JISC

Project Information			
Project Acronym	CLIF		
Project Title	Content Lifecycle Integration Framework		
Start Date	1st April 2009	**End Date**	31st March 2011
Lead Institution	University of Hull		
Project Director	Chris Awre (Head of Information Management)		
Project Manager & contact details	Richard Green r.green@hull.ac.uk		
Partner Institutions	Centre for e-Research (CeRch), King's College, London		
Project Web URL	ww.hull.ac.uk/clif		
Programme Name (and number)	IE Programme: Repositories enhancement		
Programme Manager	Balviar Notay		

Document Name			
Document Title	Final Report		
Author(s) & project role	Richard Green (Project Manager), Chris Awre (Director), Simon Waddington (Site Manager, King's College London)		
Date	31 March 2011	**Filename**	CLIF-finalReportv10.pdf
URL	*if document is posted on project web site*		
Access	☐ Project and JISC internal	☑ General dissemination	

Document History		
Version	**Date**	**Comments**
0.9	19/04/2011	Final report less technical appendix
1.0	31/05/2011	Final report, budget, technical appendix

UNIVERSITY OF Hull

and

KING'S College LONDON

CLIF Project

CLIF Final Report

April 2011

The CLIF Project

Project Director:	Chris Awre	(c.awre@hull.ac.uk)
Project Manager:	Richard Green	(r.green@hull.ac.uk)
Project Site Manager for King's College:	Mark Hedges	(mark.hedges@kcl.ac.uk)
Project Manager for King's College:	Simon Waddington	
Lead software developer (Hull):	Andrew Thompson	
Lead software developer (King's):	Suresh Thampi	

The CLIF Project was undertaken by Library and Learning Innovation, with support from the Information and Communications Technology Department (ICTD), at the University of Hull and the Centre for e-Research (CeRch) at King's College London. It was funded by the JISC Information Environment Programme 'Repositories Enhancement' strand.

Table of Contents

Acknowledgements **5**

Executive Summary **6**

1. Background **7**

2. Aims and Objectives **8**

3. Methodology **9**

4. Implementation **10**

5. Outputs and Results **12**

Literature review 12

SharePoint-Fedora integration 13

Sakai-Fedora integration 14

6. Outcomes **15**

7. Conclusions **18**

8. Implications **19**

Appendix 1: Budget **20**

Appendix 2: technical appendix **23**

Acknowledgements

The project team gratefully acknowledges the support of the JISC Information Environment Programme in funding this project. We are also grateful to the partners in the Hydra Project (at the University of Hull, Stanford University and the University of Virginia) for their contributions to the project especially in the realm of Fedora repository content modelling.

Our thanks, too, to the staff of our two institutions who took part in user needs interviews and in testing the project outputs.

Executive Summary

At the heart of meeting institutional needs for managing digital content is the need to understand the different activities that the content goes through, from planning and creation through to disposal or preservation. Digital content is created using a variety of authoring tools. Once created the content is often stored somewhere different, made accessible in possibly more than one way, altered as required, and then moved for deletion or preservation at an appropriate point. Different systems can be involved at different points: one of these may be a repository. To embed repositories in the content lifecycle, and prevent them becoming yet another content silo within the institution, they thus need to be integrated with other systems that support other parts of this lifecycle. In this way the content can be moved between systems as required, minimising the constraints of any one system.

The CLIF Project has worked with creators of digital content to understand how they would like to deal with the interaction of the authoring, collaboration and delivery of materials using three systems that are in relatively common use: the Fedora Commons repository software, Microsoft SharePoint for authoring and collaboration, and the virtual learning environment, Sakai. Armed with this information, the project team then went on to design and produce software that would allow the transfer of digital content between the systems: Fedora and SharePoint, on the one hand, Fedora and Sakai on the other. The CLIF software has been designed to try and allow the maximum flexibility in how and when users can transfer material from one system to another, integrating the tools in such a way that they seem to be natural extensions of the basic systems.

It is likely that the software produced for CLIF, which offers a general approach, will be slightly refined post-project at the partner institutions, the University of Hull and King's College London. This specialisation will allow close integration with the respective institutional information architectures such that the CLIF materials can be used to enhance the flexibility of data flow between the various systems within them.

The CLIF software is available for interested parties to use. It has been written with adherence to appropriate standards and, whilst it will undoubtedly require tailoring to others' needs, it should form a good starting point for institutions with similar requirements.

This Final Report describes the work of the CLIF Project and, through its detailed technical appendix, explains in detail the development and eventual functionality of the software produced. In addition, it addresses some of the issues involved in dealing with a flexible content lifecycle.

1. Background

"No man is an island, entire of itself"
John Donne (1572-1631)
Devotions Upon Emergent Occasions, Meditation XVII

At the heart of meeting institutional needs for managing digital content is the need to understand the different activities that the content goes through, from planning and creation through to disposal or preservation. Digital content is created using a variety of authoring tools. Once created the content is often stored somewhere different, made accessible in possibly more than one way, altered as required, and then moved for deletion or preservation at an appropriate point. Different systems can be involved at different points: one of these may be a repository. To embed repositories in the content lifecycle, and prevent them becoming yet another content silo within the institution, they thus need to be integrated with other systems that support other parts of this lifecycle. In this way the content can be moved between systems as required, minimising the constraints of any one system.

The concept of a content lifecycle is not a new one. Records managers have long recognised its importance to their work, and the JISC Supporting Institutional Records Management programme in 2003[1] looked to build on this and the previous Study of the Records Lifecycle project.[2] The MoReq2 specification[3] also refers to the document lifecycle in the context of electronic records management systems. In the commercial world of enterprise content management there is much consideration of how the content lifecycle can be improved to maximise the benefit the content offers a business, and there is no shortage of commercial offerings to enable this.[4] One of the issues that arises in this field, though, is the lack of standards for managing the content across systems.[5] Standards do exist (see section 3.2), though, and can be applied to this space. The Digital Curation Centre Curation Lifecycle Model[6] is being used to stimulate work on the use of standards to support this, and builds on an earlier piece of research at UKOLN.[7] This stated:

"The life cycle approach is necessary because:

[1] JISC Supporting Institutional Records Management programme, http://www.jisc.ac.uk/whatwedo/programmes/supportingirm

[2] Study of the Records Lifecycle project, http://www.webarchive.org.uk/pan/13734/20060324/www.jisc.ac.uk/index79bf.html?name=recordsman_papers_cycle

[3] MoReq2 specification, http://www.moreq2.eu/index.htm

[4] For example, http://www.ecmconnection.com/article.mvc/Learn-From-Content-Lifecycle-Transformation-0002

[5] See http://it.toolbox.com/blogs/pcm/standards-and-content-lifecycle-18174

[6] DCC Curation Lifecycle Model, http://www.dcc.ac.uk/docs/publications/DCCLifecycle.pdf

[7] Pennock, M. Digital curation: a life-cycle approach to managing and preserving usable digital information, Library & Archives Journal, 2007, Issue 1, http://www.ukoln.ac.uk/ukoln/staff/m.pennock/publications/docs/lib-arch_curation.pdf

- Digital materials are fragile and susceptible to change from technological advances throughout their life cycle, i.e. from creation onwards;
- Activities (or lack of) at each stage in the life cycle directly influence our ability to manage and preserve digital materials in subsequent stages;
- Reliable re-use of digital materials is only possible if materials are curated in such a way that their authenticity and integrity are retained."

Similar reasoning saw the University of Illinois Library propose a digital content management approach.[8] The LIFE project also considered the content lifecycle, though explicitly left out hardware and systems from their cost models.[9] In contrast, the Information Technology and Information Storage industries (SNIA association) have addressed this issue, referring to the "… most appropriate and cost effective IT infrastructure …" required.[10]

The CLIF project was conceived to build on this previous work and address how to facilitate content lifecycle management within an institution.

2. Aims and Objectives

At the time of its inception, the partners in the CLIF Project shared a number of technology interests: the Fedora Commons repository software, Microsoft Office SharePoint Server 2007 (hereafter just 'SharePoint') and the community source virtual learning environment, Sakai. Expressed in very simple terms, CLIF hoped to facilitate the two-way movement of material between SharePoint and Fedora, on the one hand, and Sakai and Fedora, on the other.

By linking the repository into other content creation and management environments (ie SharePoint and Sakai) it is taken upstream in the user's workflow. Where the repository is best positioned within the content lifecycle was something that the project investigated with its users and the project used its early user needs interviews to get a 'feel' for this.

We suspected that in some situations it may be relevant at the end of the creation stage to move the content into a repository for access and/or preservation; or in others it may be appropriate to move content into the repository as a staging area for subsequent processing. The aim of integrating the repository at the appropriate part of the content lifecycle was to ensure that, when user activity crosses system boundaries, users would not feel constrained in what they wished or needed to do; rather, the systems in question between them would support these wishes and needs. For example,

[8] UIUC Digital Content Lifecycle Management project, http://www.library.uiuc.edu/nsm/digcon/

[9] LIFE project, http://www.life.ac.uk/

[10] See http://en.wikipedia.org/wiki/Information_lifecycle_management

moving content used for teaching in a VLE into a repository, maybe as part of building a portfolio, supports content re-use and the potential for long-term access.

CLIF started from a point of agnosticism about the direction content would flow between the repository and other systems (the lifecycle may require movement in both directions). Nevertheless, by facilitating the links between systems it was intended to support preservation by allowing the content to be moved to a system that has preservation capability.

Feeding into the development of preservation policies for the repositories as part of the project lay behind the technical work undertaken. Whilst looking at specific policies for preservation, the potential of incorporating the principles involved into wider institutional policies supporting research, teaching and administration was also to be explored, thus linking the management of the content to the purpose for which it is being managed.

At the outset of the project we had hoped to produce:

- A better understanding of the content lifecycle for different types of content as they are used for the purposes of research, teaching and administration. Understanding this assists in planning the implementation of systems on an institution-wide basis and facilitates the integration of repositories into institutional environments.

- Documented use cases to support the understanding indicated above, which would provide insights into how content passes through an institution so that it can be managed without 'falling between the cracks'.

- Technical documentation on the integrations carried out. By highlighting how systems can work together to support the content lifecycle we hoped that perceived duplication between systems and the role each could play would be clarified.

- A technical architecture to support the content lifecycle, which demonstrated how systems could be linked together to best support the lifecycle of different types of content. Key to this would be a recognition that not all content is managed in the same way, but that systems may need to be linked in different ways to meet different needs.

As the reader will see from the rest of this report, the project has gone a long way to fulfilling these aspirations. Whilst the Hull team divided their time between Sakai and SharePoint, King's College employed a specialist SharePoint developer who was most effectively employed doing SharePoint development. We felt that this was justified since the SharePoint-Fedora integration was carried out as a "green fields" activity whereas Sakai-Fedora integration reused a certain amount of work from an existing project.

3. Methodology

Both partners in the CLIF project have a history of undertaking JISC projects and so were able to build on approaches to its work already tried and tested.

The project did not take place in isolation from the rest of the world where other work in this field has taken place and so it was important to us to conduct a literature review to inform our starting position. More locally, it was important to us that we started by gathering ideas from the potential users of the project's outputs in order to clarify our thinking. Whilst one may feel that one understands the needs of colleagues, based on everyday interaction with them, it is often instructive to ask them explicitly about their needs and their views on the approach proposed by the project. With these interviews behind us, the project team would feel more confident that they were developing something that would be of real use.

Fedora, SharePoint and Sakai all provide a rich and complex set of functionalities, thus the design and development work needed to be preceded by a review of the available functionality, with particular regard to the use cases. The ability of each system to support different stages in the content lifecycle, where appropriate integration points are located, and how they will be enabled also needed to be examined.

Integrations between systems can be carried out using point-to-point techniques according to specific need. Whilst a loosely coupled approach to point-to-point can enable wider adoption of a solution, such solutions can also be limited by the systems themselves as they change over time. Enterprise Service Buses are an approach to abstract out the ways that systems can communicate with each other, protecting integrations against software changes. The project team was aware that it needed to assess the relative merits of the two approaches in the context of the two partner institutions and also in the context of possible wider adoption of its technical outputs. This assessment formed part of the preliminary work.

The Fedora repository software offers its users great flexibility in the ways they might choose to use it. The CLIF partners felt that it would be appropriate to work with it in some generally accepted way in order to make their work more generally useful. The Hydra Project[11] has produced guidelines for the construction of digital objects using Fedora which have gained wide attention and general acceptance, CLIF has therefore used the Hydra guidelines as the basis for building its repository objects.

Thus informed, the CLIF team developed an overall architecture for the proposed integration and set about the technical work. The potential users that we had interviewed in the early stages of the project were invited to test and comment on the work that had been undertaken in order to inform future development.

4. Implementation

The first element of the project was a literature review: a piece of cross-disciplinary desk research in liaison with the contributing academics for their subject and role-related input. This resulted in a document that does not aim to produce or summarise the many different examples of lifecycle in

[11] See: https://wiki.duraspace.org/display/hydra/The+Hydra+Project

existence, but instead addresses the issues that have emerged through developing or examining such lifecycles, particularly focusing on the, limited, literature examining lifecycles across different systems. It is available for download at:

https://edocs.hull.ac.uk/splash.jsp?parentId=hull:1647%26pid=hull:2430

The next part of our work was to gather user requirements from colleagues at our two institutions. There was a recognised difference of emphasis: colleagues at King's College were particularly interested in the requirements for research data whilst at Hull the emphasis was more on text-based materials. The interviewees were from a range of backgrounds, both learning and teaching and administrative, and were chosen to cover a wide range of possibilities. The interviews sought to discover how people dealt with digital content and what kinds of software were used to manage it. From all this information two generic use cases, one for experimental data and documentation, the other for essentially textual material, were derived for CLIF to address. This use case summary can be found at:

https://edocs.hull.ac.uk/splash.jsp?parentId=hull:1647%26pid=hull:2431

The final part of CLIF's preliminary work covered three areas: a technical review of the three major software packages involved (Fedora, SharePoint and Sakai), a review of the possibilities for using an Enterprise Service Bus (ESB) approach to CLIF's technology work, these two to inform a proposed architecture for CLIF's technical outputs. Whilst these were originally planned as three reports it became clear as we developed them that a single, combined and integrated approach made more sense. It should be made clear at this point that CLIF has worked with SharePoint 2007; during the course of the project SharePoint 2010 was made available but some further work will be needed to port the CLIF materials to this new platform. This work will be investigated in Hull post-project in order that the CLIF software can be used in conjunction with a planned institutional roll-out of SharePoint which would, necessarily, be the newer version. King's will also be doing their own assessment of how SharePoint 2010 can take advantage of the CLIF outputs for their own institutional rollout later this year. This combined report can be found at:

https://edocs.hull.ac.uk/splash.jsp?parentId=hull:1647%26pid=hull:2697

A significant outcome of this technical review was that an ESB approach to CLIF's development was, whilst having a number of key advantages in supporting message abstraction between different systems, considered too complex within the existing context of infrastructure at the two partner institutions, problematic within the timeframe of the project, and potentially limiting for the use of its outputs elsewhere. A second important outcome was the decision to base CLIF's Sakai-Fedora integration on code previously developed by the JISC-funded CTREP project.[12] Whilst, at the time of its development, CTREP's code did not fully achieve its aims this was felt to be due to limitations in Sakai which had since been resolved; using CTREP as a starting point potentially shortened CLIF's Sakai-Fedora development cycle. The decision was reviewed (but not altered) at several points as the project progressed. Although the CTREP code gave us a starting point, CLIF was still left with a

[12] CTREP, see: http://www.jisc.ac.uk/whatwedo/programmes/reppres/sue/tetracam.aspx

great deal of work to achieve a useful integration. We periodically reviewed this decision and queried whether we might have been better starting from a 'blank page', each time concluding, however, that the CTREP approach was the more valid option.

The development of CLIF's integration software has been a complex process involving development at both Hull and King's. The whole process has been managed through a number of face-to-face meetings but mainly through frequent Skype conferences. At a coordination level these 'Skypes' have generally taken place fortnightly (sometimes even weekly) with the developers often having additional, intermediate calls. A single SVN software repository hosted at Hull was used by both sites.

It would be inappropriate, which is to say far too complex, to explain the technical development of the CLIF materials here in a fairly general document. Rather, the three technical reports covering the actual architecture as finally implemented, the development and implementation process, and the testing processes used are provided here as a combined technical appendix (Appendix A).

5. Outputs and Results

Literature review

The review of the literature carried out on digital content lifecycles identified a wide range of publications on this topic. The emphasis for the CLIF project was on previous work that had specifically looked at the system management aspects of managing such lifecycles, in order to inform the integration work that CLIF would be carrying out. In this the review was not so successful, with the vast majority of literature focusing on the process issues and the various steps that can be considered to be part of such lifecycles (on which there is a reasonable consensus with a number of variations on a theme). The literature was addressed according to the following themes, which the publications found largely fell into:

- Content lifecycles
- Lifecycles and digital preservation
- Lifecycles and content management
- Technology and the lifecycle
- Living the lifecycle
- Standards
- The knowledge lifecycle
- Developing the lifecycle

Key to the CLIF project were conclusions relating to lifecycle information and understanding the different stages of the lifecycle. Lifecycle information describes the information around content that describes its current state, which can help understand where it is in the lifecycle. We discussed at length the value of capturing this information and storing it, but had to concede in the end that unless there was a pressing need this was mostly overkill in trying to enable seamless movement of content between systems, and only what could be captured straightforwardly was held. Understanding the different stages of the lifecycle helped to clarify the use cases we identified and how they can be enabled. Whilst we have further work to do to fully describe our different lifecycles, noting the roles different systems can play most effectively can help to understand where the different systems are best used.

SharePoint-Fedora integration

CLIF extends the functionality of SharePoint MySite. Although MySite was used as the basis for development, the CLIF work could easily be adapted for use for document archiving in other site templates. When a new MySite user is added by an administrator, the CLIF system automatically creates a Fedora account for the user as well as creating a Fedora object as the basis for the user's private repository area under the MySite root object. Within MySite, users can have access to both the existing functionality as well as the additional features provided by CLIF. As part of the document upload process, a certain amount of general metadata is gathered, which can then be appended to the Fedora object that is deposited to the repository.

Users can 'move' a document to the associated repository: this creates a Fedora object in the private archive area of the repository and deletes the SharePoint instance of it. The associated metadata is retained and is re-associated with the object should the file be brought back from archive. This is effectively a short- or medium-term preservation strategy.

Two versions of depositing a copy of a document to the repository are provided (which option(s) are provided to the user is configurable by an administrator). 'Publishing' a document starts a SharePoint workflow which needs to be completed in order for the content to reach the general (public facing) repository (such workflow may require, for instance, approval steps). The list of locations to which a user is able to publish within the general repository can be configured at the top level by an administrator and is presented to the user in a pull-down list on a web form. The second option provided, 'Copy to Repository', takes the object created and places it in a specific place within the repository for further processing by others (this is effective as an accession queue containing materials to be dealt with by repository managers). This repository location typically has restricted access. This second process provides the user with the opportunity to provide significant metadata about the content they are publishing, appropriate to objects being exposed in an institutional repository; this is MODS metadata by default but can be Dublin Core or both. Where possible, default entries in the metadata fields are derived from the user's SharePoint environment.

Deposit of multiple documents to the repository, either copy or move, is provided by an additional feature that enables the user to select multiple documents from a document library. This could be used for instance when the user has completed a project and wishes to archive a large number of files.

Documents that have been 'moved' from SharePoint to the repository can be retrieved by navigating to the Archive list and selecting the URL of the document. This retrieves the document from the repository to the local file system.

A repository browse functionality is provided that enables the user to browse their private folders in the repository as well as the public repository folders.

A text box on the MySite pages enables the user to enter search queries. Search can be performed across metadata and text of all documents in MySite as well as the metadata of documents that have been moved to the repository (since the metadata is retained in SharePoint). A URL is provided in the search results that retrieves the document to the local file system. Due to time limitations, it was not possible to include free text searching across Fedora, although this could be added using Solr indexing.

Sakai-Fedora integration

The CLIF work integrates Fedora into the Sakai resources tool. This is an area of Sakai in which users can store digital materials for their own use and, potentially, share them with other users of Sakai. The system allows the creation of a tree structure to aid organisation of the materials held and provides a range of functions to manage them: upload, copy, edit, move, delete, and so on.

CLIF makes the linked Fedora repository appear as a folder within this resources tree with the full range of management functionality allowing movement of content between Sakai folders and repository folders. By transferring materials into repository folders they are potentially shared outside the Sakai environment and may be in a better location for medium- or long-term preservation.

In the CLIF version of this code the repository is unsecured and so users can move materials to and from any point in the repository structure below Sakai's root node there and are free to create and delete content at will. The post-project development at Hull will produce a version where users can deposit materials only into a specified part of the repository structure, effectively the accession queue, and will be able to browse (read-only) solely those parts of the repository permitted by their level of authorisation.

In terms of technology, the CLIF Project set out to provide software which would, on the one hand, provide integration of SharePoint and Fedora and, on the other hand, integrate Sakai with Fedora. In both instances the project has been successful although it is important to be clear about the scope of what has been achieved.

The 'F' in Fedora stands for 'flexible' and there are many ways in which a Fedora repository can be implemented. For the purposes of the Sakai-Fedora integration in CLIF we have developed software that can be used with an open Fedora instance, which is to say that no authentication or authorisation is required to access its content. To have done otherwise would have involved us in determining, and dealing with, the many ways in which Fedora can be secured. This base code will thus be suited only for institutions where the Fedora repository is unsecured. Whilst holding much open access content, Hull's repository does require security for access to a number of our collections. Post-project we shall undertake the small amount of additional work necessary to have the CLIF code work securely with Hull's systems and we shall share this modified code with anyone interested. This code will, it should be noted, be code modified specifically to work with Hull's secured repository and the environment in which it sits, and will not be a general solution. That said, we anticipate that it will be sufficient to point others at the modifications they may wish to make in their own situations.

A second caveat was mentioned in Section 4: the SharePoint work has been carried out against SharePoint 2007, the product available at the start of the CLIF Project. Adaptation of our code to SharePoint 2010, which has since been released, will take some further work. The CLIF solution has been structured so that feature definitions and C# code are separate. This potentially simplifies the migration process since only the code containing the feature definitions needs modification when adapting for 2010.. It is likely that the University of Hull will roll out SharePoint 2010 as an institutional service; if this initiative goes ahead then Hull will need to upgrade the CLIF code post-project. The team in Hull, of course, will be happy to share that experience and code with interested parties. At the time of writing, the nature of King's plans to incorporate the CLIF work with their institutional rollout of SharePoint 2010 is still under discussion.

As is so often the case in such matters, the simple descriptions of CLIF's outputs below belie the considerable complexity of the code needed to achieve them. This is notwithstanding the use of standards wherever feasible and highlights an implicit complexity in getting systems that manage content to work in ways they were not originally scoped to do.

Evaluation
Use cases specific to the two partner sites were identified through interviews with stakeholders. Following the integration work carried out these use cases were re-tested with as many of the original stakeholders as could be contacted. As described in the use case documentation from the project these were broadly related to teaching, research and administration.

For both SharePoint and Sakai a number of common observations emerged from the evaluation interviews carried out:

- There needs to be a clear understanding and view about where the boundaries are between the different systems being used, to avoid confusion
- There needs to be clarity over why different systems are being used, to overcome concerns about having to work with multiple systems
- There is a need for better preservation and a recognition that integrating the repository could support this, but also a need to be clear about what needs preserving
- There is benefit in being able to access other content stores from within your current working environment in order to see what is available more broadly

6. Outcomes

The project has explored what turned out to be a little studied area of interest for digital content lifecycle management, that of the practical application of a lifecycle across multiple systems. Our dissemination has, and will, raise this issue in the context of the work undertaken to explore the digital content lifecycle and how the CLIF work has sought to enable how the lifecycle can be managed across different systems.

The CLIF Project has largely been successful in its technological aims. The project has produced software that allows transfer of content between, on the one hand, SharePoint and Fedora, and on the other, Sakai and Fedora. This potentially allows users to transfer their content relatively freely between these systems in order to support their needs. There are no rigid workflows (workflow in SharePoint is configurable and optional) to define when transfer between systems must take place; rather the CLIF work allows the user flexibility in how they would like to interact with their various authoring, collaboration and delivery systems. Implementation of the project's outputs will be taken forward, and the software made available for others to further explore how the digital lifecycle can be enabled between the systems we have used.

The Fedora-Sakai integration, building as it does on the CTREP project's work, complements the DSpace-Sakai integration successfully carried out as part of that original project, and further enhances an understanding of how the established and stable Content Host Handler (CHH) can be used to surface content from different sources. The work highlights that the CHH would benefit from development as it is not as clean or rich an interface as it could be, and has fallen behind subsequent standards work in content management interfaces (notably, CMIS), but is capable of being utilised in a variety of ways: it is anticipated that the current development of Sakai 3 will address this.

The Fedora –SharePoint integration has highlighted that whilst many perceive the use of SharePoint to dictate a Microsoft-only environment that it is feasible to link SharePoint to a content management system with a very different technical architecture, making the most of the flexibility of both platforms, and the approaches used could be adapted for other content management systems as well. This relates very closely to the work of the RIC framework, the Microsoft initiative to link external content systems to SharePoint to support research, and the CLIF work has been in regular touch with the RIC community to highlight how the project outputs could be effectively used within this framework.

Hull

It is almost certain that this functionality will be made available to staff and administrators at Hull where Sakai and the institutional Fedora repository are key parts of the information infrastructure. The ability to browse a collection within the repository from Sakai will be utilised as part of an exploration of open educational resources that is planned for 2011-12, offering the scope for lecturers to more easily identify locally held OERs for use in their own courses. The discussions around preservation will be tied into parallel activity by the University Archives on the management of digital archives, to identify how integration with Sakai can be used as a mechanism for such archiving.

SharePoint integration with the repository will be further explored as part of the institutional SharePoint roadmap being planned for 2011-12. This will adopt SharePoint 2010, and migration of the CLIF outputs to this updated version of SharePoint will be undertaken as required.

King's College London

At King's College London, a strategy white paper for SharePoint 2010 was prepared during early 2011 in which CLIF staff participated. Use cases and knowledge from CLIF were contributed to the document, which was subsequently approved by the ISS leadership team. Thus CLIF has already contributed considerably to the thinking and future deployment of SharePoint at King's. Work is now underway to prepare for deployment starting with pilots in selected departments commencing towards the end of 2011.

