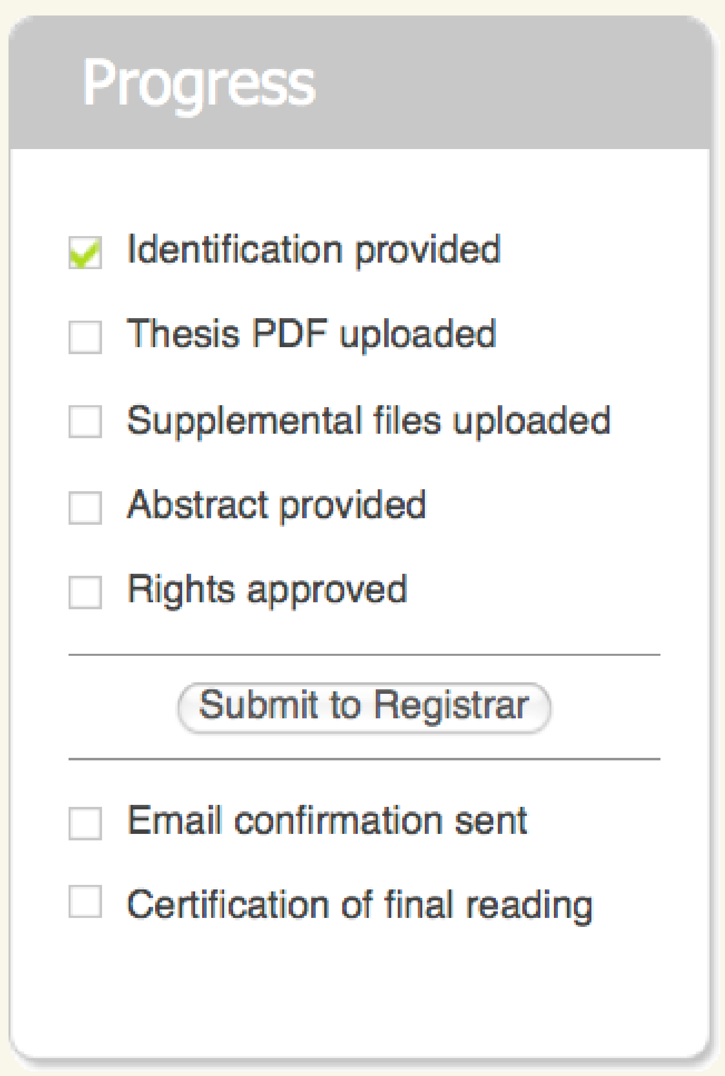
Copyright © Robert Galavan. All rights reserved. No part of this publication may be produced without the written permission of the copyright holder.

Clearance □ This submission contains copyrighted material.

Licenses □ I agree to the Stanford University license. »View Stanford University license

□ I want a Creative Commons license. »View Creative Commons license

Publication Delay □ I am requesting a publication delay of (select)



Progress

☑ Identification provided

□ Thesis PDF uploaded

□ Supplemental files uploaded

□ Abstract provided

□ Rights approved

Submit to Registrar

□ Email confirmation sent

□ Certification of final reading

Workflow Datastream for ETDs

```
<workflow id=“hydraEtd" status="active” …>
    <process name="register-object" status="completed” attempts="1" />
    <process name=”metadata" status="completed” attempts="1" />
    <process name=”upload" status="completed” attempts="1" />
    <process name=”attachments" status="completed” attempts="1" />
    <process name=”rights" status="waiting” attempts="0" />
    <process name=”submit" status="waiting” attempts="0" />
    <process name=”final-reading" status="waiting” attempts="0" />
    <process name=”registrar-approval" status="waiting” attempts="0" />
    <process name=”initiate-accession" status="waiting” attempts="0" />
</workflow>
```

Conclusion

- Hydra “out of the box” solutions must balance internal built-in solutions with dependence on institutional infrastructure
- 3 approaches will help distinguish between what Hydra apps need to do vs how they do it
- We are focusing on identifying key events within apps and coordinating service call
- Long range goals for easy assembly of dynamic workflows will take time

Contacts and links

THE UNIVERSITY OF HULL r.green@hull.ac.uk

UNIVERSITY of VIRGINIA nathan.piazza@gmail.com

SU LAIR Stanford University lmcrae@stanford.edu

FedoraCommons™ tstaples@fedora-commons.org

The Hydra Project
The University of Hull
The University of Virginia
Stanford University
Fedora Commons

r.green@hull.ac.uk

https://fedora-commons.org/confluence/display/hydra/