The CLIF prototype has already been demonstrated to staff in the King's Archives and further demonstrations are planned to senior staff within ISS. The King's Archives are already considering how important outputs stored within SharePoint can be preserved and we plan to continue to carry out further work with the Archives in reusing the CLIF outputs through the internal ODM project.

King's are also in the process of making final preparations for the public launch of the AKORD publications repository. The current version of AKORD is based on the Fedora Commons repository and is already a key resource at King's, containing over 80,000 records.

Embedding in institutional policy

In CLIF's Project Proposal, repeated in the Project Plan and in the introduction to this document, we said that "To embed repositories in the content lifecycle, and prevent them becoming yet another content silo within the institution, they … need to be integrated with other systems that support other parts of this lifecycle. In this way the content can be moved between systems as required, minimising the constraints of any one system." CLIF's work represents a significant step in this integration process.

Part of the embedding involves the related embedding in local policy, so that there is common understanding of what is required and how it will be enabled. This is particularly relevant in the sphere of digital preservation and how this is enabled by the content management systems used.

- At Hull we are currently going through a process of refreshing our Corporate Strategy, combining a new plan for the next five years with a vision for where the University should be in 2030. The need for good information management and preservation is being fed into this process. Following agreement of the Corporate Strategy, in July this year, related strategies guiding specific areas of activity, primarily learning & teaching and research, will develop their own strategies where the same good practice and needs will be scoped further. Aside from the corporate level, the Mellon-funded AIMS project in which Hull is a partner is taking forward the development of a digital preservation strategy for the University. Whilst this will be primarily developed for archival materials it is anticipated that the strategy can also be applied more broadly to other materials.
- At King's the JISC-funded Pekin[13] project has contributed much to the development of appropriate policies for use within King's College. Two draft policies were developed, one aimed at preservation and one more specifically at digital preservation, and both are being further developed under a current institutional project, ODM, being led by the College Archives.

[13] See: http://www.jisc.ac.uk/whatwedo/programmes/inf11/digpres/pekin.aspx.

7. Conclusions

The following conclusions can be derived from the work of the CLIF project.

- The management of digital content lifecycles has been extensively explored in the literature, from many different perspectives and in many different subject and content domains. The majority of these explorations focus on the processes involved in managing the different steps of the lifecycle, and whilst there is variation there is also a great deal of consensus in the descriptions of digital content lifecycles. This project has not sought to replicate this work or add to the variations in existence, rather focus on the implementation of the digital content lifecycle across multiple systems. This practical aspect of how a digital content lifecycle can be put into practice is far less explored in the literature. This may be because technologies change and consistency in process is more important that focusing on specific systems; it may be that different domains put their findings into practice using technology designed for that domain, and do not have an identified need to move out of that domain. The literature suggests both. CLIF challenges in particular this latter position by recognising that different systems used to manage digital content within a University do not have to work in isolation, but can be used together.
- The technical integration work carried out has successfully demonstrated that diverse content management systems can be brought together to allow the seamless movement of content between them. Having identified a set of use cases from interviews with local users, we were nevertheless keen to ensure that implementing these use cases did not preclude other uses for the movement of content between the systems, and implemented them in as generic a way as possible. This has resulted in a flexible set of outputs that can be further developed and applied. Our evaluations revealed additional functionality and use cases that could be implemented, and we anticipate further use cases emerging as we implement the project's outputs more widely and more users become familiar with what is feasible.
- The work required to carry out the integration has been extensive and detailed, and it can also be concluded that the lack of standards in the interfaces for content management presented by both Sakai and SharePoint does not make the task of getting such systems to work together any easier. It is concluded from this experience that all content management systems should be encouraged to make it as easy to get content out as it is to get content into them in order to facilitate seamless flow and enable the digital content lifecycle across systems.
- An assumption at the start of the project was that we would be agnostic about the direction in which content might flow between the systems once integrated. Evaluation feedback clearly suggests that the repository's archival capability is regarded as one of its strongest assets, and the area that the other systems could not offer comparable functionality on. Hence, the primary flow of content is into the repository. This suggests that the role of the repository within a University will be regarded very much in terms of what it can offer that the other systems cannot, rather than try and compete on all levels. Whilst there is clear benefit in playing to one's strengths there is a challenge to clarify better at an institutional level what functionality is offered by different content management systems, so as to more fully understand how different stages of the digital content lifecycle can be best enabled.

8. Implications

The following implications arise from the work of the CLIF project. Other implications are incorporated into the conclusions listed above.

- The software produced to enable the integration carried out can be taken by others and adapted for their own needs. Developing the software further would be beneficial to enrich the current solutions. However, it should be noted that the platforms CLIF worked with, SharePoint 2007 and Sakai 2, are both being superseded by subsequent developments themselves (SharePoint 2010 and Sakai 3), and future development effort may be best focused around these enhanced offerings. The experience of the CLIF project can, though, help to inform such future developments.
- Working with SharePoint requires a detailed knowledge set in order to make most effective use of the system and integrate it with others. As a platform the tools are there, but a good knowledge of those tools will be helpful. Sakai, too, requires a detailed, but different, knowledge set.
- Further work on the use of the repository as a provider of content, as opposed to a passive archive, would be helpful to explore more the capability that repositories have in supporting cross-system lifecycle management.
- Addressing the third side of the system triangle, the integration between SharePoint and Sakai, would also be helpful. There is no absolute reason that the repository needs to be part of the content flow, albeit that the project has helped to better establish the repository as part of the local institutional landscape.
- Receiving content from multiple sources has the implication that the repository needs to lay down standards over how that content should be structured, so it can be most effectively managed within the repository as a whole. The CLIF project has been fortunate to run in parallel with developments in the Hydra project, which have provided us with that common structure.

Appendix 1: Budget

Directly Incurred Staff	TOTAL BUDGET £	Year <09-10> Actual Expenditure	Year <10-11> Actual Expenditure	TOTAL EXPENDITURE £	TOTAL VARIANCE
Project Manager (0.5FTE, Hull, Subcontractor)	£ 36,750	£ 21,000	£ 15,750	£ 36,750	£ 0
Analyst/Developer (1.0FTE Hull)	£ 65,721	£ 14,341	£ 50,881	£ 65,222	£ (499)
Analyst/Developer (1.0FTE CeRch)	£ 77,220	£ 0	£ 82,147	£ 82,147	£ 4,927
Total Directly Incurred Staff (A)	**£ 179,691**	**£ 35,341**	**£ 148,778**	**£ 184,119**	**£ 4,428**
Non-Staff					
Travel and expenses	£ 20,000	£ 6,679	£ 7,490	£ 14,169	£ (5,831)
Hardware/software	£	£	£	£	£
Dissemination	£ 10,000	£ 25	£ 10,131	£ 10,156	£ 156
Evaluation	£ 3,000	£ 0	£ 2,777	£ 2,777	£ (223)
Other	£	£	£	£	£
Total Directly Incurred Non-Staff (B)	**£ 33,000**	**£ 6,704**	**£ 20,398**	**£ 27,102**	**£ (5,898)**
Directly Incurred Total (A+B=C)	**£ 212,691**	**£ 42,045**	**£ 169,176**	**£ 211,221**	**£ (1,470)**

Directly Allocated					
Staff	£ 44,828	£ 25,006	£ 19,822	£ 44,828	£ 0
Estates	£ 22,380	£ 13,395	£ 8,985	£ 22,380	£ 0
Other	£	£	£	£	£
Directly Allocated Total (D)	**£ 67,208**	**£ 38,041**	**£ 28,807**	**£ 67,208**	**£ 0**
Indirect Costs (E)	**£ 117,314**	**£ 69,774**	**£ 47,540**	**£ 117,314**	**£ 0**
Total Project Cost (C+D+E)	**£ 397,213**	**£ 150,220**	**£ 245,523**	**£ 395,743**	**£ (1,470)**
Funds Received from JISC	**£ 299,854**	**£ 169,443**	**£ 130,411**	**£ 299,854**	**£ 0**
Institutional Contributions	**£ 97,359**	**£ 57,250**	**£ 40,109**	**£ 97,359**	**£ 0**

Nature of Institutional Contributions

Directly Incurred Staff					
Project Director 0.1 fte	£ 10,549	£ 6,028	£ 4,521	£ 10,549	£ 0
Directly Incurred Non Staff					
Hardware/Software etc.	£	£	£	£	£
Directly Allocated					
Staff, Estates etc.	£ 4,690	£ 2,380	£ 2,310	£ 4,690	£ 0
Indirect Costs					
Indirect Costs	£ 82,120	£ 48,842	£ 33,278	£ 82,120	£ 0
Total Institutional Contributions	**£ 97,359**	**£ 57,250**	**£ 40,109**	**£ 97,359**	**£ 0**

The small budget underspend of £1,470 will be retained to fund further dissemination at the Sakai European Conference in Amsterdam during September 2011.

Appendix 2: technical appendix

We acknowledge with thanks the detailed contributions to this appendix made by our software developers Andrew Thompson (Hull) and Suresh Thampi (King's College, London

and

CLIF Project

Final Report: technical appendix

Chris Awre, Richard Green, Suresh Thampi, Andrew Thompson,
Simon Waddington.

May 2011

The CLIF Project

Project Director: Chris Awre (c.awre@hull.ac.uk)
Project Manager: Richard Green (r.green@hull.ac.uk)
Project Site Manager for King's College: Mark Hedges (mark.hedges@kcl.ac.uk)

The CLIF Project was undertaken by Library and Learning Innovation, with support from the Information and Communications Technology Department (ICTD), at the University of Hull and the Centre for e-Research (CeRch) at King's College London. It was funded by the JISC Information Environment Programme 'Repositories Enhancement' strand.

Table of Contents

1. Introduction 5

2. Summary of requirements 6
2.1 SharePoint-Fedora integration 6
2.1.1 Fedora repository 6
2.1.2 Deposit 6
2.1.3 Browse and retrieval 7
2.1.4 Search 7
2.1.5 System administration 7
2.1.6 Excel calculation 8
2.2 Sakai-Fedora Integration 8
2.2.1 Resources Tool Overview 8
2.2.2 CTREP Fedora Resource Tool Enhancement 8
2.2.3 Functional requirements 10

3. Development and implementation 12
3.1 Approach 12
3.2 Fedora configuration 13
3.2.1 Hydra-compliant Fedora objects 13
3.2.2 Repository structure 14
3.2.3 Fedora user account configuration 15
3.2.4 Fedora XACML policies 15
3.3 SharePoint-Fedora Integration 16
3.3.1 Technical overview 16
3.3.2 Integration architecture 17
3.4 Solution structure 18
3.4.1 Site map 21
3.4.2 CLIF root site functionality 22
3.4.3 Deposit to repository 23
3.4.4 Retrieval from repository 28
3.4.5 Repository Explorer 29
3.4.6 Search repository 30
3.4.7 Excel calculation 31
3.4.8 SharePoint features 31
3.4.9 Metadata entry 32
3.5 Sakai-Fedora Integration 33
3.5.1 Integration architecture 33
3.5.2 Added functionality 37
3.5.3 Performance Improvements 39
3.5.4 Future work needed 40
3.6 CLIF-Sakai functionality 42
3.6.1 Resources page 42
3.6.2 Copy/move to repository 42
3.6.3 Remote folder creation/removal 44

4. Testing 47
4.1 General approach 47
4.2 SharePoint-Fedora integration 47

4.2.1 Unit tests 47
4.2.2 Integration tests 47
4.2.3 System tests 47
4.3 Sakai-Fedora integration 47
4.3.1 Unit tests 47
4.3.2 Integration tests 48
4.3.3 System tests 48

5. Evaluation 49
5.1 General approach 49
5.2 SharePoint-Fedora integration evaluation 49
5.2.1 Technical evaluation 49
5.3 Sakai-Fedora integration evaluation 50
5.3.1 Technical evaluation 50

6. SharePoint configuration guide 52
6.1 Setting up the SharePoint site 52
6.1.1 Creating the SharePoint web application 52
6.2 Setting up MySite in SharePoint 53
6.3 Configuring SharePoint search settings 53
6.3.1 Create content source 53
6.3.2 Create search scope 53
6.3.3 Create search scope rules 53
6.3.4 Adding managed properties 54
6.3.5 Crawl SharePoint site content 54
6.4 Configure Excel web services is the SharePoint site 55
6.4.1 Overview 55
6.4.2 Configure Excel calculation services 55
6.4.3 Enable user defined functions 55
6.5 Configuring Fedora 56
6.5.1 Creating root folder 56
6.5.2 Create publishable locations folder 56
6.6 Configuring root SharePoint MySite site 56
6.7 Deploy SharePoint solution 56
6.7.1 Adding CLIF.Solutions.wsp to the server 56
6.7.2 Deploying CLIF.Solutions.wsp to a SharePoint site 56
6.8 Configuring root SharePoint site 57
6.8.1 Activating features 57
6.8.2 Setting up lookup list 58

7. Sakai configuration guide 59
7.1 Pre-Requisites 59
7.2 Installation 59

8. Acronyms and abbreviations 61

1. Introduction

The CLIF project recognises that

> "At the heart of meeting institutional needs for managing digital content is the need to understand the different activities that the content goes through, from planning and creation through to disposal or preservation. Digital content is created using a variety of authoring tools. Once created the content is often stored somewhere different, made accessible in possibly more than one way, altered as required, and then moved for deletion or preservation at an appropriate point. Different systems can be involved at different points: one of these may be a repository. To embed repositories in the content lifecycle, and prevent them becoming yet another content silo within the institution, they thus need to be integrated with other systems that support other parts of this lifecycle. In this way the content can be moved between systems as required, minimising the constraints of any one system."

CLIF Project Plan, April 2009

The project thus set out to investigate how flexible support of such a lifecycle might be enabled in an institution that uses a Fedora-based repository[1] and manages documents using Microsoft Office SharePoint Server[2] (henceforth just 'SharePoint') and/or the Sakai academic collaboration platform.[3]

The project's Final Report and this appendix bring together three separate reports that were outlined in the CLIF Project Plan:[4]

- D6 A description of the CLIF architecture as implemented
- D7 A document describing the development and implementation of CLIF
- D8 A document describing the testing of the CLIF outputs

With the benefit of hindsight it is clear that the three reports are intimately related and it now seems more appropriate to present them as a single appendix to the project's Final Report where appropriate cross-referencing can be achieved more easily. We have attempted to avoid unnecessary duplication of information between the Final Report proper and this appendix and thus readers will find here technical detail whilst more general matters are dealt with in the body of the Report.

[1] Fedora Commons website: http://fedora-commons.org
[2] Microsoft SharePoint site: http://sharepoint2007.microsoft.com
[3] Sakai Project See: http://sakaiproject.org/
[4] CLIF Project Plan at https://edocs.hull.ac.uk/muradora/objectView.action?parentId=hull:1647&type=1&pid=hull:1808

2. Summary of requirements

This section contains a brief overview of the main requirements that were implemented in the CLIF system prototype. This section has been included in order to assist the reader in understanding the subsequent technical sections on system implementation and testing. The reader is referred to the CLIF deliverables for more detailed discussion of the user requirements and system architecture.

2.1 SharePoint-Fedora integration

The requirements for the SharePoint-Fedora integration can be divided into five main categories: the requirements for the Fedora repository configuration, deposit of documents from SharePoint to Fedora, retrieval of documents in Fedora from SharePoint, search of the Fedora repository from SharePoint, system administration and Excel spreadsheet calculation. Each of these categories is detailed below.

2.1.1 Fedora repository

R1.1: The repository must comprise a public and a private area. The public area can comprise one or more separate repository folders. Public folders are assumed to allow uniform access to a large number of users (which could be internal or external). Access to the private area is restricted to the owner of the content, system administrators and possibly other users (e.g. the SharePoint group to which the user belongs).

R1.2: The repository is organised using a hierarchical folder structure.

R1.3: The Fedora repository will be assumed to only allow deposits from Hydra-compliant systems depositing simple Fedora objects. (The meaning of this compliance is discussed at section 3.2.1).

R1.4: Documents deposited in the repository are read-only. No facility is provided from SharePoint to remove documents from the repository.

2.1.2 Deposit

R2.1: The system shall enable deposit of a document from a SharePoint document library list to a folder in Fedora (either public or private) from the context menu of that document.

R2.2: The deposit to a public area in Fedora can be configured to run an approval workflow. The approval workflow is configured by an administrator at the top level of SharePoint. This should allow any SharePoint user to be assigned as the approver.

R2.3: The user can select to deposit documents to the public folder in Fedora from a pull-down list of available public folders.

R2.4: The system will allow users to enter basic metadata about the document. Fields in the metadata form should be pre-populated automatically where appropriate from the metadata available in SharePoint.

R2.5: The system can be optionally configured to enable users to enter more detailed (MODS) metadata that is required for preservation.

R2.6: The user can copy a document and associated metadata to a private area, leaving the original document *in situ*.

R2.7: The user can move a document to the repository leaving a hyperlink in the SharePoint document library containing the URL of the document in Fedora.

R2.8: The list of archived documents that have been moved from Fedora can be browsed from an Archive List within the user's MySite area.

R2.9: The user shall be able to browse their private area in Fedora, create folders and select the location to which files should be deposited.

R2.10: If a document is deposited to Fedora from SharePoint that had previously been retrieved, a new Fedora object is created.

2.1.3 Browse and retrieval

R3.1: The system should enable documents that have been moved to Fedora (c.f. R2.7) to be retrieved to the user's local file system by clicking on the hyperlink inserted in place of the document.

R3.2: The system should enable documents that have been copied to Fedora (c.f. R2.6) to be retrieved to the user's local file system by clicking on a link in the SharePoint list of documents in that have been archived.

R3.3: When documents are retrieved from Fedora to SharePoint, only the document file is returned and not the metadata. The original copy of the document is retained in the repository.

R3.4: The system enables a user to select a repository location to browse (either their private folder or a public folder), navigate through the folder structure and list the files contained in that repository folder.

R3.5: The system should allow users to retrieve documents that have been selected as in R3.4 from the repository to their local file system.

2.1.4 Search

R4.1: The system should allow keyword search of the structured metadata and full text of documents in the SharePoint MySite document library.

R4.2: The search of document libraries MySite in R4.1 includes query of public and private folders in Fedora.

2.1.5 System administration

R5.1: Upon creation of a MySite SharePoint site for a given user, the system shall automatically create a Fedora user account for the user, create a root folder to the private repository area for the user and configure the security policy settings for the private repository area.

R5.2: Upon creation of a MySite in requirement R5.1, the system should configure the user controls and lists required to support the deposit, browse and retrieval of documents from the repository.

R5.3: The CLIF extensions to MySite should be extended to any new document libraries created by the user within their MySite.

R5.4: The system administrator can add or remove locations of public folders in the repository where the items can be deposited according to requirements R2.3.Excel calculation

2.1.6 Excel calculation

R6.1: The user can upload a master Excel spreadsheet to a trusted location in their MySite.

R6.2: The user can run the calculations in the master spreadsheet against data stored in an input spreadsheet.

R6.3: The user can archive the results of spreadsheet calculations, either by copying or moving the spreadsheet to the repository and adding appropriate metadata.

2.2 Sakai-Fedora Integration

2.2.1 Resources Tool Overview

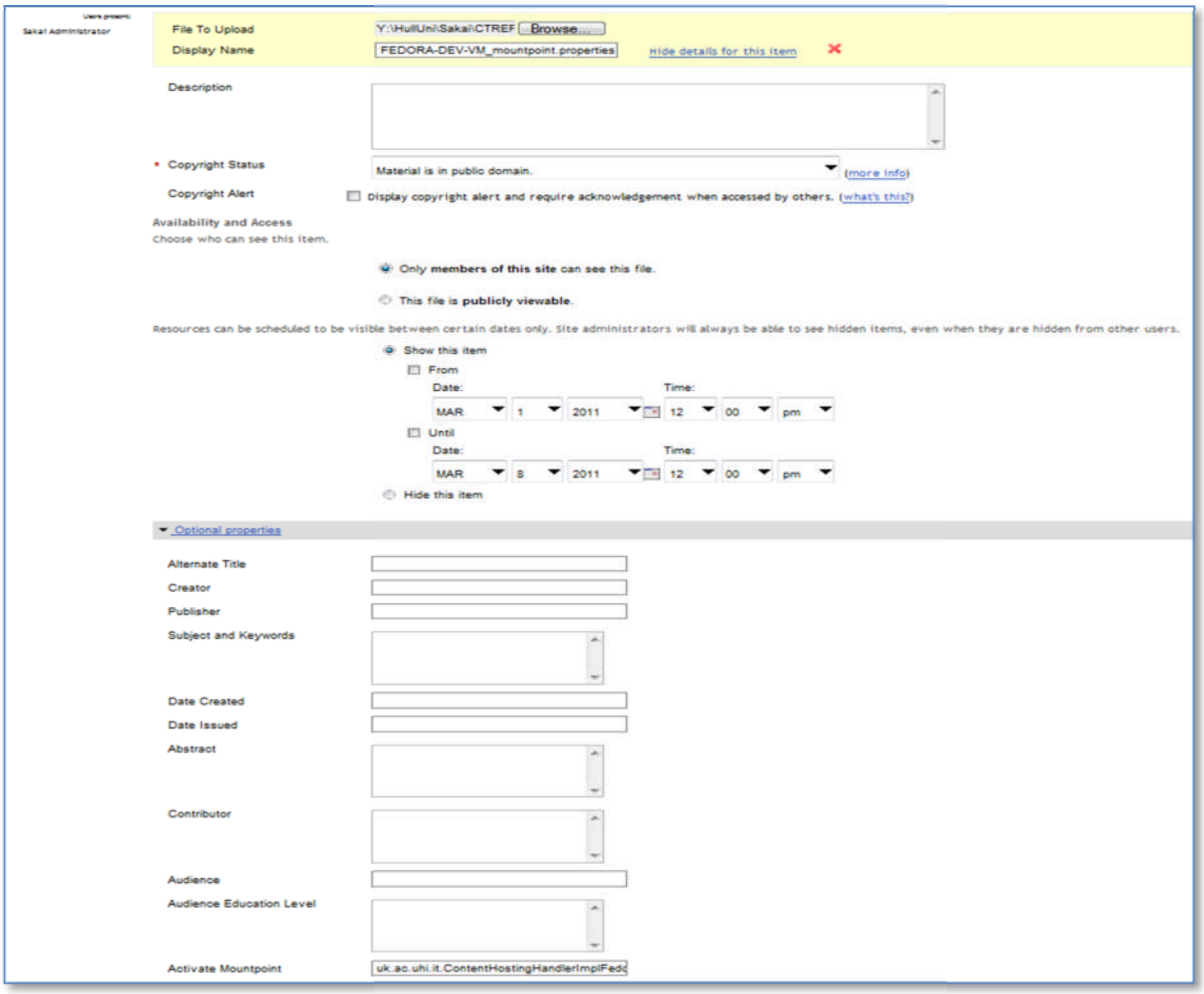
Sakai stores content in an internal database termed its 'repository' (not to be confused with the external institutional repository with which the CLIF tools will link). The resources tool in Sakai provides a means to manage content in the repository. When the resources tool is added to a worksite in Sakai, a separate "container" for that worksite's content is created in the Sakai repository. Learners in a particular workspace cannot see the content in another workspace unless permissions are explicitly granted for the content on an item-by-item basis.

Sakai provides many tools that create a learning workflow based on the content stored using the resources tool. For example, the assignment tool can link to content in resources. Announcements, tests, discussion postings, e-portfolios and messages can draw on content in a resources container.

The resources tool, as of Sakai 2.4 onwards, has made use of a pluggable architecture called Content Hosting Handlers. By uploading (mounting) a special descriptor file, a certain repository collection in the content repository can be made virtual, such that all access to content at that folder level or below will occur using a designated CHH handler.

2.2.2 CTREP Fedora Resource Tool Enhancement

Previous work done by the University of the Highlands and Islands (UHI), as part of the Cambridge TETRA Repositories Enhancement Project, produced a basic CHH implementation to allow a Fedora repository to be used to provide virtual resources. This work seemed to offer the CLIF Project a good potential starting point from which might be built a seamless and powerful way of integrating Sakai with Fedora. Following CLIF's work, the process of mounting a Fedora collection as a folder within the resource tool is very straightforward - one simply uploads the '.properties' file with the desired Fedora configuration values (refer to section 3.4.2) as a normal resource and set the activate mountpoint value to 'uk.ac.uhi.it.ContentHostingHandlerImplFedora' (see diagram below):



Sakai Administrator

File To Upload	Y:\HullUni\Sakai\CTREF
Display Name	FEDORA-DEV-VM_mountpoint.properties
Description	
* Copyright Status	Material is in public domain.
Copyright Alert	☐ Display copyright alert and require acknowledgement when accessed by others.

Availability and Access
Choose who can see this item.

● Only members of this site can see this file.
○ This file is publicly viewable.

Resources can be scheduled to be visible between certain dates only. Site administrators will always be able to see hidden items, even when they are hidden from other users.

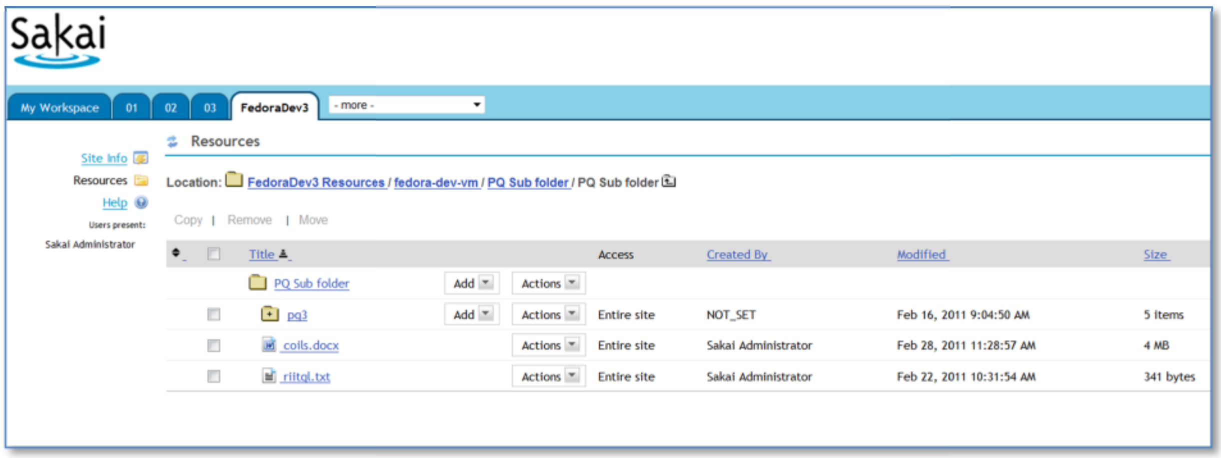
● Show this item
☐ From
Date: MAR 1 2011 Time: 12 00 pm
☐ Until
Date: MAR 8 2011 Time: 12 00 pm
○ Hide this item

Optional properties

Alternate Title	
Creator	
Publisher	
Subject and Keywords	
Date Created	
Date Issued	
Abstract	
Contributor	
Audience	
Audience Education Level	
Activate Mountpoint	uk.ac.uhi.it.ContentHostingHandlerImplFedo

Figure 1: Screenshot of how to 'mount' the Fedora CHH root folder

If all goes well, and Sakai is able to successfully communicate with Fedora, the resources tool will show the resources contained in the desired Fedora collection. The figure below is a screenshot taken from a CLIF-configured Sakai instance.



Location: FedoraDev3 Resources / fedora-dev-vm / PQ Sub folder / PQ Sub folder

Title			Access	Created By	Modified	Size
PQ Sub folder	Add	Actions				
pq3	Add	Actions	Entire site	NOT_SET	Feb 16, 2011 9:04:50 AM	5 items
coils.docx		Actions	Entire site	Sakai Administrator	Feb 28, 2011 11:28:57 AM	4 MB
riitql.txt		Actions	Entire site	Sakai Administrator	Feb 22, 2011 10:31:54 AM	341 bytes

Figure 2: Screenshot of a Fedora repository collection 'mounted' in the Sakai Resource Tool

Note one important change required by the CLIF CHH version of the UHI Fedora Handler: for the handler to operate correctly in all situations, the mountpoint properties file must end with the extension ".properties".

2.2.3 Functional requirements

It is important for the reader to appreciate that there is a very significant distinction to be made between 'normal' Sakai resources, which are essentially files, and virtual resources represented by Fedora objects, which are much more complex entities comprising a number of related datastreams only one of which generally represents the file (see Section 3.2 for more explanation of this). Thus the CLIF project needed to deal with three significant processes.

1. The structure of the Fedora repository needed to be represented on a Sakai screen to look like a conventional file tree which supported as much of the normal Sakai resource functionality as could be achieved.
    a. The repository should be organised using a hierarchical folder structure.
    b. Parts of the Fedora repository may be security restricted. In such a repository the CLIF Sakai component will function only below a specified node in the repository tree.
    c. The Fedora repository will be assumed to accept and work with Hydra-compliant objects. (The meaning of this compliance is discussed at section 3.2.1).
    d. In a secured repository (see 1b) users should not be able to change content that has been put in the repository.

2. Uploading a file to this virtual resources area required the creation of an appropriately structured Fedora object which would then be ingested into the repository.
    a. The system must be able to deposit content from a Sakai resources area to a folder in Fedora using the normal Sakai copy/move functionality.
    b. The system will allow users to enter basic metadata about the document. Fields in the metadata form should be pre-populated automatically where appropriate from the metadata available in the Sakai context.

   c. Optionally, it should be possible to configure the system to enable users to enter more detailed (MODS) metadata.

3. Downloading a Fedora-hosted virtual resource required routines to interrogate the repository and retrieve a copy of the file represented in the Fedora object.
   a. The system should enable a user to select a repository location to browse, navigate through the folder structure, and list the files contained in that repository folder.
   b. The system should enable retrieval of content that has been moved to Fedora to the user's local file system using normal Sakai copy/move functionality.
   c. When content is retrieved from Fedora to Sakai, only the content file is returned and not the metadata. The original copy of the content is retained in the repository.

3. Development and implementation

3.1 Approach

The integration of SharePoint and Fedora on one hand, and Sakai and Fedora on the other was implemented as separate development tasks. The common feature of the integration work was to implement a common Fedora content model based on the models developed in the Hydra project.[5] This would enable content to be moved from SharePoint to Sakai or vice versa, using the repository as a staging area, as required by several of the use cases.

The Sakai-Fedora integration was carried out starting from existing software written by the CTREP project. This provided the most direct route to the integration and avoiding reinventing the considerable basic development work already carried out. This decision was not as straightforward as it may appear and is discussed further in Section 3.5.1.

In an earlier CLIF report on our technical design[6] we evaluated the use of an Enterprise Service Bus (ESB) to provide a common interface between SharePoint, Sakai and Fedora. This option was discounted for several reasons including complexity.

For SharePoint-Fedora integration, we decided to implement a .NET middleware layer (based on previous internal work done at the University of Hull known as the Hydranet project) that provides a framework for the creation of simple Fedora objects as well as providing a wrapper for the Fedora web services that can be called from within SharePoint. On top of the middleware, we implemented extensions of SharePoint MySite to enable documents to be moved to and from the repository. The specific features we developed are on one hand grounded in the use cases, and on the other designed to be as generic as possible, to provide a reference implementation that can be adapted to support other use cases in the future.

The CLIF use cases covered both deposit of documents to public areas such as institutional publication repositories and archival deposit of university records and experimental data. The repository can offer both categories longer term preservation than may be expected within SharePoint or Sakai and can also support content that may have restricted access.

The SharePoint 2007 development was performed using Visual Studio 2008, although any recent version of Visual Studio would be suitable. The project was structured using the WSP Builder build and continuous integration tool that imposes a structure on the solution.

Due to time restrictions, we have developed a generic solution that supports a wide range of document formats (e.g. .docx, .xslx). More specific customisation to support Excel for instance could be performed to provide support for specific use cases.

We have also not attempted in CLIF to provide preservation tools. The development of such tools from scratch would have required considerably more effort than was available. SharePoint, in any case, provides for deploying format conversion services and repositories used with the CLIF tools may have their own associated preservation services.

[5] See: https://wiki.duraspace.org/display/hydra/The+Hydra+Project
[6] Awre c, Green R, Thompson A, Waddington S (2010) CLIF technical design See: https://edocs.hull.ac.uk/splash.jsp?parentId=hull:1647%26pid=hull:2697

3.2 Fedora configuration

The Fedora repository[7] is an open source repository platform now maintained by DuraSpace, a not-for-profit organisation in the US. Fedora defines a set of abstractions for expressing digital objects, asserting relationships among digital objects, and linking services to digital objects. The Fedora repository implements the Fedora abstractions through a core repository service, exposed as web-based services with well-defined APIs. In addition, Fedora provides an array of supporting services and applications. Fedora provides RDF support and the repository software is integrated with a semantic triple store technology, including an RDF database.

Fedora provides a flexible platform for developing repository services, rather than an end user application. Other projects such as Hydra and Islandora are developing fully integrated repository and preservation applications on top of the Fedora repository, but this is outside the scope of CLIF except to the extent that Hydra guidelines have been followed (as discussed below).

In CLIF, we are deploying Fedora "out-of-the-box" in order to maximise compatibility with existing systems. Accepting that Fedora repositories, and the digital objects in them, can be structured in many different ways the CLIF team chose to adopt the approach to object modelling recommended by the Hydra Project and a hierarchical set structure (often referred to as a collection structure) maintained through the use of RDF relationships. By taking this approach to use a pattern which is rapidly gaining wide support we hoped to make the CLIF Project software outputs more widely useful.

3.2.1 Hydra-compliant Fedora objects

Here is not the place to deal in detail with the potential complexity of objects within a Fedora repository. Suffice it to explain that a Fedora digital object comprises a number of so-called datastreams and Fedora implementations may structure these in a wide variety of ways. The CLIF team chose to adopt an emerging standard developed by the Hydra Project,[8] of which Hull is a founder member. A Hydra-compliant object is here represented in simplified form:

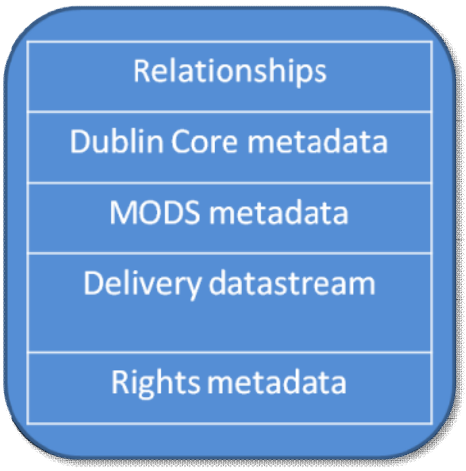


Figure 3: Simplified Hydra-compliant Fedora object

[7] See http://www.fedora-commons.org.
[8] See: http://projecthydra.org (in construction) and https://wiki.duraspace.org/display/hydra/The+Hydra+Project

For the purposes of CLIF, a Hydra-compliant Fedora object has a number of key datastreams:

- A datastream containing the object's relationships to other objects within the repository, represented using RDF. This is used, for example, to denote the collection structure of the repository and determine an object's place within that.
- A Dublin Core metadata stream containing very basic information about the object and intended solely for administration purposes.
- A datastream containing MODS metadata describing the digital content of the object.
- The delivery datastream itself which points to the source of the digital content: this may be managed by Fedora itself, as is the case in the new Hydra-compliant repository being developed at Hull, or may be managed by some external system in which case the datastream contains a URL which allows Fedora to fetch the content on demand.
- A rights metadata datastream which contains, represented in XML, the access permissions around the object and its content.

In practice there may be further datastreams but they are not germane to this discussion. Within a Fedora repository, objects are represented using Fedora Object XML (FOXML). Each object has a unique persistent identifier (pid) within the repository.

Thus, tools developed by CLIF to interact with a Fedora repository are not required simply to deposit and retrieve files. The deposit process requires the construction of a Fedora object FOXML file and ingest of this to the repository. Retrieval requires the use of Fedora's APIs to make appropriate service calls in order to retrieve the digital content and/or properties relating to it.

3.2.2 Repository structure

The Fedora objects in the repository are structured in a hierarchical manner. The objects generated by CLIF tools are classified as "folder" or "file" objects according to the following XML attributes in the RELS-EXT datastream:

- folder objects are characterised by the RDF statement

  ```
  <rel:isCollection xmlns:rel="info:fedora/fedora-system:def/relations-
  external#">
         True
  </rel:isCollection>
  ```

  (in this context the term 'collection' may be considered synonymous with 'set' or 'folder'.)

- and file objects by

  ```
  <rel:isCollection xmlns:rel="info:fedora/fedora-system:def/relations-
  external#">
         False
  </rel:isCollection>.
  ```

Folder objects do not contain any binary payload. In the case of SharePoint-Fedora integration a single folder object is created for each user as a root of their private repository folder; this occurs when the MySite for a given user is added by an administrator.

Relationships between Fedora objects and their containing folder are expressed by the RDF relation `isMemberOf`. In the snippet below of the RELS-EXT file of a Fedora object, the object is a member of the folder that is represented by a Fedora object with PID `CLIF:CERCH-LAPTOP-2-Administrator`:

```
<isMemberOf xmlns="info:fedora/fedora-system:def/relations-external#"
rdf:resource="info:fedora/CLIF:CERCH-LAPTOP-2-Administrator">
</isMemberOf>.
```

3.2.3 Fedora user account configuration

In order to enable creation of private repository folders for application users, individual Fedora user accounts are configured. The advantage of this approach compared to using a single Fedora user account is that it allows multiple systems to access content in the repository, enabling preservation of the appropriate permissions. Locking down to a single Fedora user account would require managing access through a single external application such as SharePoint or Sakai.

In the case of SharePoint, this process is carried out when a MySite is created. The user account creation is performed by the feature `CLIF.AddRepositoryUsers` that is linked to the MySite creation feature `CLIF.MySite` using the feature stapling feature `CLIF.MySiteStapler`.

Accounts in Fedora can be added to the file

```
C:\fedora\server\config\fedora-users.xml.
```

For the purposes for CLIF, we create the Fedora accounts by writing to this file directly, assuming that applications are running on the same server. In a production environment, additional security mechanisms would be required such as encrypting user account details and using secure transfer protocols to move user credentials across the network. Alternatively, and preferably, the system would be integrated with an institution’s security systems, for example an LDAP server.

3.2.4 Fedora XACML policies

Fedora can use XACML[9] policies to control access to objects in the repository. A Fedora-specific policy vocabulary is defined to enable the creation of XACML policies for Fedora repositories and digital objects. This vocabulary provides a set of identifiers (URNs) that can appear in XACML policies to refer to Fedora API operations (Actions in XACML), any aspects of a Fedora digital object (Resources in XACML), key attributes of the environment in which Fedora runs in (Environment in XACML), and common subject (i.e., user) attributes. There are two modes of operation for XACML policies:

1. Repository-wide policies: This is the default mode for managing access in Fedora. Such policies are used, for instance, to control access to Fedora API operations.
2. Finer-grained object-specific policies which can be used to control access to individual digital objects or collections of objects.

Policies in mode 1 are typically employed in public repositories where there are uniform access requirements to content items. Policies in mode 2 are object-specific policies that refer to one

[9] https://wiki.duraspace.org/display/FCR30/XACML+Policy+Enforcement

particular digital object. An object-specific policy is stored in the "POLICY" datastream of the digital object to which it applies.

In the CLIF SharePoint integration, we needed to control access to content in both private and public repository folders. We have not specified the exact form that an XACML policy should take, since that will depend on the specific deployment context.

Within the SharePoint integration, method 2 above of assigning policies to digital objects has been extended to object collections. The policy for a folder object in Fedora can be added as a POLICY datastream to that object. All objects in that folder are provided with a POLICY datastream that uses the Fedora Externally Referenced mode to point to the folder. In this manner multiple digital objects can point to the same policy. In the CLIF SharePoint integration we have included user-specific policies in the root folder of the user’s private folder area as well as more general policies for each of the public repository folders.

3.3 SharePoint-Fedora Integration

3.3.1 Technical overview

The SharePoint MySite template was chosen as the base site template for the implementation of CLIF. As a dedicated personal site, MySite provides users with a single location to manage all of the documents, content, and tasks that they have in any site. Content and documents can also be shared with other users. It enables users to create their own workspaces and share selected personal information with other users. MySite provides functionality for management of documents, tasks, links, calendar and contacts. Privacy groups allow users to specify permissions to access information on shared pages.

Much of the CLIF functionality is independent of the site template chosen and could equally well be deployed on team sites and other forms of SharePoint site.

3.3.2 Integration architecture

Figure 4 illustrates the relationship between the main components in the SharePoint-Fedora integration.

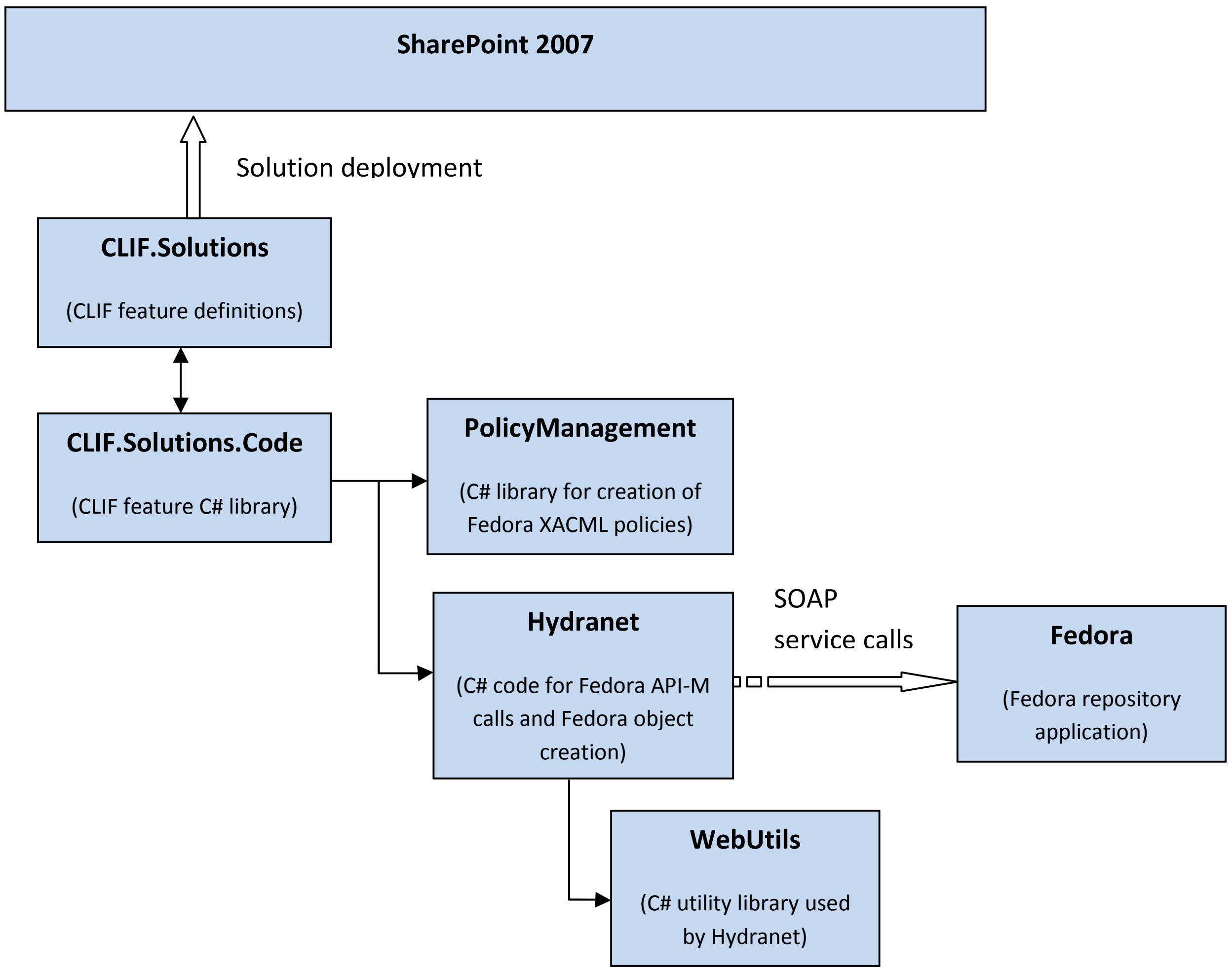
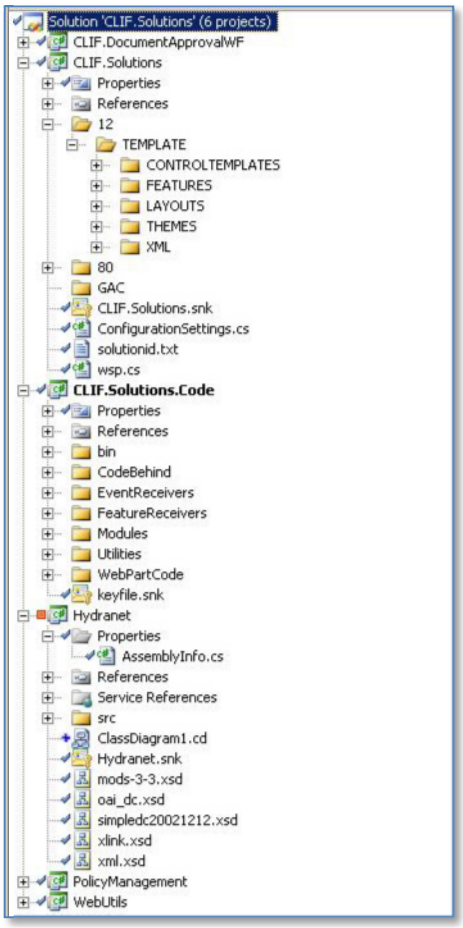

SharePoint 2007
Solution deployment
CLIF.Solutions
(CLIF feature definitions)
CLIF.Solutions.Code
(CLIF feature C# library)
PolicyManagement
(C# library for creation of Fedora XACML policies)
Hydranet
(C# code for Fedora API-M calls and Fedora object creation)
SOAP service calls
Fedora
(Fedora repository application)
WebUtils
(C# utility library used by Hydranet)

Figure 4: Integration architecture for the CLIF SharePoint-Fedora solution

The CLIF solution comprises a CLIF.Solutions project that contains the feature definitions and other configuration information. The "code behind" and other software is contained in a separate project CLIF.Solutions.Code. This project interacts with the other custom projects. PolicyManagement contains the code for creating Fedora XACML policies. Hydranet is a project which contains calls to the Fedora API-M interface via the SOAP protocol, as well as performing Fedora object creation. WebUtils is a project that contains utility code.

3.4 Solution structure

Figure 5 illustrates the structure of the CLIF solution in Visual Studio 2008.



```
Solution 'CLIF.Solutions' (6 projects)
  CLIF.DocumentApprovalWF
  CLIF.Solutions
    Properties
    References
    12
      TEMPLATE
        CONTROLTEMPLATES
        FEATURES
        LAYOUTS
        THEMES
        XML
    80
    GAC
    CLIF.Solutions.snk
    ConfigurationSettings.cs
    solutionid.txt
    wsp.cs
  CLIF.Solutions.Code
    Properties
    References
    bin
    CodeBehind
    EventReceivers
    FeatureReceivers
    Modules
    Utilities
    WebPartCode
    keyfile.snk
  Hydranet
    Properties
      AssemblyInfo.cs
    References
    Service References
    src
    ClassDiagram1.cd
    Hydranet.snk
    mods-3-3.xsd
    oai_dc.xsd
    simpledc20021212.xsd
    xlink.xsd
    xml.xsd
  PolicyManagement
  WebUtils
```

Figure 5: CLIF SharePoint solution structure

The solution has been developed using the WSPBuilder build and continuous integration tool. The solution comprises six separate projects.

- **CLIF.DocumentApprovalWF** contains the code to implement the approval workflow for publication of documents to the repository.
- **CLIF.Solutions** defines the features and look and feel of the site.
    - The FEATURES folder contains the feature definition files comprising the elements.xml and feature.xml files.
    - The GAC folder contains the DLL files that are required for the solution. Building the C# code projects automatically deposits the resulting DLLs into the GAC folder.
- **CLIF.Solutions.Code** contains the C# code for the features defined in the CLIF.Solutions project. This includes "code behind" classes for the ASP.NET pages, event and feature receivers and web part code.
- **Hydranet** contains the C# middleware to interface with the Fedora API-A and API-M web service interfaces. It also includes code for Fedora object creation using the FOXML schema.
- **PolicyManagement** implements the creation of the POLICY datastreams that are used to implement access control in Fedora.

- **WebUtils** is a set of utilities that are used by the Hydranet project.

The CLIF.Solutions project implements features in the SharePoint 12-hive, which enables deployment at the web application level. The SharePoint features that have been developed for CLIF are contained in the CLIF.Solutions\12\TEMPLATE\FEATURES and are described in detail in Section 3.4.3.

The WSPBuilder tool generates a single WSP file, which packages all the required files and greatly simplifies the deployment of the developed solution to a SharePoint instance.

In order to facilitate the deployment of the solution, PowerShell scripts were developed in the PowerShell GUI script editor, which can be run to deploy the CLIF solution and activate the features.

```
$solutionPath="C:\Projects\CLIF.Solutions\CLIF.Solutions\CLIF.Solutions.
wsp"
$solutionName="CLIF.Solutions.wsp"

cls

Write-Host "Please wait upgrading solution ..."
stsadm -o upgradesolution -name $solutionName -filename

$solutionPath -immediate -allowCasPolicies -allowGacDeployment
stsadm -o execadmsvcjobs

Write-Host "Solution Upgraded Sucessfully !!" -ForegroundColor Green -
BackgroundColor Yellow
```

Figure 6: PowerShell script UpgradeSolution.ps1 for CLIF solution deployment

Solution deployment is performed by UpgradeSolution.ps1, as illustrated in Figure 6 above. This script removes any existing version of the CLIF solution before deploying the new version. Feature activation is carried out by ActivateFeatures.ps1 as illustrated below.

```
$siteUrl = "http://clif/"
cls
# create array of new Fields
$Activefeatures=  @(
"CLIF.XSL|d9394c71-006d-4594-85b6-6c1006785a46",
"CLIF.Scripts|B0A0A13D-FC80-408e-8F61-E36FEFC29F98",
"CLIF.Styles|b80cce94-386b-4124-812a-f9c4565afd01",
"CLIF.Images|b65e6fd8-4874-4eaa-93c3-c16447343aae",
"CLIF.MasterPage|7306b8a0-7490-4bfc-9756-9b0a31d5065c",
"CLIF.Search|c683eef0-a0d9-479d-886c-bf6ae02817ee"
"CLIF.Fields|8fcd6181-9fbd-49ad-9983-a6f483ceb4c6",
"CLIF.ContentTypes|7d7aefc4-8494-4ce7-875a-ff08212dd403",
"CLIF.LookUpList|52fb0409-3619-4b04-a7dc-477bfff63367",
"CLIF.LookUpWithPicker|10c19a31-9710-42d9-a1f2-4ac3f4aabb2d",
"CLIF.Lists|59d73371-02cb-4b45-a203-cd7e4904f27b"
)
# create array of new Fields
$DeActivefeatures=  @(
#"CLIF.Scripts|B0A0A13D-FC80-408e-8F61-E36FEFC29F98",
#"CLIF.Styles|b80cce94-386b-4124-812a-f9c4565afd01",
#"CLIF.Images|b65e6fd8-4874-4eaa-93c3-c16447343aae",
#"CLIF.MasterPage|7306b8a0-7490-4bfc-9756-9b0a31d5065c"
#"CLIF.Fields|8fcd6181-9fbd-49ad-9983-a6f483ceb4c6",
#"CLIF.ContentTypes|7d7aefc4-8494-4ce7-875a-ff08212dd403",
#"CLIF.Lists|59d73371-02cb-4b45-a203-cd7e4904f27b"
)
$totalFeatures=$Activefeatures.Length
$featureCounter=1
$activity=""
foreach($feature in $DeActivefeatures)
{
      $value=$feature.Split("|");
      $activity= "Please wait de-activating feature ["  + $value[0] + "]"
      write-progress -Activity $activity  -Status "% Complete:" -
PercentComplete ($featureCounter/$totalFeatures*100);
      stsadm -o deactivatefeature -id $value[1] -url $siteUrl
      $featureCounter++;
}
$featureCounter=1
foreach($feature in $Activefeatures)
{
      $value=$feature.Split("|");
      $activity= "Please wait activating feature ["  + $value[0] + "]"
    write-progress -Activity $activity   -Status "% Complete:" -
PercentComplete ($featureCounter/$totalFeatures*100);
      stsadm -o activatefeature -id $value[1] -url $siteUrl
      Write-Host "[" $value[0] "]-  activated sucessfully!" -ForegroundColor
White -BackgroundColor Green
      $featureCounter++;
}
Write-Host "****************************************" -BackgroundColor
DarkGreen -ForegroundColor White
Write-Host "Script Execution Completed." -BackgroundColor DarkGreen -
ForegroundColor White

Write-Host "****************************************" -BackgroundColor
DarkGreen -ForegroundColor White
```

Figure 7: PowerShell script Activate Features.ps1 for CLIF feature activation

3.4.1 Site map

Figure 8 illustrates the structure of the CLIF root site, including the custom lists developed and the associated columns.

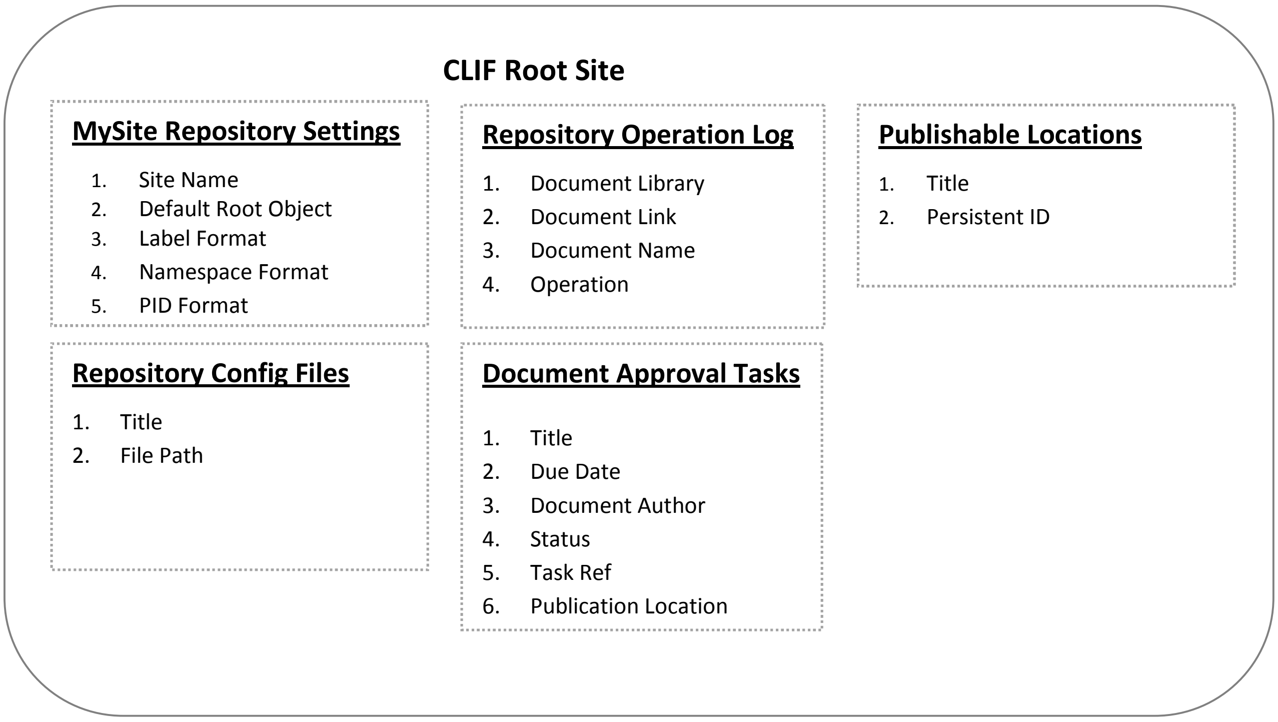
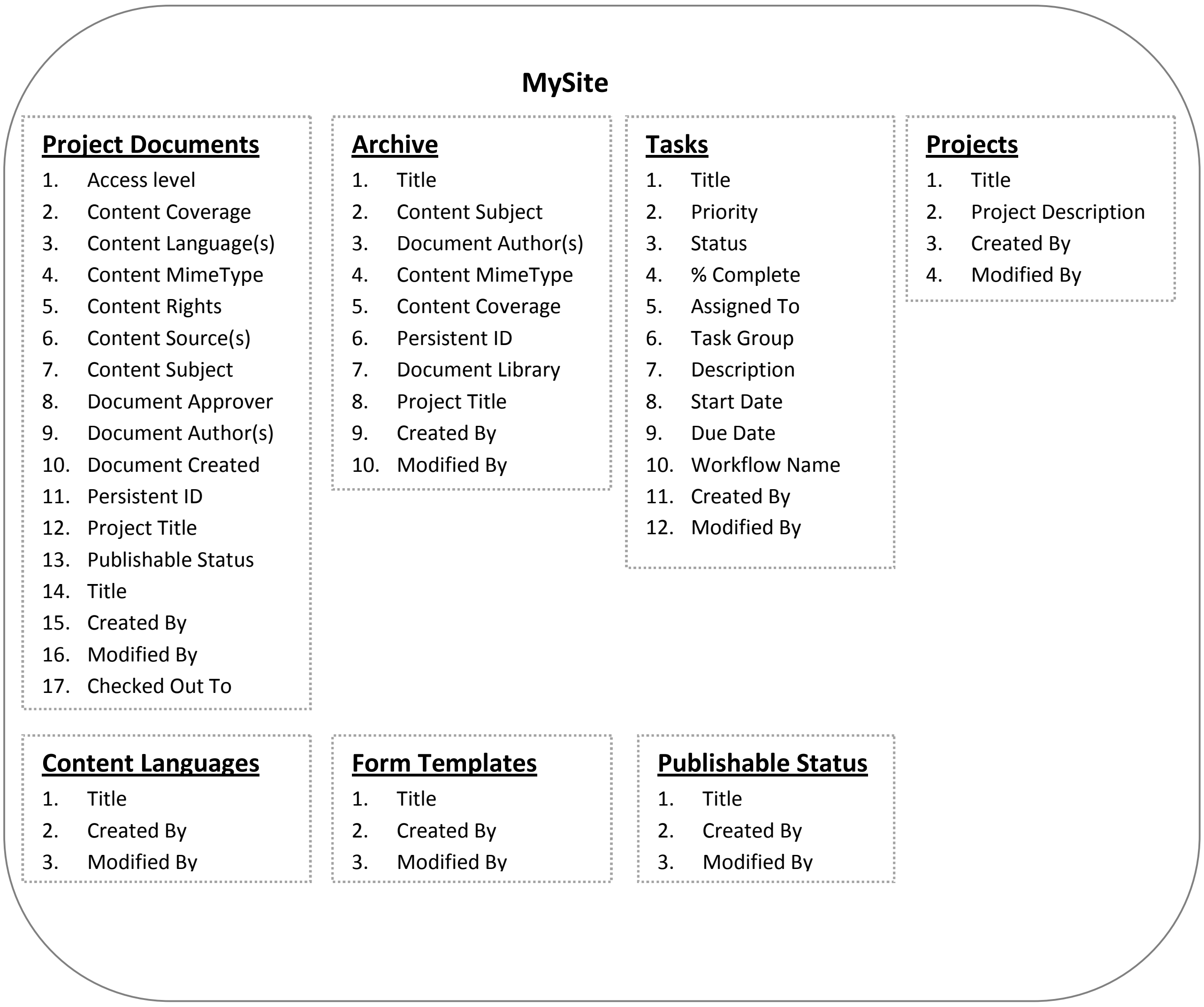


Figure 8: CLIF Root Site map

The structure of the end user MySite is illustrated in Figure 9.



MySite

Project Documents

1. Access level
2. Content Coverage
3. Content Language(s)
4. Content MimeType
5. Content Rights
6. Content Source(s)
7. Content Subject
8. Document Approver
9. Document Author(s)
10. Document Created
11. Persistent ID
12. Project Title
13. Publishable Status
14. Title
15. Created By
16. Modified By
17. Checked Out To

Archive

1. Title
2. Content Subject
3. Document Author(s)
4. Content MimeType
5. Content Coverage
6. Persistent ID
7. Document Library
8. Project Title
9. Created By
10. Modified By

Tasks

1. Title
2. Priority
3. Status
4. % Complete
5. Assigned To
6. Task Group
7. Description
8. Start Date
9. Due Date
10. Workflow Name
11. Created By
12. Modified By

Projects

1. Title
2. Project Description
3. Created By
4. Modified By

Content Languages

1. Title
2. Created By
3. Modified By

Form Templates

1. Title
2. Created By
3. Modified By

Publishable Status

1. Title
2. Created By
3. Modified By

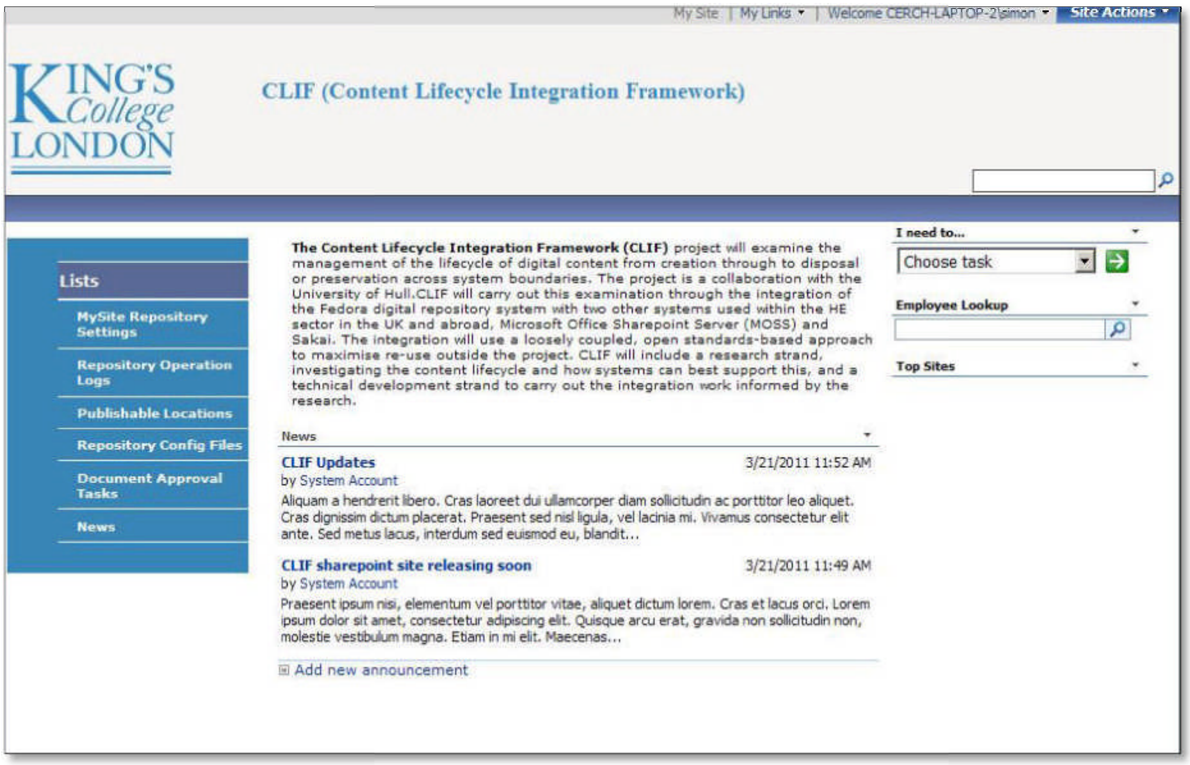
Figure 9: MySite map

3.4.2 CLIF root site functionality

The CLIF root site implements requirements R5.1-R5.4

This top-level site allows administration level users to define configuration settings relating to each MySite created and assigned to a user. This site assigns a predefined location for each user where they can upload file(s) and create folder(s).It also allows assigning document approvers to each publishable location in the repository.

The home page of the CLIF root site is illustrated in Figure 10.



CLIF (Content Lifecycle Integration Framework)

The Content Lifecycle Integration Framework (CLIF) project will examine the management of the lifecycle of digital content from creation through to disposal or preservation across system boundaries. The project is a collaboration with the University of Hull.CLIF will carry out this examination through the integration of the Fedora digital repository system with two other systems used within the HE sector in the UK and abroad, Microsoft Office Sharepoint Server (MOSS) and Sakai. The integration will use a loosely coupled, open standards-based approach to maximise re-use outside the project. CLIF will include a research strand, investigating the content lifecycle and how systems can best support this, and a technical development strand to carry out the integration work informed by the research.

News

CLIF Updates 3/21/2011 11:52 AM
by System Account
Aliquam a hendrerit libero. Cras laoreet dui ullamcorper diam sollicitudin ac porttitor leo aliquet. Cras dignissim dictum placerat. Praesent sed nisl ligula, vel lacinia mi. Vivamus consectetur elit ante. Sed metus lacus, interdum sed euismod eu, blandit...

CLIF sharepoint site releasing soon 3/21/2011 11:49 AM
by System Account
Praesent ipsum nisi, elementum vel porttitor vitae, aliquet dictum lorem. Cras et lacus orci. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quisque arcu erat, gravida non sollicitudin non, molestie vestibulum magna. Etiam in mi elit. Maecenas...

Figure 10: CLIF root site homepage.

The basic features of the CLIF root site are as follows.

- It enables user management including creation of new MySite sites.
- It stores configuration settings for MySites.
- For each user, it enables creation of a user accounts in Fedora and allocates private user folders. This is performed automatically on MySite creation.
- It defines the publication location for document libraries in MySite.
- It defines persistent ID and namespace formats when files are copied to Fedora.
- It allocates document approvers for each publishable location.
- It enables user management including creation of new MySite sites.

The various configuration settings can be configured by editing the lists in the left toolbar of the root site, as illustrated in Figure 10.

3.4.3 Deposit to repository

This section describes the implementation of the requirements R2.1-R2.10 involving deposit from SharePoint to Fedora.

MySite is an area where an end user can manage documents in SharePoint and Fedora. It allows uploading document as well as creating container content objects (folders) within their predefined user area in Fedora.

There are three mechanisms for depositing individual documents from SharePoint MySite to Fedora. These are

- Publish to Repository.
- Copy to Repository.
- Move to Repository.

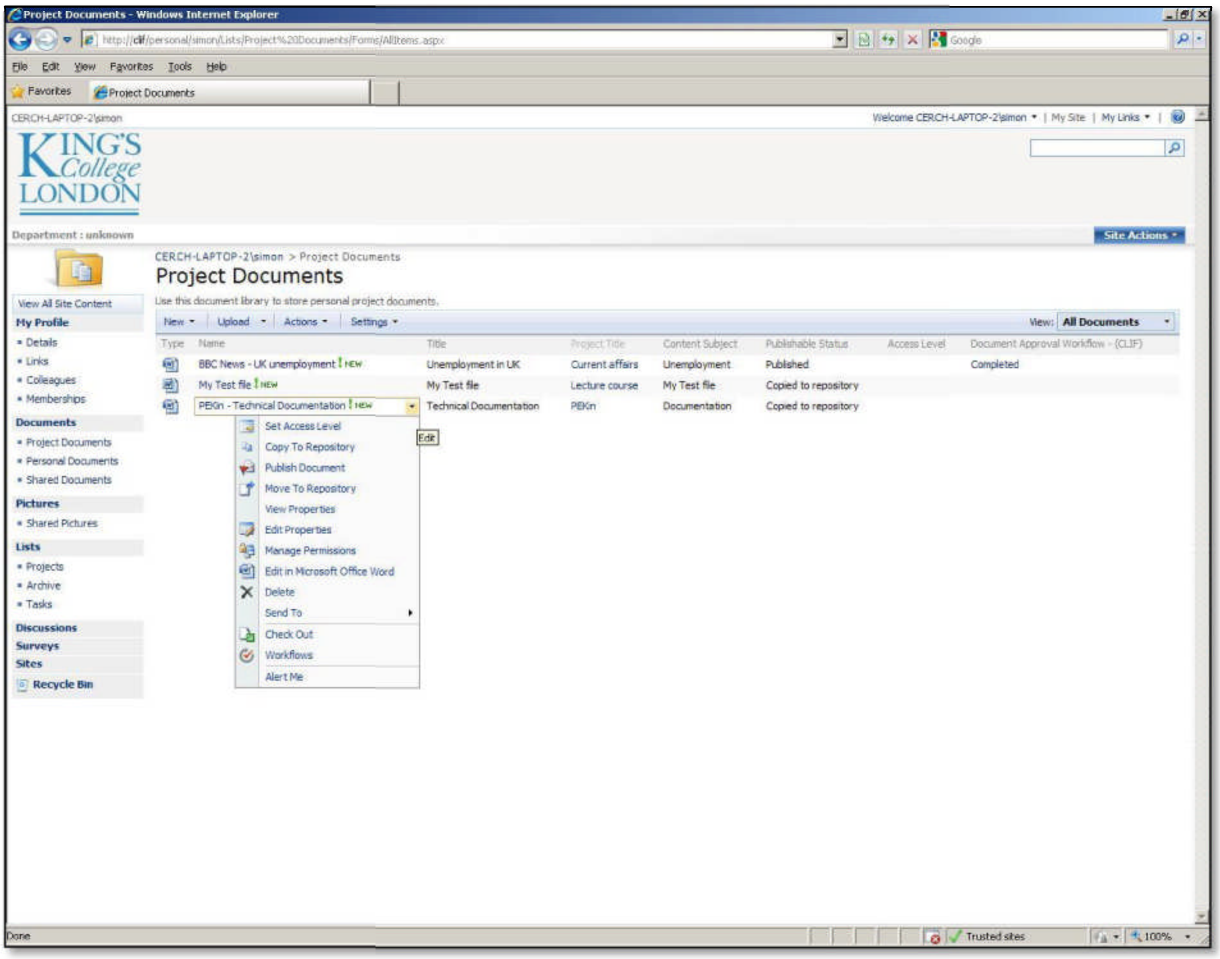
Additionally, there are two bulk operations:

- Copy Multiple Files.
- Move Multiple Files.

These operations enable deposit of multiple documents from a document library, which are selectable using check boxes, to the repository. These are analogous to the corresponding operations for individual documents.

An additional feature called “Set Access Level” is provided in the menu to enable sharing of personal documents with the user group.

Deposit of an individual document can be enabled by selecting the pop-up menu on the document tab in the Project Documents list as illustrated in Figure 11.



CERCH-LAPTOP-2\simon > Project Documents

Project Documents

Use this document library to store personal project documents.

Type	Name	Title	Project Title	Content Subject	Publishable Status	Access Level	Document Approval Workflow - (CLIF)
	BBC News - UK unemployment ! NEW	Unemployment in UK	Current affairs	Unemployment	Published		Completed
	My Test file ! NEW	My Test file	Lecture course	My Test file	Copied to repository		
	PEKin - Technical Documentation ! NEW	Technical Documentation	PEKin	Documentation	Copied to repository		

Figure 11: Pop-up menu with options for depositing item to repository.

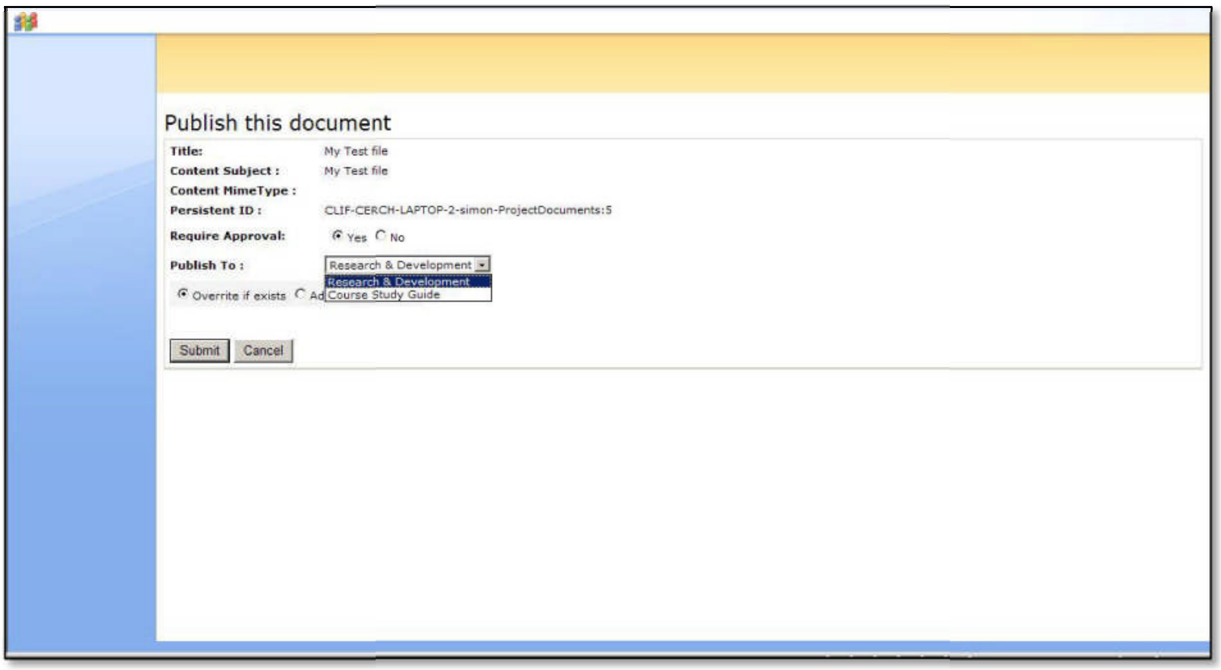
Once a document has been deposited to the repository through either Publish to Repository or Copy to Repository, the Publishable Status column in the Project Documents list is updated appropriately.

Publish to Repository

Publish to Repository (requirement R2.3) enables deposit of a document and metadata to a public repository folder. The list of publishable locations is configured by an administrator in the root site and is made visible to a MySite user via a pull-down menu.

Publish to Repository provides the user with the option to initiate an approval workflow (requirement R2.2) by selecting the radio button provided on the web form. Once submitted by the user, an approval task is created for the approver. The task appears on the MySite homepage of the approver. Additionally, if an email server has been configured, the approver receives a notification by email of the pending task. The approver has the option to approve or reject the document as well as providing comments in a text box for the submitter. Once the review task has been completed, the submitter can review the task status on their MySite homepage, as well as receiving an email notification.

Figure 12 illustrates the form displayed to the user prior to submission of the document that enables configuration of the publishable location and selection of the approval requirement.



Publish this document

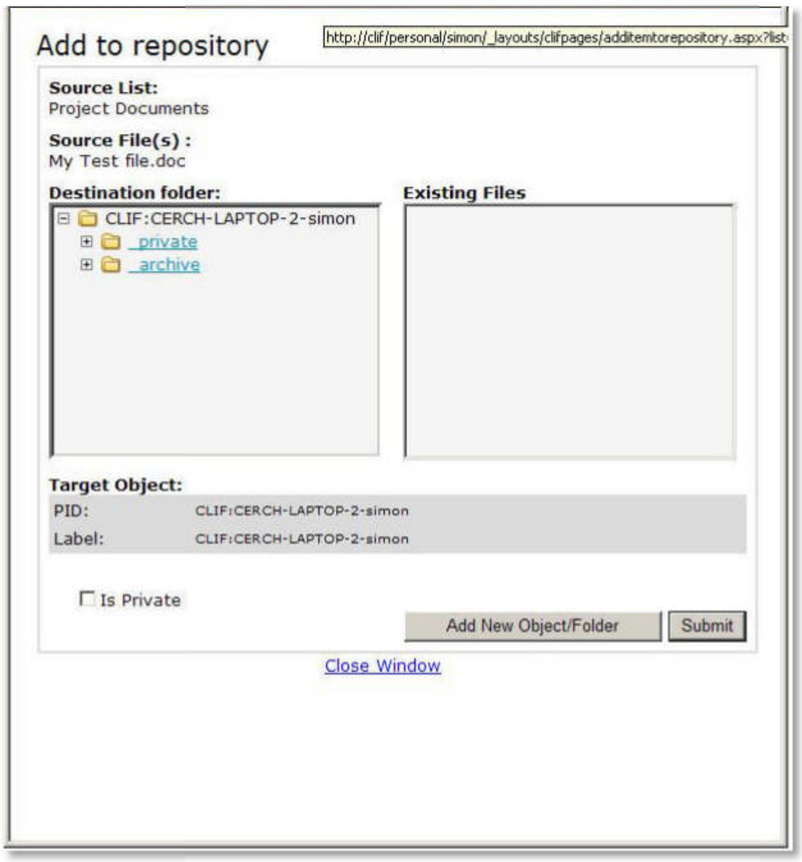
Title:	My Test file
Content Subject :	My Test file
Content MimeType :	
Persistent ID :	CLIF-CERCH-LAPTOP-2-simon-ProjectDocuments:5
Require Approval:	[✓ Yes] No
Publish To :	Research & Development

[✓ Overrite if exists] Ad

Figure 12: Form for Publish to Repository

Copy to Repository

Copy to Repository (requirement R2.6) enables the user to deposit a copy of a document from a SharePoint document library to a private folder in Fedora together with associated metadata. Copy to Repository provides a window where the user can browse to the appropriate folder, create new folders and browse existing files. This is illustrated in Figure 13. A copy of the original document is retained in the SharePoint document library. If Copy to Repository is carried out more than once on the same document, this results in multiple objects being created in the repository.


Add to repository

http://clif/personal/simon/_layouts/clifpages/additemtorepository.aspx?list

Source List:
Project Documents

Source File(s) :
My Test file.doc

Destination folder:

CLIF:CERCH-LAPTOP-2-simon
_private
_archive

Existing Files

Target Object:

PID:	CLIF:CERCH-LAPTOP-2-simon
Label:	CLIF:CERCH-LAPTOP-2-simon

☐ Is Private

Figure 13: Copy to Repository – browse to private repository location

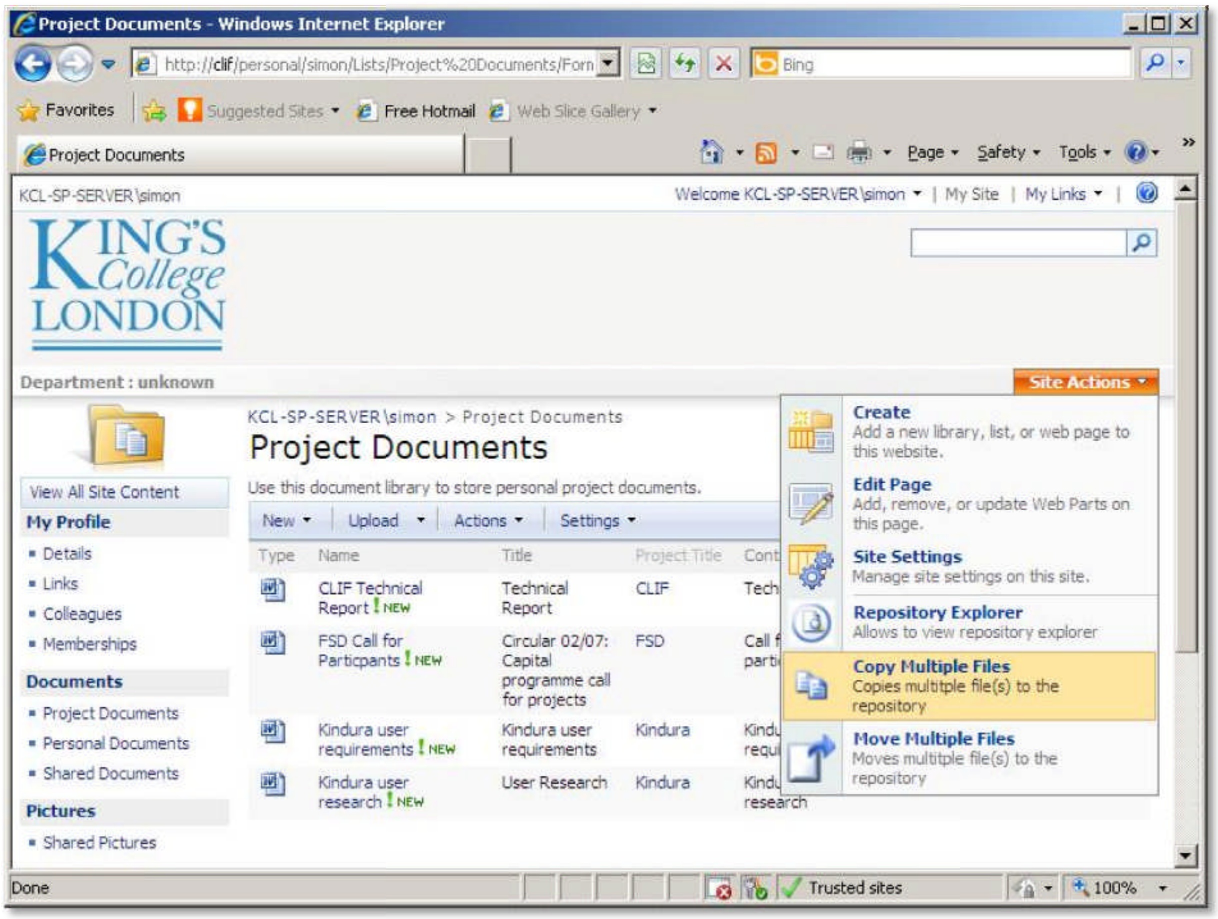
The folders '_private' and '_archive' are configured as default. Such default folders can be configured by a system administrator depending on the specific user context.

Move to Repository

Move to Repository (requirement R2.7) also enables deposit of documents to the repository. In contrast to Copy to Repository, the document file is not retained in SharePoint. A page analogous to Figure 13 is presented to the user from which they can browse to a specific repository location. When the document is moved to Fedora, an entry is made in the Archive list, which can be accessed through the left side bar of MySite.

Copy Multiple Files

Copying and moving multiple documents from a SharePoint library to Fedora can be achieved through the Site Actions pop-up menu in the document library. This is illustrated in Figure 14.



Project Documents - Windows Internet Explorer

http://clif/personal/simon/Lists/Project%20Documents/Forn

KCL-SP-SERVER\simon

Welcome KCL-SP-SERVER\simon | My Site | My Links

Department : unknown

Site Actions

View All Site Content

My Profile
- Details
- Links
- Colleagues
- Memberships

Documents
- Project Documents
- Personal Documents
- Shared Documents

Pictures
- Shared Pictures

KCL-SP-SERVER\simon > Project Documents

Project Documents

Use this document library to store personal project documents.

New | Upload | Actions | Settings

Type	Name	Title	Project Title	Cont
	CLIF Technical Report ! NEW	Technical Report	CLIF	Tech
	FSD Call for Particpants ! NEW	Circular 02/07: Capital programme call for projects	FSD	Call f parti
	Kindura user requirements ! NEW	Kindura user requirements	Kindura	Kindu requi
	Kindura user research ! NEW	User Research	Kindura	Kindu research

Create Add a new library, list, or web page to this website.

Edit Page Add, remove, or update Web Parts on this page.

Site Settings Manage site settings on this site.

Repository Explorer Allows to view repository explorer

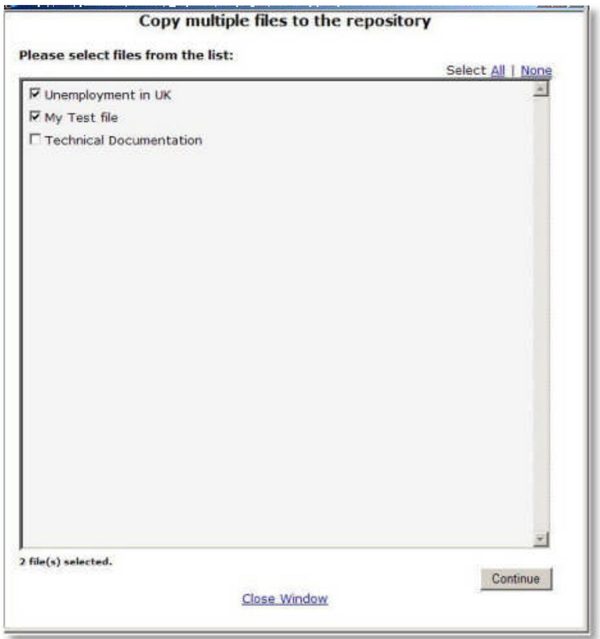
Copy Multiple Files Copies multitple file(s) to the repository

Move Multiple Files Moves multitple file(s) to the repository

Done | Trusted sites | 100%

Figure 14: Deposit of multiple documents to the repository.

Copy Multiple Files enables the user to select multiple files from a separate window in their browser by the use of check boxes as illustrated in Figure 15.



Copy multiple files to the repository

Please select files from the list:

Select All | None

- [x] Unemployment in UK
- [x] My Test file
- [ ] Technical Documentation

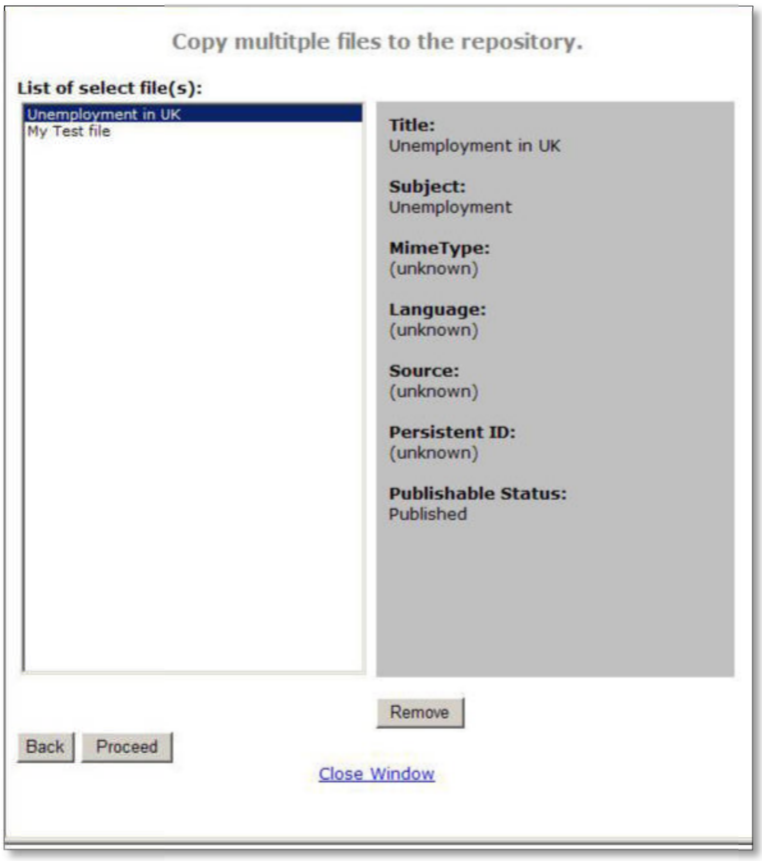
2 file(s) selected.

Continue

Close Window

Figure 15: Selection of multiple files to be copied to repository.

Once the files have been selected and the Continue button selected, the user can review the files selected for deposit together with the associated metadata, as illustrated in Figure 16.


Copy multitple files to the repository.

List of select file(s):

Unemployment in UK
My Test file

Title:
Unemployment in UK

Subject:
Unemployment

MimeType:
(unknown)

Language:
(unknown)

Source:
(unknown)

Persistent ID:
(unknown)

Publishable Status:
Published

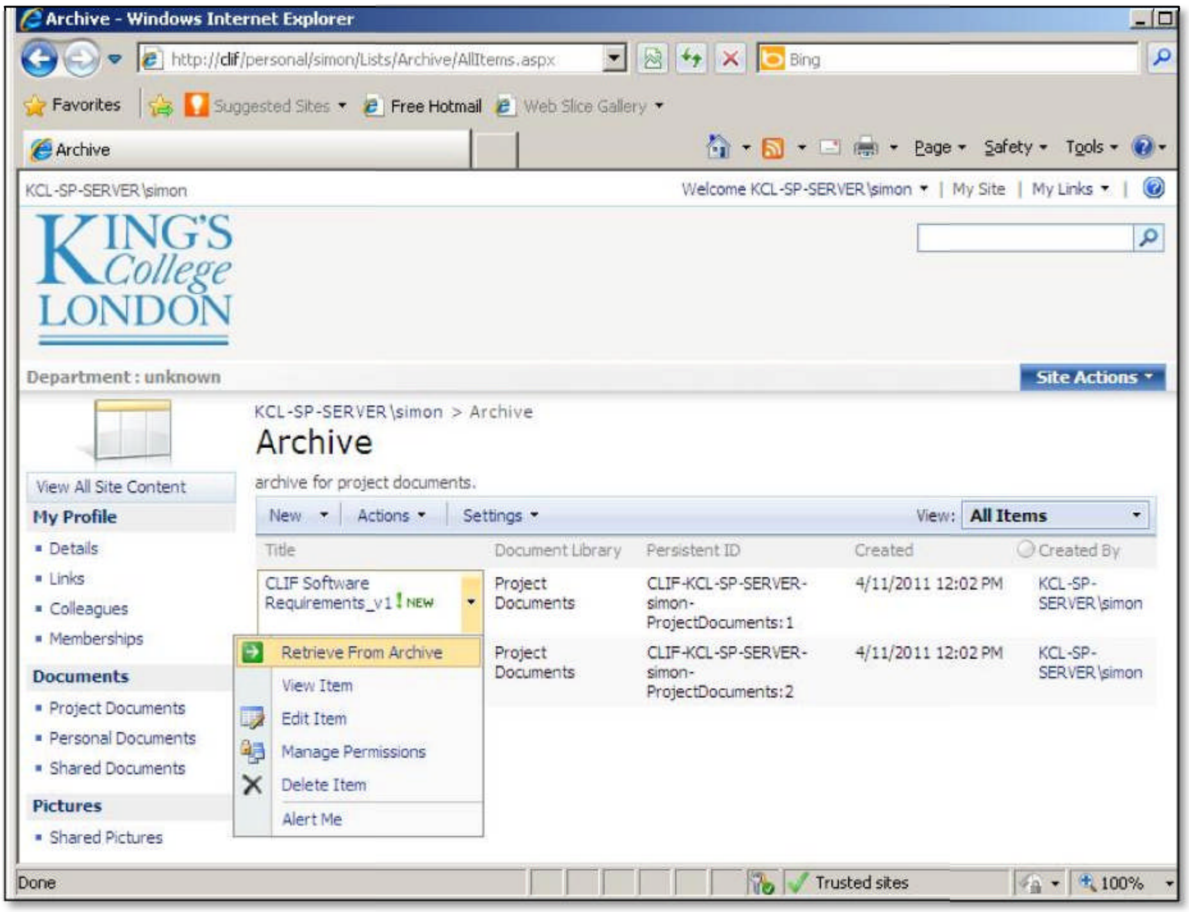
Figure 16: Selected files for Copy Multiple Files to repository

Move Multiple Files

Move Multiple Files performs an analogous operation to Move to Repository, but for multiple files. As with Copy Multiple Files, it is selectable from the Site Actions menu on the document library page as in Figure 11. Move Multiple Files allows selection of multiple files from a document library for deposit to the repository, analogous to Figures 14 and 15. Links to the files are placed in the Archive list that is accessible from the left side bar of the MySite document library.

3.4.4 Retrieval from repository

Move to Repository and Move Multiple Files enable deposit of documents from SharePoint to Fedora and removal of the documents from the SharePoint MySite, as described above. Such documents can be retrieved (requirement R3.1) by navigating to the Archive list from the left side bar of the document library, selecting the pop-up menu from the required document and clicking on Retrieve from Archive. This feature is illustrated in Figure 17.



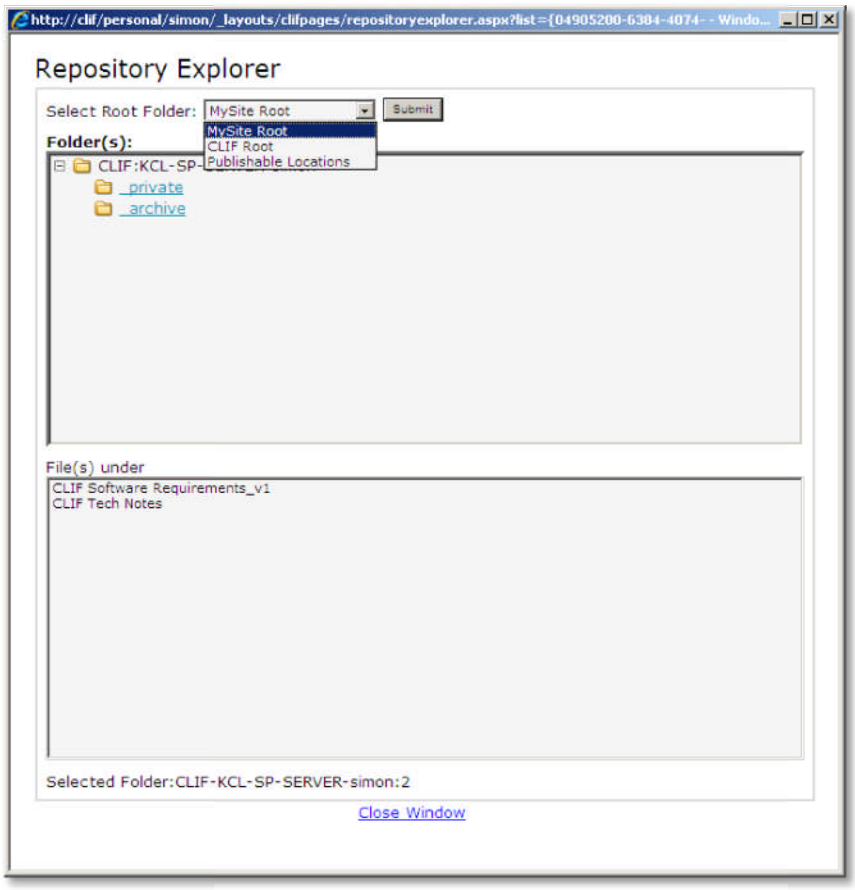
Title	Document Library	Persistent ID	Created	Created By
CLIF Software Requirements_v1 ! NEW	Project Documents	CLIF-KCL-SP-SERVER-simon-ProjectDocuments:1	4/11/2011 12:02 PM	KCL-SP-SERVER\simon
	Project Documents	CLIF-KCL-SP-SERVER-simon-ProjectDocuments:2	4/11/2011 12:02 PM	KCL-SP-SERVER\simon

Figure 17: Retrieve from archive

3.4.5 Repository Explorer

Repository Explorer allows browse of the Fedora repository from a pop-up window. This option is selectable from the Site Actions tab of the MySite document library as illustrated in Figure 14. This feature implements requirement R3.4. The Repository Explorer window is shown is Figure 18.

The user has the option to select the root of the repository folder tree they wish to browse. The options to browse from the MySite root folder and the Publishable Locations are standard for every user. If the user has administrative rights, they can also navigate the CLIF root folder.


http://clif/personal/simon/_layouts/clifpages/repositoryexplorer.aspx?list={04905200-6384-4074- - Windo...
Repository Explorer
Select Root Folder: MySite Root
Submit
MySite Root
CLIF Root
Publishable Locations
Folder(s):
CLIF:KCL-SP-
_private
_archive
File(s) under
Close Window

CLIF Software Requirements_v1
CLIF Tech Notes

Selected Folder:CLIF-KCL-SP-SERVER-simon:2

Figure 18: Repository Explorer

3.4.6 Search repository

The CLIF solution includes configuration of the SharePoint web crawler that enables indexing of all metadata in the CLIF web application including:

1. Structured metadata associated with documents in MySite document library.
2. Full-text indexing of documents stored in SharePoint.
3. Structured metadata associated with documents stored in Fedora that have been deposited via Move to Repository.

Option 3 is enabled since metadata associated with documents that have been moved to Fedora is retained in SharePoint. However, SharePoint automatically removes the entries in its search index of documents that have been deleted from SharePoint.

Configuration of the web crawler is performed from the SharePoint administrative interface under the Shared Services Configuration option.

Requirements R4.1 and R4.2 from section 2.1.4 were implemented. Users can enter search queries in the text box in the right hand corner of the MySite document library as illustrated in Figure 11. Search results are displayed in SharePoint in the standard manner. In case documents have been moved to Fedora, the document file can be accessed via a hyperlink in the search result.

In order to enable full text searching of documents in Fedora, an indexing service such as Solr is required that runs within the Tomcat instance of Fedora. Due to time limitations, it was not feasible to implement this capability within the CLIF project.

3.4.7 Excel calculation

Although we have not written any custom features for Excel calculation due to time limitations, we have configured Excel services to enable calculations to be performed on a spreadsheet stored at a trusted location. This method allows an administrator to place a reference spreadsheet at a location in SharePoint configured to be a trusted location. An end user can upload a spreadsheet with input data and perform calculations on the reference spreadsheet. The resulting spreadsheet is stored in the SharePoint document library, and can be deposited to the Fedora repository using the CLIF features.

3.4.8 SharePoint features

This section lists all the SharePoint features that were developed during the CLIF project.

No.	Sharepoint Feature	Description
1.	CLIF.AddRepositoryUsers	Adds a user to the Fedora repository users via the Fedora-users.xml file.
2..	CLIF.AddToRepository	Adds a new ECB menu to all document libraries that allows users to add selected file(s) to the repository.
3.	CLIF.AddToRepositoryAdvanced	Adds MODS form template and advanced options for adding files to the repository.
4.	CLIF.ApproversTask	Creates an approval task control for the workflow associated to the publish to repository function.
5.	CLIF.Archive	Adds ECB menu to archive documents in the library.
6.	CLIF.BulkCopy	Adds menu option to copy multiple files to repository.
7.	CLIF.BulkMove	Adds menu option to move files in the site action menu.
8.	CLIF.ConfigureDocumentLibrary	Configures MySite document library.
9.	CLIF.ContentTypes	Defines all content types for the root level site.
10.	CLIF.Fields	Defines all SharePoint site columns in the root site.
11.	CLIF.Images	Adds images to “Site Collection Images library” in the root Site.
12.	CLIF.ImportFromRepository	Adds a new ECB menu to all document libraries that allows users to import files from the repository.
13.	CLIF.Lists	Defines templates for all SharePoint lists as well as creates instance of list in the root SharePoint site.
14.	CLIF.LookUpWithPicker	Handles the loop up with picker controls.
15.	CLIF.MasterPages	Defines SharePoint master pages used as a basis for the site collection.

16.	CLIF.MySite	Defines a template for "MySites" created for CLIF users which determine the look and feel of the site.
17.	CLIF.MySiteContentTypes	This feature defines all content types for MySite"".
18.	CLIF.MySiteFields	This feature defines all SharePoint site columns in the "MySite".
19.	CLIF.MySiteLists	This feature defines templates for all SharePoint lists as well as creates instance of list in "MySite".
20.	CLIF.MySiteLookupList	Adds the MySite Lookup list.
21.	CLIF.MySitePageLayouts	Activating this feature adds MySite page layouts.
22.	CLIF.MySiteStapler	Activates "CLIF.MySite" feature every time a new "MySite" is created for CLIF user. It also adds required sharepoint list required to achieve required functionalities.
23.	CLIF.PageLayouts	Activating this feature adds page layouts to the root site.
24.	CLIF.ProjectDocumentsItem EventsReceiver	Events receiver for project documents.
25.	CLIF.PublishDocument	This feature adds an ECB menu to all document list libraries.
26.	CLIF.RetrieveFromArchive	Activating this feature adds ECBMenu to retrieve files from the repository.
27.	CLIF.Scripts	Adds ".JS" files to "Styles library" in the root Site.
28.	CLIF.Search	Replaces default search controls with a custom control.
29.	CLIF.ShareDocuments	This feature adds an ECB menu to all document list libraries.
30.	CLIF.ShareThisDocument	Adds "Share this document" menu item.
31.	CLIF.Styles	This feature adds ".CSS" files to "Styles library" in the root Site.
32.	CLIF.Themes	Activates the CLIF themes.
33.	CLIF.WebConfigEntries	This feature adds all the required tags for "CLIF" in the "web.config" in the SharePoint application.
34.	CLIF.XSL	Activating this feature copies the required XSL files.

3.4.9 Metadata entry

Copy To Repository (Advanced) invokes a page for creating Hydra compliant metadata for a file. Note that unchecking all metadata options results in only a default DC metadata stream being embedded in the fedora object.

The basic design philosophy behind the metadata entry page is to load User Controls (or infopath forms) dynamically onto the page depending on the metadata option checkboxes the user selects. Each User Control builds a complete metadata form and JQuery helper javascript libraries build the complete XML for it. User Controls can utilise the full range of avaiable Ajax controls (e.g DatePickers / ComboBoxes). Prior to page postback some JQuery local field validation is performed for certain fields (e.g name fields). Initial postback is done asynchronously (via Ajax/ASP.NET client script postback) and validates all metadata fields against any relevant XSD. Validation errors result in the relevant form fields being highlighted (Infopath forms contain their own Office Server routines for highlighting errors).

If all forms validate without error, a second full page postback occurs and the DepositDocument method gets invoked to send the document to the Fedora repository along with the metadata streams.

A full MODS 3.3 Infopath 2010 form has been created as an additional output of the project.

3.5 Sakai-Fedora Integration

3.5.1 Integration architecture

At the start of the project the CLIF team was aware of work undertaken by the Cambridge Tetra Repositories Enhancement Project (CTREP) and, rather than re-invent the wheel, elected to use the output of this project as a starting point for CLIF's Fedora-Sakai integration. Specifically, we elected to investigate the Fedora component developed at the University of the Highlands and Islands (UHI).

The CTREP UHI Fedora content hosting handler is an extension to the Sakai resources tool that provides a snapshot tree view of the hierarchical resources within a Fedora repository. This is achieved through the Content Hosting Handler (CHH) layer extension mechanism[10] (a provider bean class *org.sakaiproject.content.chh.fedora.ContentHostingHandlerImplFedora* is injected into the Sakai AXIS framework and a mount point file uploaded via the resources tool).

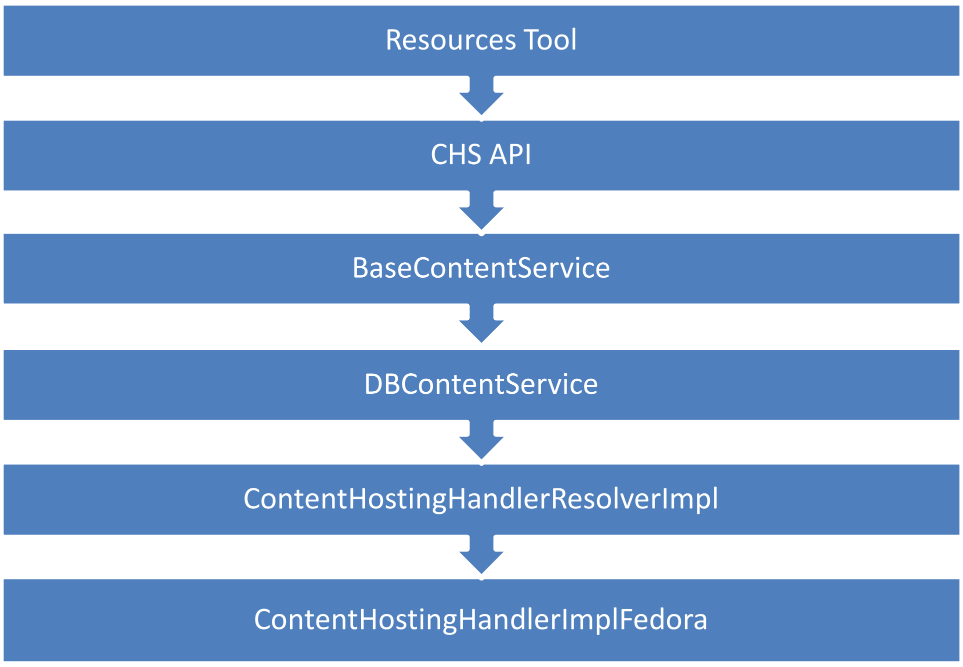
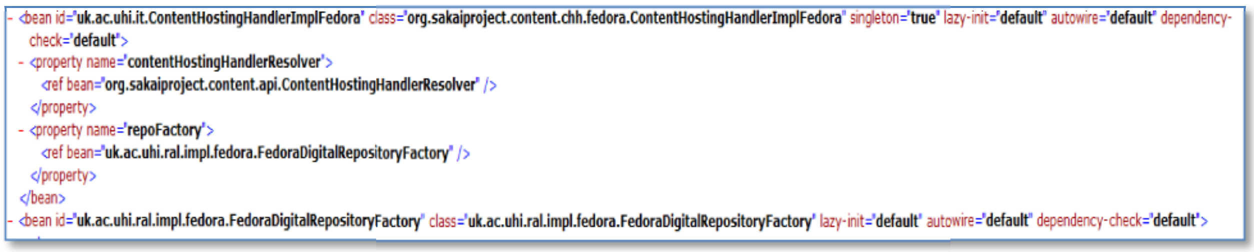

Resources Tool
CHS API
BaseContentService
DBContentService
ContentHostingHandlerResolverImpl
ContentHostingHandlerImplFedora

Figure 19: Diagram of Sakai Content Hosting Handler model

[10] This was developed by the CARET project

The structure of the Content Hosting Service (CHS) is basically composed ofa *BaseContentService* class that implements the API, and then an extension class that specialises *BaseContentService for different types of content store e.g. database , filestore, fedora.* The term 'virtual' resource is seen throughout the code to refer to resources which appear below a certain database folder point (termed the 'membrane between the real and virtual worlds' in one code comment). Even if the membrane point is known to be reached, it would appear that an attempt is still made to read a resource with the given identifier from the database, and on failing the request, control is passed onto the CHH Resolver instance (which delegates to the real *ContentHostingHandlerImplFedora* bean).



```
- <bean id="uk.ac.uhi.it.ContentHostingHandlerImplFedora" class="org.sakaiproject.content.chh.fedora.ContentHostingHandlerImplFedora" singleton="true" lazy-init="default" autowire="default" dependency-
    check="default">
  - <property name="contentHostingHandlerResolver">
      <ref bean="org.sakaiproject.content.api.ContentHostingHandlerResolver" />
    </property>
  - <property name="repoFactory">
      <ref bean="uk.ac.uhi.ral.impl.fedora.FedoraDigitalRepositoryFactory" />
    </property>
  </bean>
- <bean id="uk.ac.uhi.ral.impl.fedora.FedoraDigitalRepositoryFactory" class="uk.ac.uhi.ral.impl.fedora.FedoraDigitalRepositoryFactory" lazy-init="default" autowire="default" dependency-check="default">
```

Figure 20: Spring configuration file for CTREP CHH Fedora

The diagram above shows how the various classes are hooked together at runtime by the Spring IOC framework hosted, in our case, by Tomcat 5.5.28.

Fedora SOAP messaging (HTTP / HTTPS Guanxi)

The CHH Fedora handler communicates with the Fedora system using two SOAP web service APIs already within Fedora; API-A and API-M. To communicate with these web service APIs the original CHH Fedora handler source code makes sole use of a Guanxi HTTPS transport layer (developed at UHI). This requires Java keystore (JKS certificate) security setting up on both client and server and makes it difficult to debug in a development environment (all SOAP messaging traffic is HTTPS encrypted). After some initial problems setting up the system, it was found to work acceptably (albeit very slowly). It was decided to supplement this layer with the more basic HTTP protocol, not only in order to try and boost performance, but also to facilitate easier debugging. In any case, there is questionable benefit in securing the connection between Fedora and Sakai as both servers are likely to be on the same security partitioned part of a campus network and HTTPS requires more server processing cycles than HTTP. In the end, code modifications have been put in place which allow Guanxi HTTPS or HTTP SOAP transport by setting a configuration parameter in the mountpoint.properties file.

Code design

The design of the existing code was found to be very complex with hardly any comments present. Searching the internet came up with only the barest outline block diagram to serve as documentation. Thus a good third of the project time working on Sakai was spent trying to make the code work and then to understand it sufficiently that we could modify it for CLIF with some level of confidence. At the time of its development, limitations of the Sakai code prevented the UHI team from completing and testing a fully functional version of their project; accordingly we also found a range of bugs that needed fixing. We are grateful to staff at UHI who were involved with the original CTREP work for their assistance in resolving a number of these issues.

On a number of occasions the CLIF team did step back and consider whether continuing to work with the CTREP code base was an appropriate strategy or whether it might not be better to start again from scratch. On each of these occasions we decided that the project would best be served by continuing as we were. Reviewing these decisions in hindsight, we feel justified in our approach as the resulting repository resource tool integrates nicely into the core functionality of Sakai (for example, copy and paste with other site resource tools works seamlessly).

As part of our work trying to understand the CTREP code a very much cut down UML diagram class diagram was produced (see below) which shows at least some of the relationships between the main CHH Java classes (to avoid over-complexity, many relationships are not shown).

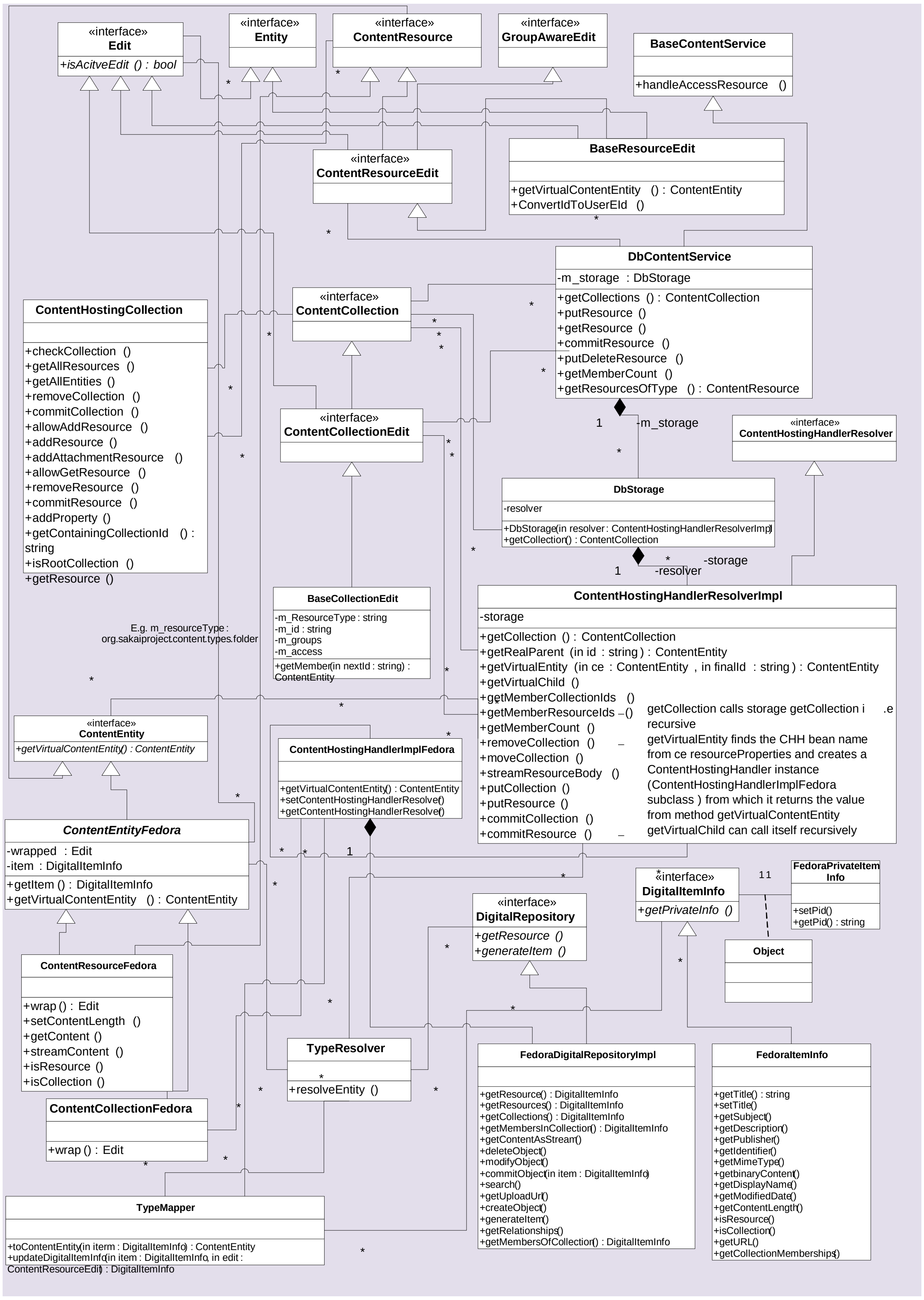

«interface» Edit
+isAcitveEdit () : bool
«interface» Entity
«interface» ContentResource
«interface» GroupAwareEdit
BaseContentService
+handleAccessResource ()
BaseResourceEdit
+getVirtualContentEntity () : ContentEntity
+ConvertIdToUserEId ()
«interface» ContentResourceEdit
DbContentService
-m_storage : DbStorage
+getCollections () : ContentCollection
+putResource ()
+getResource ()
+commitResource ()
+putDeleteResource ()
+getMemberCount ()
+getResourcesOfType () : ContentResource
ContentHostingCollection
+checkCollection ()
+getAllResources ()
+getAllEntities ()
+removeCollection ()
+commitCollection ()
+allowAddResource ()
+addResource ()
+addAttachmentResource ()
+allowGetResource ()
+removeResource ()
+commitResource ()
+addProperty ()
+getContainingCollectionId () : string
+isRootCollection ()
+getResource ()
«interface» ContentCollection
«interface» ContentCollectionEdit
1
-m_storage
«interface» ContentHostingHandlerResolver
DbStorage
-resolver
+DbStorage(in resolver : ContentHostingHandlerResolverImpl)
+getCollection() : ContentCollection
1
-storage
-resolver
BaseCollectionEdit
-m_ResourceType : string
-m_id : string
-m_groups
-m_access
+getMember(in nextId : string) : ContentEntity
E.g. m_resourceType : org.sakaiproject.content.types.folder
ContentHostingHandlerResolverImpl
-storage
+getCollection () : ContentCollection
+getRealParent (in id : string) : ContentEntity
+getVirtualEntity (in ce : ContentEntity , in finalId : string) : ContentEntity
+getVirtualChild ()
+getMemberCollectionIds ()
+getMemberResourceIds ()
+getMemberCount ()
+removeCollection ()
+moveCollection ()
+streamResourceBody ()
+putCollection ()
+putResource ()
+commitCollection ()
+commitResource ()
getCollection calls storage getCollection i .e recursive
getVirtualEntity finds the CHH bean name from ce resourceProperties and creates a ContentHostingHandler instance (ContentHostingHandlerImplFedora subclass) from which it returns the value from method getVirtualContentEntity
getVirtualChild can call itself recursively
«interface» ContentEntity
+getVirtualContentEntity() : ContentEntity
ContentHostingHandlerImplFedora
+getVirtualContentEntity() : ContentEntity
+setContentHostingHandlerResolver()
+getContentHostingHandlerResolver()
1
ContentEntityFedora
-wrapped : Edit
-item : DigitalItemInfo
+getItem () : DigitalItemInfo
+getVirtualContentEntity () : ContentEntity
«interface» DigitalItemInfo
+getPrivateInfo ()
1 1
FedoraPrivateItemInfo
+setPid()
+getPid() : string
Object
«interface» DigitalRepository
+getResource ()
+generateItem ()
ContentResourceFedora
+wrap () : Edit
+setContentLength ()
+getContent ()
+streamContent ()
+isResource ()
+isCollection ()
TypeResolver
+resolveEntity ()
FedoraDigitalRepositoryImpl
+getResource() : DigitalItemInfo
+getResources() : DigitalItemInfo
+getCollections() : DigitalItemInfo
+getMembersInCollection() : DigitalItemInfo
+getContentAsStream()
+deleteObject()
+modifyObject()
+commitObject(in item : DigitalItemInfo)
+search()
+getUploadUrl()
+createObject()
+generateItem()
+getRelationships()
+getMembersOfCollection() : DigitalItemInfo
FedoraItemInfo
+getTitle() : string
+setTitle()
+getSubject()
+getDescription()
+getPublisher()
+getIdentifier()
+getMimeType()
+getbinaryContent()
+getDisplayName()
+getModifiedDate()
+getContentLength()
+isResource()
+isCollection()
+getURL()
+getCollectionMemberships()
ContentCollectionFedora
+wrap () : Edit
TypeMapper
+toContentEntity(in iterm : DigitalItemInfo) : ContentEntity
+updateDigitalItemInfo(in item : DigitalItemInfo, in edit : ContentResourceEdit) : DigitalItemInfo

Figure 21: UML class diagram for CTREP UHI Fedora Content Hosting Handler Implementation

Examples of existing code complication

As can be seen from the diagram there is a complicated hierarchy of interface, abstract and derived classes involved. It is not always apparent that the best casted type for an object is passed as a reference down the call stack. One example of this was the case of a *BaseResourceEdit* argument passed as *ContentEntity* to a lower call stack method which then, just occasionally, ended up being cast to *Edit.* For example in the wrap method of *ContentCollectionFedora,* the *RealParent* property (type *ContentEntity*) is cast to *Edit*, causing - in the case of nested sub folders - a difficult exception to track down and fix. Some higher level defensive code in DbContentService.java, to prevent too much stack recursion, also got in the way of sub-folder drill down working correctly. Removing the *in*() / *out*() stack recursive protection code was not an option as this caused an exception from another Sakai tool on Sakai start-up; also it would be difficult to predict what other ramifications might occur if this approach was adopted. Therefore an overloaded method *putCollectionAllowRecursive* was introduced to circumvent this defensive code in the sole case of the Fedora Content Hosting Handler being used (see *addValidPermittedCollection* method modification in *BaseContentService*).

Observation on BaseCollectionEdit.getMemberResources method

As well as the issues foundwith the CTREP code as discussed already, problems were also identified with the Sakai Content Hosting Handler interface itself. For instance:

The following two lines of high level system code (in the BaseCollectionEdit class getMemberResources method) have major ramifications in the ContentHostingHandler implementation classes lower down:

```
mbrs.addAll(m_storage.getCollections(this));
mbrs.addAll(m_storage.getResources(this));
```

In particular, and code comments suggest developers were not aware of this, the nature of getResources() has to change in all lower level classes: for example, some methods expect to return a collection that contains references to both contentCollection and contentResource objects.

If, as we intend, the CLIF work is to serve as the foundation for a production system at Hull it may be that we should consider some significant further code-refactoring in order to rectify this situation (possibly it would be better still to significantly re-write the CHH interface).

3.5.2 Added functionality

The CTREP work did not offer all the functionality that CLIF envisaged and thus the following additional features, not present in the original UHI code, have now been implemented within our version of the Fedora resources tool :

Creation of Folders: The original UHI code displayed the contents of one root level Fedora collection only; CLIF now implements creation of new Fedora collections (folders in the Sakai display) allowing uploading and display of resources in those folders.

Copy , paste and move of folders or resources is now fully supported: Folders with nested sub-folders and content can be copied or moved within the Sakai mounted folder or copied and moved to another Sakai resources area (if the 'show other sites' configuration option is selected).

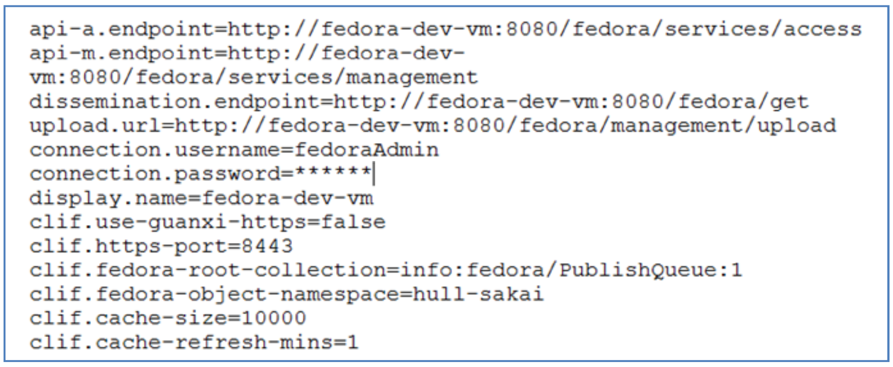
Nested folder deletion: This is now fully supported.

Correct drill down display of sub folders: (to any depth): Selecting a sub-folder in either the resource tool tree view or the breadcrumb trail now causes a new tree view to be displayed reflecting the resources contained within that sub-folder.

MODS metadata: Basic MODs metadata is now automatically added to Fedora objects ingested from Sakai as the 'descMetadata' datastream (as recommended by Hydra).

HTTP or HTTPS Guanxi: It is now possible to change the SOAP transport layer via an additional configuration parameter 'clif.use-guanxi-https'.

Mountpoint file supports additional configuration parameters: The screenshot below shows the additional configuration parameters supported in the Sakai mountpoint file.



```
api-a.endpoint=http://fedora-dev-vm:8080/fedora/services/access
api-m.endpoint=http://fedora-dev-
vm:8080/fedora/services/management
dissemination.endpoint=http://fedora-dev-vm:8080/fedora/get
upload.url=http://fedora-dev-vm:8080/fedora/management/upload
connection.username=fedoraAdmin
connection.password=******
display.name=fedora-dev-vm
clif.use-guanxi-https=false
clif.https-port=8443
clif.fedora-root-collection=info:fedora/PublishQueue:1
clif.fedora-object-namespace=hull-sakai
clif.cache-size=10000
clif.cache-refresh-mins=1
```

Figure 22: Example mountpoint.properties file with the CLIF additions

3.5.3 Performance Improvements

Quite early on in our work with the CTREP code, it was apparent that the speed of rendering resources from the Fedora repository within the Sakai resources tool was very poor (over a minute to render a dozen or so resources within the Fedora repository). The following code re-working has been done to alleviate this (mainly in and around *FedoraDigitalRepositoryImpl.java*):

1. Refactor direct calls to web service method as persisted properties (populate multiple file object properties from one web service method call)
2. No pre-fetch of content datastream for each resource.
3. Caching of resources
4. Caching of collection/folder contents from RDF query results

Refactor direct calls to web service method as persisted properties (populate multiple properties from one web service method call)

Methods *isInCollection*() and *isCollection*() both made two direct SOAP web-service API calls (approximately 3 seconds duration each) and these were used *ad hoc* throughout the code wherever these 'properties' of a particular resource were required. The code was re-factored to add two new methods on class *DigitalItemInfo*, namely *isCollection*() and *isInCollection*() and to populate these on the initial fetch of the resource. A resource cache would then allow the property to be read directly from a cached memory instance of the resource rather than a repeated SOAP request from Fedora.

No pre-fetch of content datastream

The content datastream (as well as the metadata datastreams) of each resource returned from the Fedora resources query would be fetched from Fedora in order to populate the *binaryContent* and *Size* properties on each Sakai *DigitalItemInfo* object. This is very wasteful of server memory (especially since only a few of the potentially large number of resources are ever likely to be opened to view the actual content). So after some thought, it was decided to adopt a 'just-in-time' read methodology. Instead of pre-fetching the content datastreams, the *binaryContent* buffer would be left empty until actually required. The *contentLength* property was obtained via an alternative API method which only returned basic object metadata. However a bug in Fedora meant that this size (when populated) was always returned as zero for Fedora ‘managed’ datastreams (listed in the Fedora JIRA instance as issue FCREPO-64). This was resolved mid-way through the project with the Fedora 3.4 release. However, a fix was made to the Sakai kernel whereby should the CHH-handler return zero length the content is streamed initially to determine its size (which ends up in the HTTP Content-Length header) before streaming again to the user (streaming twice means that a small buffer size can be used saving server memory). In the context of Sakai, this may be a better long-term solution to the problem anyway, because Fedora does not provide for returning the filesize of ‘external’ content which is the way that many Fedora repositories store content.[11]

[11] The Fedora repository software allows a number of ways to store digital content. Amongst these ‘managed’ content, as the name suggests, is managed entirely by the Fedora software itself but this requires that the content store be, or be represented as, a disk system

Caching of resources

Caching of resources is now done at the individual resource and collection levels:

- **Individual resource cache:** Because web-service requests by their nature are relatively time consuming, typically in the order of seconds per call, a software cache was implemented to store resources fetched during their initial retrieval. Subsequent collection revisits (for instance, a refresh of a folder in the Resources tool page) reads resources from the cache rather than from Fedora directly. This then makes speed of rendering a potential issue only for the initial fetching of resources. A mountpoint configuration parameter determines the time interval before the cache is deemed to be out of date, requiring an actual re-reading of data from Fedora. Careful consideration should be given by the Sakai administrator to ensure the most appropriate value is chosen for this; the more resources a collection contains the less frequently the cache should be refreshed and vice-versa.
- **Collection level ITQL RDF query results cache:** An ITQL RDF query against the Fedora Resource Index is used to discover all the Fedora objects that form part of a collection (displayed as a Sakai resource folder). The results obtained from the query are cached in a similar manner to that implemented for individual resources.

  At present, the mountpoint configuration properties (size and refresh interval) apply to both types of cache.

Even with a substantial Fedora content collection structure in place it is hoped that, with these caching techniques in place, users will find the initial delay to render the resources acceptable.

Kernel alterations

A basic philosophy was to minimize alterations to the Sakai kernel to ensure existing Sakai tools using the code would be unaffected. Because of the move to a 'just-in-time' read design a change was made to the org.sakaiproject.content.impl.BaseContentService handleAccessResource method to handle resources with a reported 0 size (see previous explanation on this). This modification causes a default buffer size to be allocated, and the content datastream to be read an additional time in order to determine the size of the content (required by the HttpServletResponse.setContentLength method).

3.5.4 Future work needed

Additional functionality

There is always a point when one has to decide what can be achieved in the allotted project time-frame and what must be pushed to a list of desirable, additional functionality. If, as we intend, the

local to the server. Fedora retrieves 'external' content from an HTTP(S)-accessible source on request: the Fedora datastream contains only a URL and other systems outside Fedora must be used to create and manage the content in this external store.

CLIF work will be the basis for a production system at the University of Hull there are a number of items to consider:

- There is no way of telling presently whether a new version of a resource has been uploaded in Sakai. There is also no way of viewing past versions or reverting to past versions of a resource in Sakai.
- Many of the options in the Edit resource details screen (e.g. Availability and access, Change file type, Copyright status) have no effect on the underlying Fedora object.
- Only a limited number of Sakai resource properties (via resource 'edit details') are presently conveyed in Fedora object metadata (e.g. title, description, subject) when ingesting to Fedora. On import from Fedora even more metadata' goes missing' (e.g. description, subject and publisher are not displayed in the edit details screen for a newly loaded Fedora object); this looks to be the fault of some other part of the Sakai system and not to do with the Content Hosting Handler layer.
- If the collection being browsed holds a significant number of resources, the resource tool tree structure browse is potentially slow. The ability to search the Fedora collection for a particular object or add a 'filter' to the browse function would therefore be a very desirable additional capability. Another approach to improving the speed of rendering would be to factor out all calls for additional fedora object information (not possible to obtain from the ITQL RDF) from FedoraDigitalRepositoryImpl.java (specifically the method gatherMoreResourceData), refactor those references as 'lazy' object property requests on a specific FedoraItemInfo object and re-work the Resource Tool UI to load these extra properties (e.g. contentLength and mimeType) using an asynchronous AJAX request as the Resources Tool page renders.
- Upload of zero byte files results in them showing as length -1. (The relevant code is deep inside the Fedora.FedoraSystemDef SOAP proxy so may not be straightforward to fix.)

Additional security considerations

The CLIF code as implemented assumes that Sakai has unrestricted read and write access to a Fedora repository, in other words it assumes that the repository is not secured (thus it does not currently meet requirement 1b). This will certainly not be the case if the project's work is deployed at Hull where the institutional repository has a range of security restrictions. Users browsing the repository from Sakai should only be allowed to view areas of the repository appropriate to their status within the institution (undergraduate, member of staff etc) and most should not be allowed to deposit content without it first being subject to quality assurance.

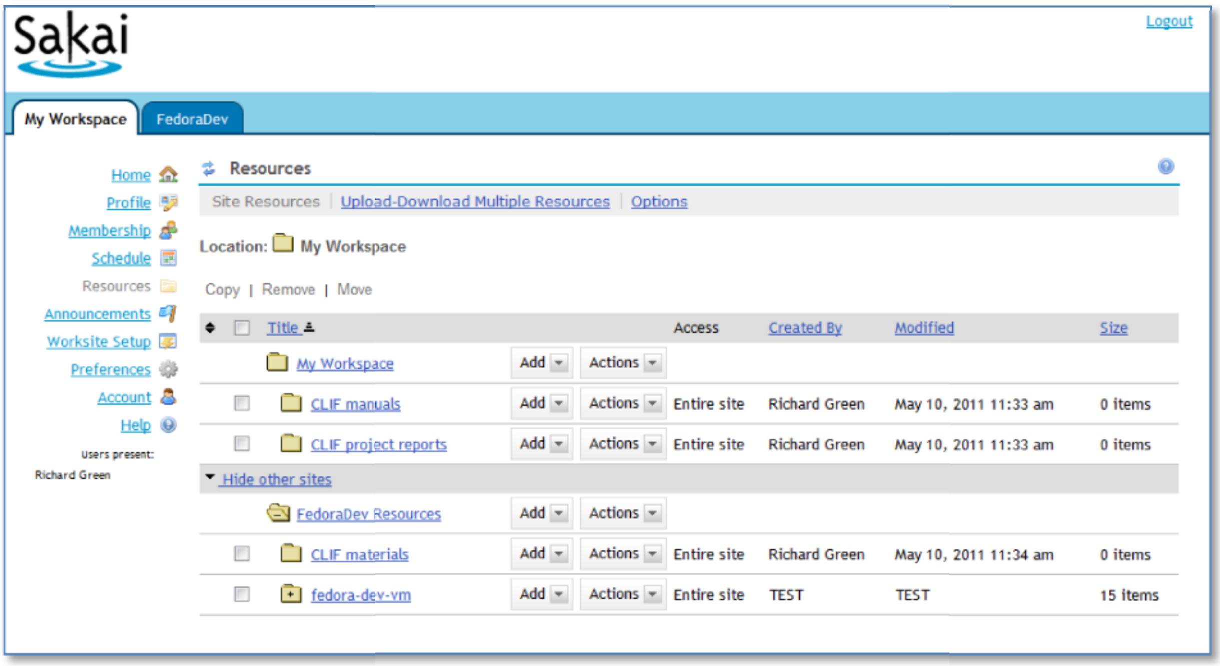
Post-project work is anticipated at Hull to link CLIF's Fedora resources tool to a particular node in the repository structure below which its users will be able to read. Write access will be limited to a particular node within that readable structure, for most people this will take the form of a submissions box whose content is then subject to quality assurance before being moved by repository management staff to an appropriate point in the repository structure. The *mountpoint.properties* file illustrated in section 3.4.2 has a setting *clif.fedora-root-collection* which allows the read-node to be defined.

Now that improved groundwork has been laid, we anticipate that one outcome of this project will be that it will be very much easier for other developers, at Hull or elsewhere, to carry on development of the Fedora CHH handler and implement additional features at some future date.

3.6 CLIF-Sakai functionality

3.6.1 Resources page

Sakai can be configured so that when a user logs in they see multiple site tabs at the top of their workspace. If a user goes to their own workspace resources area and chooses to 'Show other sites' they will be presented with two panes, one representing their local Sakai resources and the other the virtual resources in the Fedora repository.



Location: My Workspace

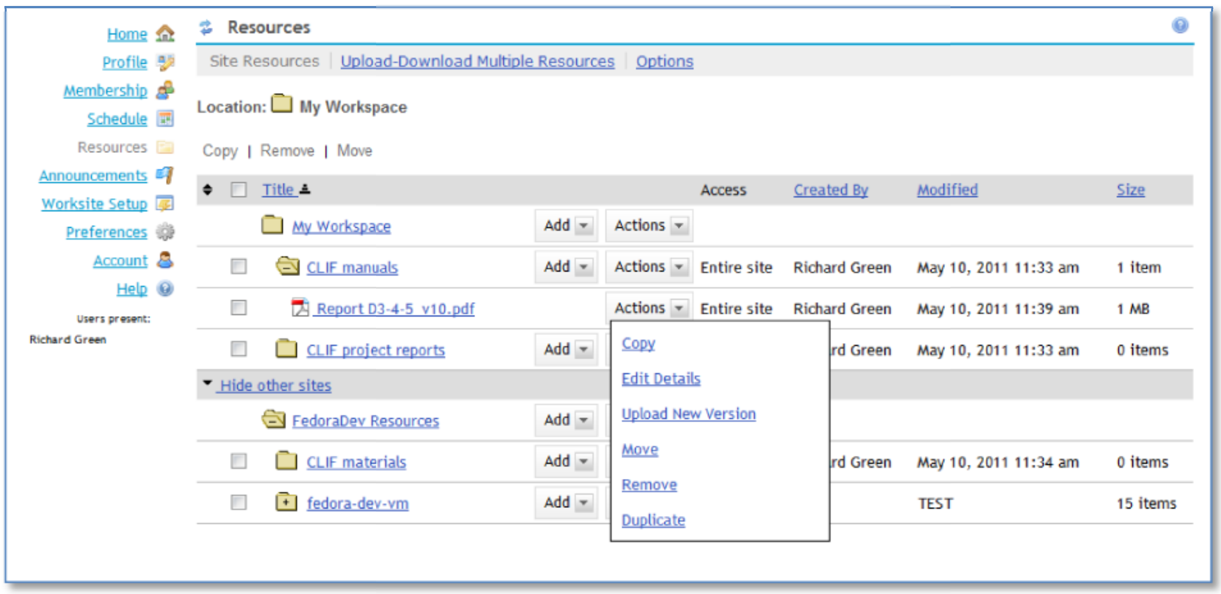
Copy | Remove | Move

Title			Access	Created By	Modified	Size
My Workspace	Add	Actions				
CLIF manuals	Add	Actions	Entire site	Richard Green	May 10, 2011 11:33 am	0 items
CLIF project reports	Add	Actions	Entire site	Richard Green	May 10, 2011 11:33 am	0 items
Hide other sites						
FedoraDev Resources	Add	Actions				
CLIF materials	Add	Actions	Entire site	Richard Green	May 10, 2011 11:34 am	0 items
fedora-dev-vm	Add	Actions	Entire site	TEST	TEST	15 items

Figure 23: Sakai screen showing both local resource folders and folders of virtual materials on Fedora

3.6.2 Copy/move to repository

The 'CLIF manuals' folder in the local resources shown above contains a file. This can be copied and pasted to the repository using the normal Sakai approach. The copy option for the file is used:

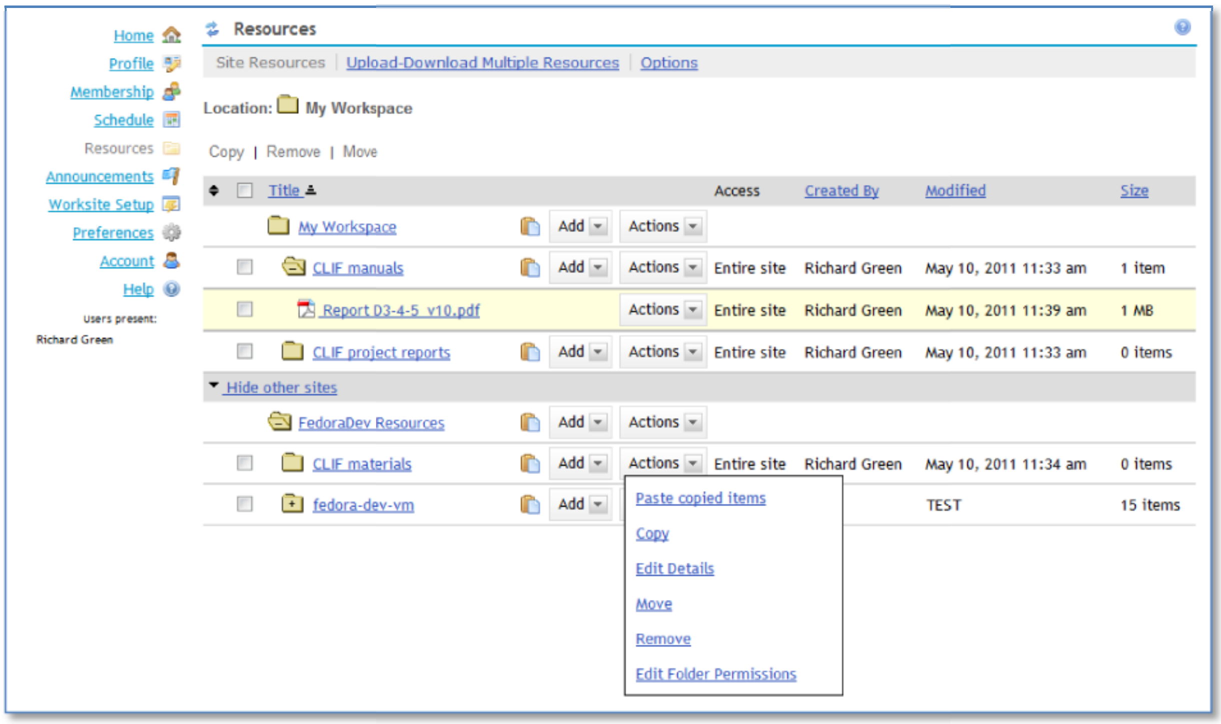


Location: My Workspace

Title			Access	Created By	Modified	Size
My Workspace	Add	Actions				
CLIF manuals	Add	Actions	Entire site	Richard Green	May 10, 2011 11:33 am	1 item
Report D3-4-5_v10.pdf		Actions	Entire site	Richard Green	May 10, 2011 11:39 am	1 MB
CLIF project reports	Add			rd Green	May 10, 2011 11:33 am	0 items
Hide other sites						
FedoraDev Resources	Add					
CLIF materials	Add			rd Green	May 10, 2011 11:34 am	0 items
fedora-dev-vm	Add				TEST	15 items

Figure 24: Sakai resources screen: File selected for copying

The user then navigates to the appropriate point in their virtual Fedora tree and uses the 'paste copied item' option:

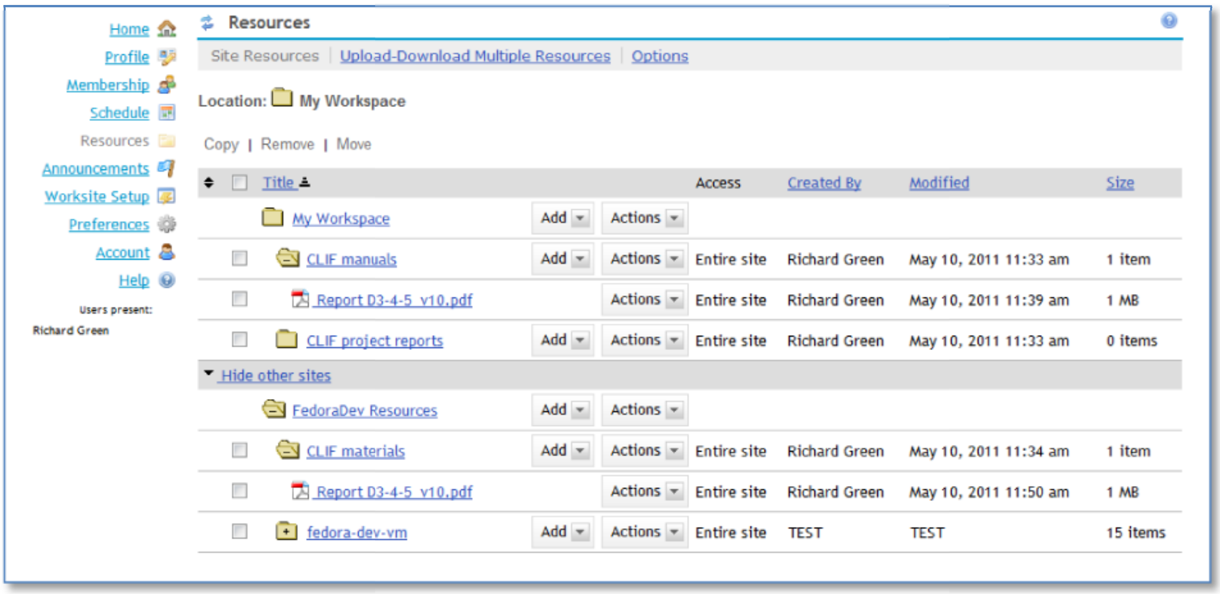


Location: My Workspace

Title			Access	Created By	Modified	Size
My Workspace	Add	Actions				
CLIF manuals	Add	Actions	Entire site	Richard Green	May 10, 2011 11:33 am	1 item
Report D3-4-5_v10.pdf		Actions	Entire site	Richard Green	May 10, 2011 11:39 am	1 MB
CLIF project reports	Add	Actions	Entire site	Richard Green	May 10, 2011 11:33 am	0 items
Hide other sites						
FedoraDev Resources	Add	Actions				
CLIF materials	Add	Actions	Entire site	Richard Green	May 10, 2011 11:34 am	0 items
fedora-dev-vm	Add				TEST	15 items

Figure 25: Sakai resources screen: Folder selected for pasting

…with the result that the item is copied as requested. The 'Edit details' option can be used to configure minimal metadata in the Fedora object.


Home
Profile
Membership
Schedule
Resources
Announcements
Worksite Setup
Preferences
Account
Help
Users present:
Richard Green

Resources

Site Resources | Upload-Download Multiple Resources | Options

Location: My Workspace

Copy | Remove | Move

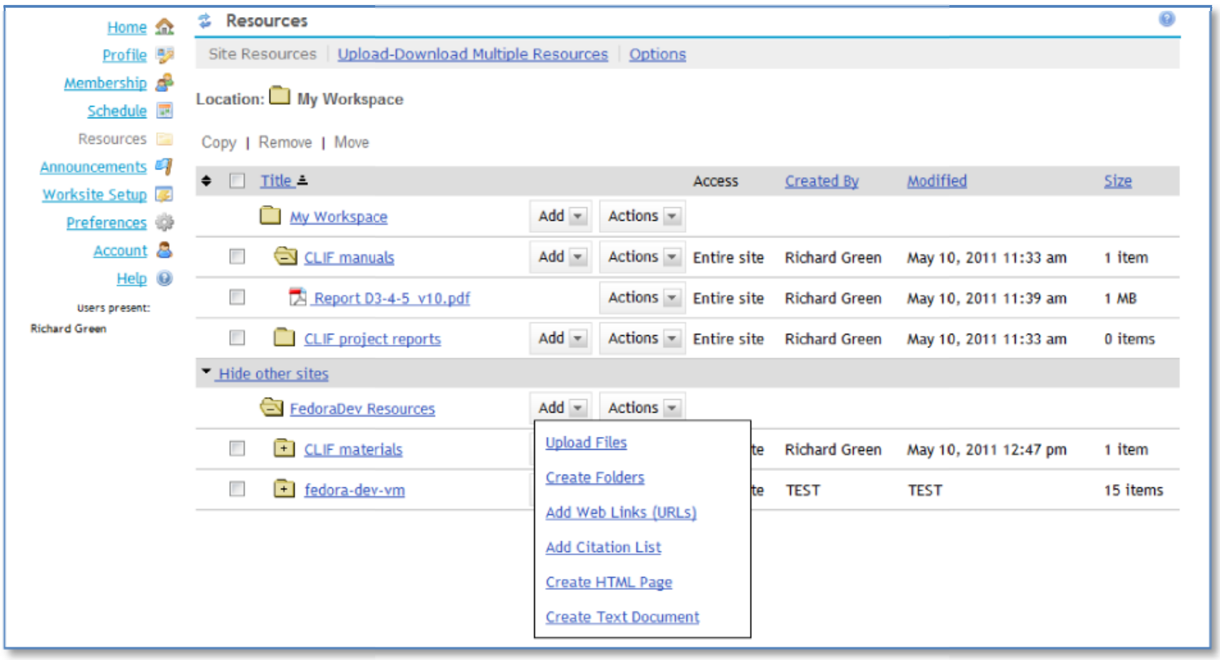
Title			Access	Created By	Modified	Size
My Workspace	Add	Actions				
CLIF manuals	Add	Actions	Entire site	Richard Green	May 10, 2011 11:33 am	1 item
Report D3-4-5_v10.pdf		Actions	Entire site	Richard Green	May 10, 2011 11:39 am	1 MB
CLIF project reports	Add	Actions	Entire site	Richard Green	May 10, 2011 11:33 am	0 items
Hide other sites						
FedoraDev Resources	Add	Actions				
CLIF materials	Add	Actions	Entire site	Richard Green	May 10, 2011 11:34 am	1 item
Report D3-4-5_v10.pdf		Actions	Entire site	Richard Green	May 10, 2011 11:50 am	1 MB
fedora-dev-vm	Add	Actions	Entire site	TEST	TEST	15 items

Figure 26: Sakai resources screen: File successfully copied

Move works in very similar manner. It is possible to copy or move entire structures, not just single files. The two 'Edit' options and the 'Remove' option function as expected.

3.6.3 Remote folder creation/removal

Folders in the repository are manipulated from the 'Add' menu at the appropriate node.


Home
Profile
Membership
Schedule
Resources
Announcements
Worksite Setup
Preferences
Account
Help
Users present:
Richard Green

Resources

Site Resources | Upload-Download Multiple Resources | Options

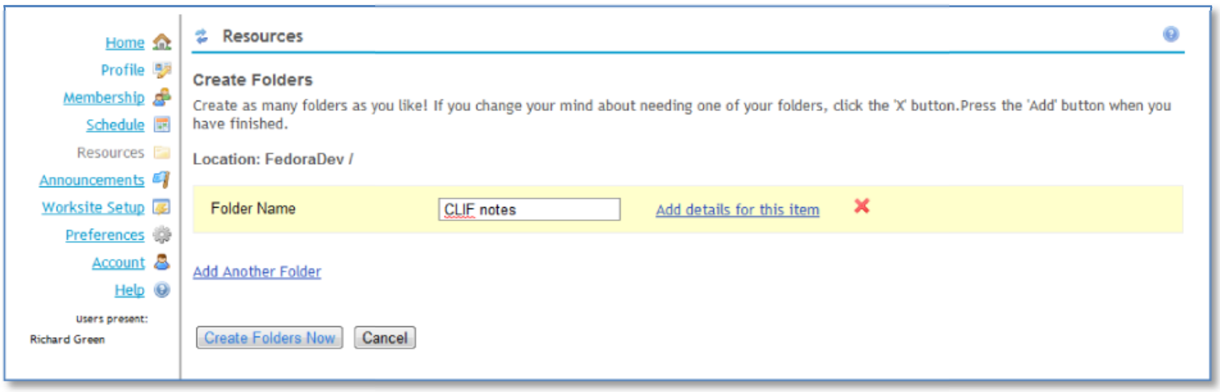
Location: My Workspace

Copy | Remove | Move

Title			Access	Created By	Modified	Size
My Workspace	Add	Actions				
CLIF manuals	Add	Actions	Entire site	Richard Green	May 10, 2011 11:33 am	1 item
Report D3-4-5_v10.pdf		Actions	Entire site	Richard Green	May 10, 2011 11:39 am	1 MB
CLIF project reports	Add	Actions	Entire site	Richard Green	May 10, 2011 11:33 am	0 items
Hide other sites						
FedoraDev Resources	Add	Actions				
CLIF materials			te	Richard Green	May 10, 2011 12:47 pm	1 item
fedora-dev-vm			te	TEST	TEST	15 items

Figure 27: Sakai 'create folder' option

Thus, clicking 'Add' at the FedoraDev Resources node and choosing to create a folder requests information…



Resources

Create Folders

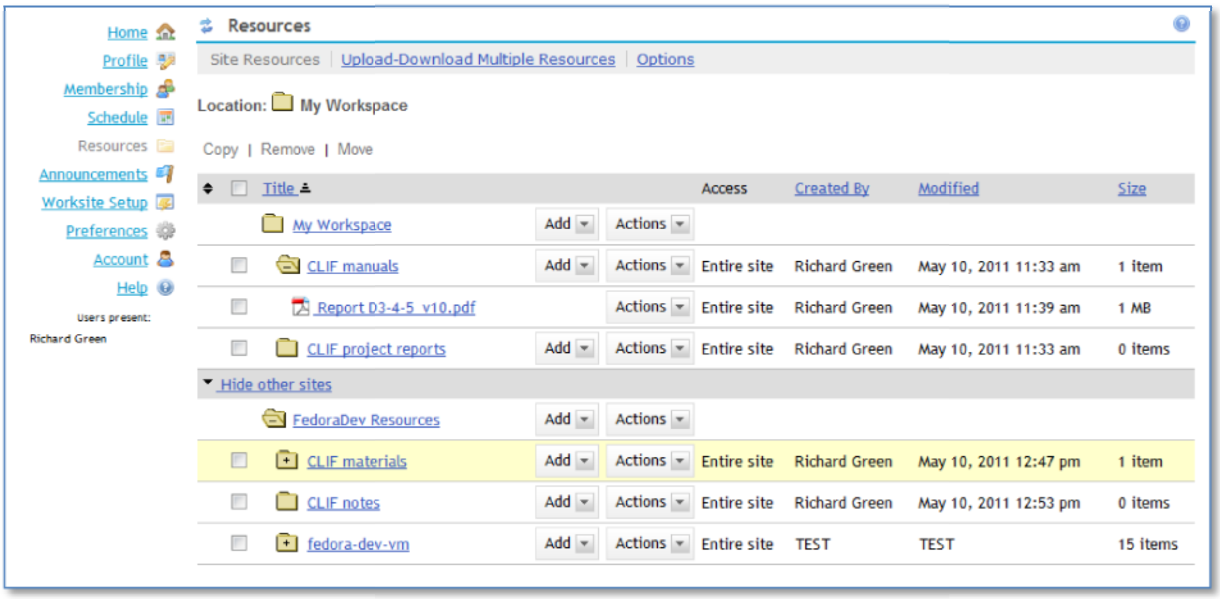
Create as many folders as you like! If you change your mind about needing one of your folders, click the 'X' button.Press the 'Add' button when you have finished.

Location: FedoraDev /

Folder Name: CLIF notes — Add details for this item

Figure 28: Sakai screen to create repository folder

… and then generates the appropriate object in Fedora under that node.

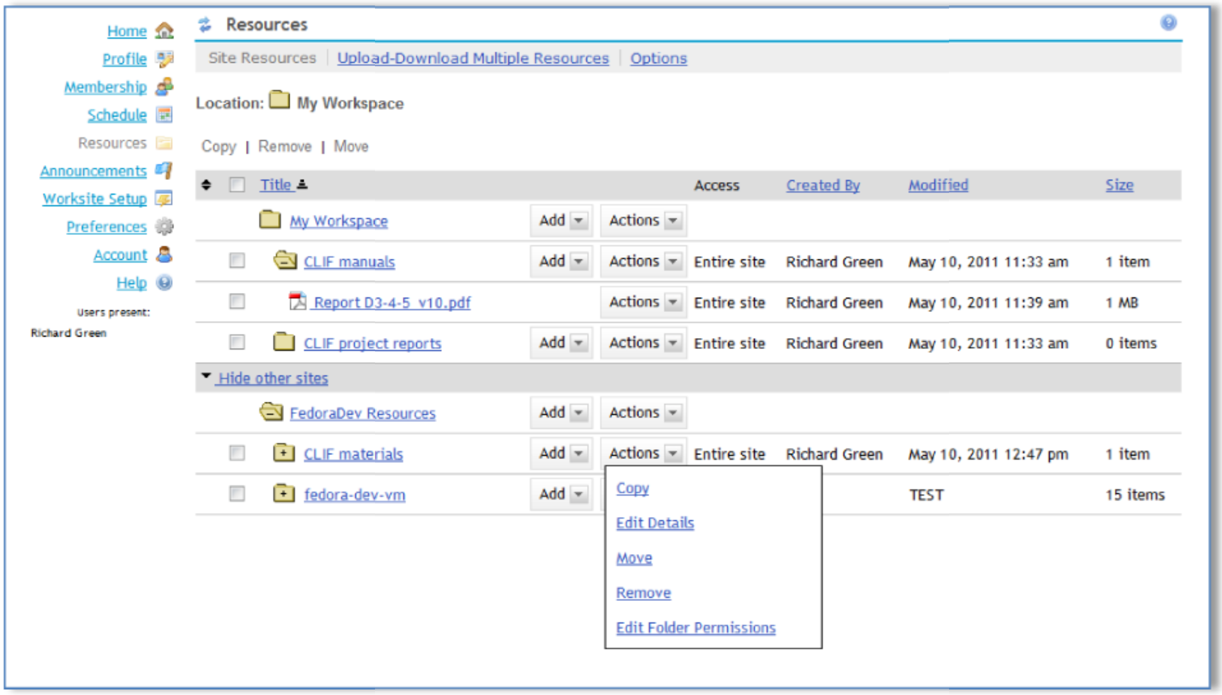


Location: My Workspace

Title			Access	Created By	Modified	Size
My Workspace	Add	Actions				
CLIF manuals	Add	Actions	Entire site	Richard Green	May 10, 2011 11:33 am	1 item
Report D3-4-5_v10.pdf		Actions	Entire site	Richard Green	May 10, 2011 11:39 am	1 MB
CLIF project reports	Add	Actions	Entire site	Richard Green	May 10, 2011 11:33 am	0 items
Hide other sites						
FedoraDev Resources	Add	Actions				
CLIF materials	Add	Actions	Entire site	Richard Green	May 10, 2011 12:47 pm	1 item
CLIF notes	Add	Actions	Entire site	Richard Green	May 10, 2011 12:53 pm	0 items
fedora-dev-vm	Add	Actions	Entire site	TEST	TEST	15 items

Figure 29: Sakai screen showing folder successfully created

A folder is removed via its ‘Actions’ menu:


Home
Profile
Membership
Schedule
Resources
Announcements
Worksite Setup
Preferences
Account
Help

Users present:
Richard Green

Resources

Site Resources | Upload-Download Multiple Resources | Options

Location: My Workspace

Copy | Remove | Move

Title			Access	Created By	Modified	Size
My Workspace	Add	Actions				
CLIF manuals	Add	Actions	Entire site	Richard Green	May 10, 2011 11:33 am	1 item
Report D3-4-5_v10.pdf		Actions	Entire site	Richard Green	May 10, 2011 11:39 am	1 MB
CLIF project reports	Add	Actions	Entire site	Richard Green	May 10, 2011 11:33 am	0 items
Hide other sites						
FedoraDev Resources	Add	Actions				
CLIF materials	Add	Actions	Entire site	Richard Green	May 10, 2011 12:47 pm	1 item
fedora-dev-vm	Add				TEST	15 items

Copy
Edit Details
Move
Remove
Edit Folder Permissions

Figure 30: Sakai folder 'Action' menu showing the 'remove' option

A warning is given if the folder is not empty. 'Remove', if invoked, removes the folder and all underlying contents.

4. Testing

4.1 General approach

We have divided testing into unit tests, integration tests and system tests. Since neither of the integration projects involved development of complex self-contained code, lengthy unit testing was not deemed necessary within the limited time span available.

4.2 SharePoint-Fedora integration

4.2.1 Unit tests

The code for creating SharePoint features is based on configuration files stored in the project CLIF.Solutions and the corresponding code behind files stored in the project CLIF.Code. Most of the individual classes are very simple and testing was required of the integration with SharePoint as well as thorough user testing.

The Hydranet code comprises fairly simple units for wrapping the Fedora API-M and API-A web service interfaces and did not require complex unit tests. The code had been in production use at Hull for some time before its use in the CLIF project.

4.2.2 Integration tests

Individual SharePoint features were deployed and tested independently. Since all the features implement UI components, the main integration tests could be carried out by validating responses against fixed user inputs.

4.2.3 System tests

System testing was performed deploying the full solution to SharePoint and validating each of the deployed features against a set of test inputs. Fedora content objects were inspected via the Fedora web administration interface in order to confirm that objects had been deposited and conformed to the required syntax.

4.3 Sakai-Fedora integration

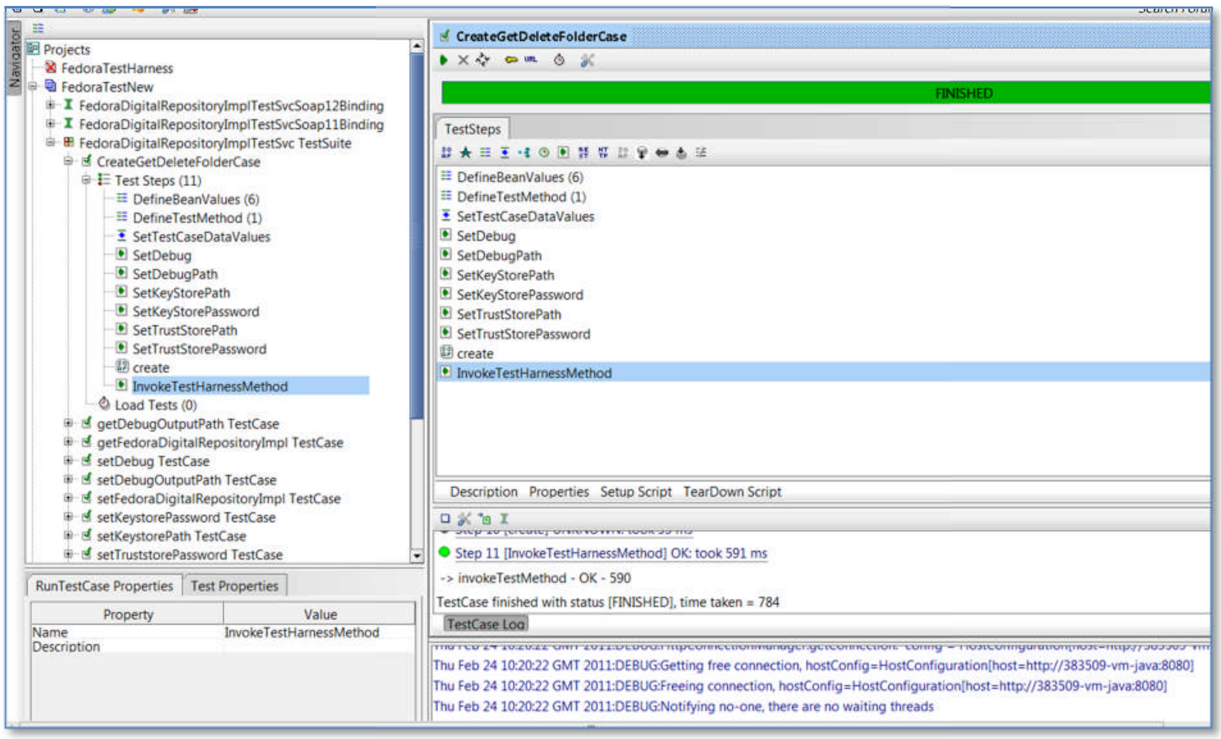
4.3.1 Unit tests

As the Sakai aspect of CLIF was mainly adaptation and re-working of existing third party code, and due to the code's rather complicated nature, it was not practical within the allotted project time frame to write any unit tests - for example, creating mock Fedora web-service API-M or API-A response classes would be fairly time consuming projects in their own right.

4.3.2 Integration tests

A novel way of testing the FedoraDigitalRepositoryImpl CHH handler was devised which allows it to be tested *in situ* within the Tomcat hosting environment. This approach makes use of Apache AXIS2's ability to expose a Spring bean class (albeit with some factory code plumbing) as a web service; Sakai conveniently makes use of AXIS to host some of its own web services. Thus the class can be tested and configured whilst running within its usual hosting environment; this approach also offers up other possibilities for extension UI's and administration tools in the future.

The freeware SOAPUI tool was used to develop some simple CRUD (create, read, update, delete) tests for folders and resources. SOAP messages are easily displayed in the SOAPUI tool, and any issues are indicated in the response message from the InvokeTestHarnessMethodStep (see Figure 31 below).



```
Projects
  FedoraTestHarness
  FedoraTestNew
    FedoraDigitalRepositoryImplTestSvcSoap12Binding
    FedoraDigitalRepositoryImplTestSvcSoap11Binding
    FedoraDigitalRepositoryImplTestSvc TestSuite
      CreateGetDeleteFolderCase
        Test Steps (11)
          DefineBeanValues (6)
          DefineTestMethod (1)
          SetTestCaseDataValues
          SetDebug
          SetDebugPath
          SetKeyStorePath
          SetKeyStorePassword
          SetTrustStorePath
          SetTrustStorePassword
          create
          InvokeTestHarnessMethod
        Load Tests (0)
      getDebugOutputPath TestCase
      getFedoraDigitalRepositoryImpl TestCase
      setDebug TestCase
      setDebugOutputPath TestCase
      setFedoraDigitalRepositoryImpl TestCase
      setKeystorePassword TestCase
      setKeystorePath TestCase
      setTruststorePassword TestCase
```

Property	Value
Name	InvokeTestHarnessMethod
Description	

```
CreateGetDeleteFolderCase
FINISHED
TestSteps
DefineBeanValues (6)
DefineTestMethod (1)
SetTestCaseDataValues
SetDebug
SetDebugPath
SetKeyStorePath
SetKeyStorePassword
SetTrustStorePath
SetTrustStorePassword
create
InvokeTestHarnessMethod
```

```
Step 11 [InvokeTestHarnessMethod] OK: took 591 ms
-> invokeTestMethod - OK - 590
TestCase finished with status [FINISHED], time taken = 784
```

```
Thu Feb 24 10:20:22 GMT 2011:DEBUG:Getting free connection, hostConfig=HostConfiguration[host=http://383509-vm-java:8080]
Thu Feb 24 10:20:22 GMT 2011:DEBUG:Freeing connection, hostConfig=HostConfiguration[host=http://383509-vm-java:8080]
Thu Feb 24 10:20:22 GMT 2011:DEBUG:Notifying no-one, there are no waiting threads
```

Figure 31: SOAPUI Test Tool running a Folder CRUD test case

The code to generate the AXIS2 testing web service bean 'FedoraContentHandlerImplSvc' is included with the project source code, as is the SOAPUI project file.

4.3.3 System tests

One of the advantages of using the SOAPUI test tool, is that it also serves as a useful test program that can be used to verify the integrity of the CTREP configuration and installation. We would recommend running the test harness if problems are encountered with getting the CTREP code up and running. Experimenting with different settings (especially for HTTPS) in the mountpoint.properties files, for example, is relatively quick and painless compared with doing the same thing in Sakai (very often a wrong properties setting will crash the Resources Tool in Sakai).

5. Evaluation

5.1 General approach

Evaluation of the CLIF project outputs was an important step in validating the work carried out. This was carried out from two perspectives:

- Testing the outputs against the original aims and objectives of the project (see Annex A of the Project Plan)
- Gathering feedback from end users on the potential of being able to move content between systems to facilitate the digital content lifecycle.

The following approaches were used to investigate these perspectives.

1. An internal project assessment at each site was carried out to test the generic text and data use cases derived from user interviews undertaken as part of work package 3.
2. A demonstration of the outputs from the project was arranged for nominated end users. These were the same as for the original user interviews where feasible, but extended to cover other valid end user groups where appropriate (acknowledging institutional changes since the project began).
3. The generic use cases were used to gather feedback from these end users following the demonstrations, assessing whether they had been met by the outputs. Where there was more detailed user interview information this was also re-visited.
4. Key questions asked when testing the generic use cases were:
    a. How information is captured and stored
    b. How much content is moved between systems and for what purposes (not seeking to identify new use cases as such, but to place the CLIF work in a user context).
    c. What it is considered cannot be done with content in any one system currently used, and how being able to move the content to another system as enabled by CLIF might assist this.
5. An assessment of current institutional preservation policy and strategy was carried out to highlight how the CLIF outputs can contribute towards its further development.

The results of this user assessment are given in the Final Report itself at the end of Section 5 and in Section 6; they are not repeated here.

5.2 SharePoint-Fedora integration evaluation

5.2.1 Technical evaluation

In this section, we review the requirements for SharePoint-Fedora integration listed in Section 2.1.

- Fedora repository (2.1.1). All requirements were fully implemented. For the current prototype we do not allow import of Fedora objects from systems that are not Hydra compliant as described in R1.3, as this would require significant further development of the

Hydranet middleware. For R1.1, we create entries in the Fedora root folders to register new users. This method works satisfactorily for a prototype system where the SharePoint server runs on the same hardware as the Fedora repository. For production environments, where SharePoint and Fedora communicate across a network, a secure protocol would be required for exchange of user credentials.

- Deposit (2.1.2). All requirements were fully implemented. The current CLIF system does not include the facility to make use of versioning in Fedora as described in requirement R2.10, but rather creates new Fedora objects every time a given document is deposited from SharePoint.
- Browse and retrieval (2.1.3). All requirements were implemented. For requirement R3.4, we enabled the user to browse their private repository folders as well as the public Publishable Locations folders.
- Search (2.1.4). All requirements were implemented.
- System administration (2.1.5). All requirements were implemented.
- Excel calculations (2.1.6). Requirements R6.1 and R6.2 were implemented by configuring existing SharePoint features. Deposit to the repository of Excel worksheets as described in R6.3 can be achieved by using the deposit mechanisms already implemented. Given the time limitations, we were not able to provide more functionality aimed specifically at processing of Excel spreadsheets.

5.3 Sakai-Fedora integration evaluation

5.3.1 Technical evaluation

Features implemented.

The great majority of CLIF's original functional requirements have been implemented successfully. In particular, ingest of, browsing of, and retrieval from objects within a controlled Fedora root collection is now possible. This has been carried out in such a way that much of the native functionality of Sakai's Resources tool is available in the context also of virtual resources in a Fedora repository.

Features not fully implemented

Authentication / Authorization: The ideas outlined in Section 4.2.4 of the early CLIF technical design project document[12] were not deemed practical within the project timescale and it was, in any case, considered more fitting to the likely needs of the JISC community to treat the Sakai Fedora mounted collection as an 'open access repository' collection with full read - write access (the Fedora/Sakai administrator having the deciding say on which collection should be made the Sakai mount point collection was considered a more open, flexible and less prescribed approach). At Hull, the institutional repository has multi-level security and so something akin to the original design will be implemented post-project. This enhancement will enable read access below the Fedora node

[12] Awre c, Green R, Thompson A, Waddington S (2010) CLIF technical design See: https://edocs.hull.ac.uk/splash.jsp?parentId=hull:1647%26pid=hull:2697

nominated as the Sakai root and write access only below a lower node which will serve as a deposit point for material to be reviewed and processed by the repository team.

External Web Services: In Section 6 of the same technical design document, mention was made of making both Sharepoint and Sakai configurable to use external web services for harvesting additional metadata to encapsulate in new objects. Again, time constraints have prevented this aspect of CLIF's plans being implemented.

6. SharePoint configuration guide

In this section, we provide details of the configuration of the CLIF system for SharePoint. The code is available at: https://github.com/uohull/clif-sharepoint.

6.1 Setting up the SharePoint site

6.1.1 Creating the SharePoint web application

1. Click the Start button, point to All Programs, then point to Microsoft Office Server, and then click SharePoint 3.0 Central Administration.
2. On the Central Administration home page, click Application Management.
3. On the Application Management page, in the SharePoint Web Application Management section, click Create or extend Web application.
4. On the Create or Extend Web Application page, in the Adding a SharePoint Web Application section, click Create a new Web application.
5. On the Create New Web Application page, in the **IIS Web Site** section, you can configure the settings for your new Web application.
    a. On the Create New Web Application page, in the IIS Web Site section, you can configure the settings for your new Web application.
    b. To choose to create a new Web site, select Create a new IIS Web site, and type the name of the Web site in the Description box.
    c. In the Port box, type the port number you want to use to access the Web application. If you are creating a new Web site, this field is populated with a suggested port number. If you are using an existing Web site, this field is populated with the current port number.
    d. In the Host Header box, type the URL you wish to use to access the Web application. This is an optional field.
    e. In the Path box, type the path to the site directory on the server. If you are creating a new Web site, this field is populated with a suggested path. If you are using an existing Web site, this field is populated with the current path.
6. In the Security Configuration section, leave as it is.
7. In the Application Pool section, create a new application pool by selecting create a new application pool. In the Application pool name box, type the name of the new application pool.
8. In the Select a security account for this application pool section, select Predefined to use an existing application pool security account, and then select the security account from the drop-down menu.
9. Select Configurable to use an account that is not currently being used as a security account for an existing application pool. In the User name box, type the user name of the account you wish to use, and type the password for the account into the Password box.
10. In the Reset Internet Information Services section, choose whether to allow Windows SharePoint Services to restart IIS on other farm servers. The local server must be restarted manually for the process to finish. If this option is not selected and you have more than one server in the farm, you must wait until the IIS Web site is created on all servers and then run iisreset /noforce on each Web server. The new IIS site is not usable until that action is completed. The choices are unavailable if your farm only contains a single server.

11. Under **Database Name and Authentication**, choose the database server, database name, and authentication method for your new Web application.
12. Click **OK** to create the new Web application.

6.2 Setting up MySite in SharePoint

Assumption: The SharePoint Portal is located at http://clif/.

1. First you have to make sure you have 2 managed paths set up in the http://clif web application. To do this jump into central admin, "Application Management" tab, "Define Managed Paths". Make sure you are working on the right web application once you are in that screen.
2. Create an Explicit inclusion for path "mysite".
3. Create a Wildcard inclusion for path "personal".
4. Go back to "Application Management" tab & choose "Create site collection"
5. Create a new site (call it MySiteHost if you like), pick "Create site that this URL" and choose "mysite", also pick the "My Site Host" template (important).
6. Once that site is created jump back into Central Admin, go into your Shared Services Provider configuration, go into the "My Site settings" page.
7. Set "Personal site provider" to "http:// clif /mysite/" and "Location" to "personal", click OK.

6.3 Configuring SharePoint search settings

6.3.1 Create content source

1. To add a new content source to your SharePoint server open the Central Administration of the server and enter the relevant Shared Services.
2. Open the Search Settings page and open the “Content sources and crawl schedules” link.
3. Click the “New Content Source” button and enter name as “CLIF”, Content Source Type as “SharePoint Sites” and Start Address URL as http://clif/ in the “Add Content Source” page and click ok.

6.3.2 Create search scope

1. Open the Search Settings page and open the “Scopes” link.
2. Click the “New Scope” button and enter name as “CLIF” and click ok.

6.3.3 Create search scope rules

1. Click on the “CLIF” scope from the scopes list.
2. On “Scope Properties and Rules” page, click “new rule” link.

   Adding Scope Rule 1

   Select “Content Source” as “Scope Rule Type” and select “CLIF” as the Content Source from the dropdown list and choose “Require - Every item in the scope must match this rule” as the behaviour. Click on ok.

Adding Scope Rule 2

Select “Property Query (Author = John Doe)” as “Scope Rule Type” and under section Property Query, select “contentclass” as the “Add property restrictions” and “STS_ListItem_20007” from the dropdown list and choose “Include - Any item that matches this rule will be included, unless the item is excluded by another rule” as the behaviour. Click on ok.

Adding Scope Rule 3

Select “Property Query (Author = John Doe)” as “Scope Rule Type” and under section Property Query, select “contentclass” as the “Add property restrictions” and “STS_ListItem_20008” from the dropdown list and choose “Include - Any item that matches this rule will be included, unless the item is excluded by another rule” as the behaviour. Click on ok.

6.3.4 Adding managed properties

1. Open the Search Settings page and open the “Metadata property mappings” link.

Adding Managed Property 1

a. On Metadata Property Mappings page, click “New Managed Property” button.
b. Enter “PersistentID” as the Name and type.
c. Add “ows_Persistent_x0020_ID” as the Mappings to crawled properties and check “Allow this property to be used in scopes” and click ok.

Adding Managed Property 2

a. On Metadata Property Mappings page, click “New Managed Property” button.
b. Enter “ListId” as the Name and type.
c. Add “ows_ListID” as the Mappings to crawled properties and check “Allow this property to be used in scopes” and click ok.

6.3.5 Crawl SharePoint site content

1. Go to the central administration of your SharePoint Server
2. On the left side under ‘Shared Services Administration’, click on ‘SharedServices1′ or an equivalent of it
3. In the area ‘Search’, click ‘Search Settings’
4. There, click ‘Content sources and crawl schedules’
5. Finally, find the arrow on the right end of the name of ‘CLIF’, click it and start the crawling. Attention: If you start a Full Crawl, it may be that you have to wait a really long time.

6.4 Configure Excel web services is the SharePoint site

6.4.1 Overview

Excel Web Access (EWA) is a Web Part that allows a user to open an Excel workbook in a browser and allows users to interact with the workbook in the browser with the familiarity of the Excel client. It displays and enables interaction with the Microsoft Office Excel workbook in a browser by using Dynamic Hierarchical Tag Markup Language (DHTML) and JavaScript without the need for downloading ActiveX controls or additional software on the client computer. This component can also be connected to other Web Parts on dashboards and other Web Part Pages to provide more advanced capability. Excel Web Services (EWS) is a service used to programmatically access Excel workbooks stored in MOSS 2007. This Web service is hosted in MOSS and provides an application programming interface (API) to build custom applications based on the Excel workbook.

6.4.2 Configure Excel calculation services

1. On the Start menu, click All Programs.
2. Point to Microsoft Office Server and click SharePoint Central Administration.
3. Go to the Shared Services Provider (SSP) Administration site for the Web application that's hosting the Web site.
4. Under Excel Services Settings click trusted file locations.
5. Click Add Trusted File Location. Unless Trusted File Location is configured in SharePoint, Excel Services will not work.
6. In the Address box type the URL to the trusted file location, e.g. http://clif/Project Documents/Forms/AllItems.aspx.

6.4.3 Enable user defined functions

1. On the Start menu, click All Programs.
2. Point to Microsoft Office Server and click SharePoint Central Administration.
3. On the Quick Launch, click your Shared Services Provider (SSP) link—for example, "SharedServices1"—to view the Shared Services home page for that particular SSP.
4. Under Excel Services Settings, click User-defined functions.
5. On the Excel Services User-Defined Functions page, click Add User-Defined Function to open the Excel Services Add User-Defined Function Assembly page.
6. In the Assembly box, type the path to the UDF assembly. For example, C:\MyUdfFolder\MyUdf.dll.
7. Under Assembly Location, click Local file.
   Note: -- The Local file option will be replaced with File path in future releases of Excel Services. If you see File path, select that instead.
8. Under Enable Assembly, the Assembly enabled check box should be selected by default.
9. Click OK.

6.5 Configuring Fedora

6.5.1 Creating root folder

Create a “CLIF:Root” content object that acts as a root container object for the SharePoint Site.

6.5.2 Create publishable locations folder

Create a publishable locations root content object where documents can be published from any SharePoint MySite.

6.6 Configuring root SharePoint MySite site

Projects List:

Add new projects in the “Projects” List in the MySite which provides lookup values for the “Project Documents” Content Type.

Content Sources List:

Add all available content sources in the “Content Sources” List in the MySite which provides lookup values for “Project Documents” Content Type.

Content Languages List:

Add all available content languages in the “Content Languages” List in the MySite which provides lookup values for “Project Documents” Content Type

6.7 Deploy SharePoint solution

6.7.1 Adding CLIF.Solutions.wsp to the server

STSADM command to add the solution:

```
stsadm -o addsolution -filename <Solution Path>.
```

6.7.2 Deploying CLIF.Solutions.wsp to a SharePoint site

STSADM commands to deploy the solution:

```
stsadm -o deploysolution -name “CLIF.Solutions.wsp” -immediate -url “http://clif/”
-allowGacDeployment -allowCasPolicies -force

stsadm -o execadmsvcjobs
```

6.8 Configuring root SharePoint site

6.8.1 Activating features

Assumption: SharePoint Portal at http://clif/

Attention: All features must be activated in the same order as listed below:

1. CLIF.XSL

   ```
   stsadm -o activatefeature -id “9394c71-006d-4594-85b6-6c1006785a46" –url
   “http://clif”
   ```

2. CLIF.Scripts

   ```
   stsadm -o activatefeature -id “B0A0A13D-FC80-408e-8F61-E36FEFC29F98" –url
   “http://clif”
   ```

3. CLIF.Styles

   ```
   stsadm -o activatefeature -id “b80cce94-386b-4124-812a-f9c4565afd01" –url
   “http://clif”
   ```

4. CLIF.Images

   ```
   stsadm -o activatefeature -id  “b65e6fd8-4874-4eaa-93c3-c16447343aae" –url
   “http://clif”
   ```

5. CLIF.MasterPage

   ```
   stsadm -o activatefeature -id  “7306b8a0-7490-4bfc-9756-9b0a31d5065c"  –url
   “http://clif”
   ```

6. CLIF.Search

   ```
   stsadm -o activatefeature -id  “c683eef0-a0d9-479d-886c-bf6ae02817ee"  –url
   “http://clif”
   ```

7. CLIF.Fields

   ```
   stsadm -o activatefeature -id  “8fcd6181-9fbd-49ad-9983-a6f483ceb4c6"  –url
   “http://clif”
   ```

8. CLIF.ContentTypes

   ```
   stsadm -o activatefeature -id  “7d7aefc4-8494-4ce7-875a-ff08212dd403"  –url
   “http://clif”
   ```

9. CLIF.LookUpList

   ```
   stsadm -o activatefeature -id  “52fb0409-3619-4b04-a7dc-477bfff63367"  –url
   “http://clif”
   ```

10. CLIF.LookUpWithPicker

```
stsadm -o activatefeature -id  “10c19a31-9710-42d9-a1f2-4ac3f4aabb2d"  –url
“http://clif”
```

11. CLIF.Lists

```
stsadm -o activatefeature -id  “59d73371-02cb-4b45-a203-cd7e4904f27b"  –url
“http://clif”
```

6.8.2 Setting up lookup list

Publishable Location List:

Add entries in the “Publishable Locations” List in the root site that points to a “Content Object” in Fedora by specifying a Persistent ID.

7. Sakai configuration guide

In this section, we provide details for installing the CLIF code onto a Sakai server.

7.1 Pre-Requisites

- Sakai 2.6.1 or higher
- Fedora 3.4 or higher repository configured with:
  - API-A, API-M allowed for external access
  - Hydra Content Model objects pre-populated

7.2 Installation

The following advice assumes a Windows Tomcat server system; change paths accordingly on a Linux /Solaris/Apache system. Also there is a Hull-specific adaption of the code in the SakaiHull subfolder of the GitHub source repository https://github.com/uohull/clif-sakai.

1. If wanting a brand new vanilla 2.6.1 download full source for Sakai-2.6.1 and kernel-1.0.12, else proceed to step 2.

2. Into another directory download source from https://github.com/uohull/clif-sakai

3. For a brand new 2.6.1 vanilla install, copy Sakai-2.6.1 and kernel-1.0.12 subfolders over vanilla folders. For all other Sakai versions either
   a. Use a merge tool such as WinMerge to merge the two folder structures together, and handle any conflicts by manually editing the conflicting files
   b. Use a tool such as TortoiseSvn to generate a unified ‘diff’ between the two folder structures and apply the resultant patch

4. Using Maven (version 2.2-1 was used here), build Sakai and Sakai Kernel in the normal way :

   e.g `mvn clean sakai:deploy -Dmaven.tomcat.home=c:\apache-tomcat-5.5.28`

5. Due to a fault with the Maven POMs for deploy you may find you need to manually copy some of the jars within C:\apache-tomcat-5.5.28\components\sakai-content-chh-pack\WEB-INF\lib to C:\apache-tomcat-5.5.28\components\sakai-kernel-component\WEB-INF\lib before starting the apache web server.

6. Before starting Apache or Tomcat add following to sakai.properties file :

   ```
   content.useCHH=true    // this enables the Mountpoint functionality
   ```

7. Create a new blank site for accessing resources in Fedora and include in that site the standard resources tool

8. In the Fedora Site resources tool upload a suitable mountpoint.properties file. Edit the activate mountpoint value to be 'uk.ac.uhi.it.ContentHostingHandlerImplFedora'.

9. If communications between Sakai and Fedora are successful, the user should see a list of the resources in the specified fedora collection (clif:fedora-root-collection setting).

8. Acronyms and abbreviations

API	Application Programming Interface
API-A	The Fedora access API
API-M	The Fedora management API
CARET	Centre for Applied Research in Educational Technologies (at Cambridge University)
CHH	Content hosting handler
CHS	Content hosting service
CLIF	Content Lifecycle Integration Framework
CMA	Content Model Architecture (a Fedora term)
CMS	Content Management System
CRUD	Create, read, update, delete
CTREP	Cambridge Tetra Repositories Enhancement Project
DC	Dublin Core – a particular form of metadata
ESB	Enterprise Service Bus
Fedora	Flexible, extensible, digital object repository architecture
FOXML	Fedora Object XML – a form of XML used for encoding Fedora repository objects
HEI	Higher Education Institution
HTTP	Hypertext Transfer Protocol
HTTPS	Secure Hypertext Transfer Protocol
ITQL	A query language used with RDF
JIRA	A proprietary bug tracking system developed by Atlassian
JISC	The Joint Information Systems Committee
JKS	Java keystore
JMS	Java Messaging Service
LDAP	Lightweight Directory Access Protocol: LDAP servers frequently form the basis of institutional network user security systems in universities.
LMS	Learning Management System
METS	Metadata Encoding and Transmission Standard
MIME (-type)	Multipurpose Internet Mail Extension
MOSS	Microsoft Office SharePoint Server

PDF	Portable Document Format
PID	Persistent identifier
QA	Quality assurance
RDF	Resource description framework
RELS-EXT	A Fedora object reserved datastream for dealing with external relationships
REST	Representational State Transfer
SOAP	Simple Object Access Protocol – although use of this expansion is now deprecated
SWORD	Simple Web-service Offering Repository Deposit
UHI	University of the Highlands and Islands
UI	User interface
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
URN	Uniform Resource Name
VLE	Virtual Learning Environment
VRE	Virtual Research Environment
WSS	Windows SharePoint Services
XACML	Extensible Access Control Markup Language
XML	Extensible Markup Language
XSLT	Extensible Stylesheet Language Transformations