Intersection between Sakai and Fedora

Edited by Peter Murray, OhioLINK. This document represents a summary of comments on the Sakai Developers mailing list on April 26-28, 2006 to a question posted by this document's editor regarding possible integration points between Fedora and Sakai. The resulting threads were:

- Seeking status of 'Rep API' and 'JSR-170' shim from "Sakai Repository API Design"
- Repository integration with Sakai
- Fedora in Content Hosting - An Alternative

The editor is indebted to the public and private responses of: Ben Brophy, Ian Dolphin, Glenn Golden, Adam Hochman, Jeffrey Kahn, Beth Kirschner, Peter Knoop, Anoop Kumar, Jim Martino, Mark Norton, and Charles Severance. Information from these individuals is supplemented by additional research by the editor, and the whole should not be viewed as a coherent or even advisable strategy. Direct quotes in this summary are not attributed to individuals, and the editor alone is responsible for omissions and quotes-out-of-context errors.

The editor's question related to the integration of Sakai and Fedora. OhioLINK's desire with the Digital Resource Commons is to build a unified content repository by meeting Sakai (and other tools such as OJS/DpubS for electronic journals, etc.) right where they store the data. In doing so, it is our expectation that allowing data to flow across applications will be much easier because there is one underlying repository.

Sakai Infrastructure

Description of ContentHostingService interface

ContentHostingService is one of the "backbones" of Sakai. To make Fedora the content repository (i.e. the attachment helper and resources tool, presentation tool, and any other tool using "content" in Sakai) then one must implement the ContentHostingService API. The API is found in https://source.sakaiproject.org/svn/content/ as org.sakaiproject.content.api.ContentHostingService (JavaDoc) and the baseline implementation can be found as BaseContentService.

The ContentHostingService API supports WebDav, the AccessServlet, the WebServlet, Resources Tool, Presentation Tool, interoperates closely with Sakai's Authorization model, generates well-formed Sakai notifications, and real-time events, and is cluster aware and on and on. This is a key API used heavily by Sakai; an alternative implementation probably needs to implement it completely to be of any use.

Plugin Pattern

Most current open-source repository systems cannot handle the on-line performance demands of a large-scale Sakai installation. As an example, a Sakai installation supporting 500 users could easily store all of its resources in a separate repository. But a system supporting 100,000 users may not be able to "outsource" all of its storage.

What is needed is a sophisticated set of plug-ins that allow great flexibility for different Sakai installations. Instead of a complete re-write of the ContentHostingService interface, a plugin pattern that allows ContentHosting to precisely outsource things like blob storage and metadata storage with simpler APIs than ContentHosting and a very clean and simple API contract. Then one can write plugins for things like Fedora, JSR-170, DSpace, iTunes, etc.

Extend ContentCollection

Suppose one were to extend the existing ContentHostingService, specifically the ContentCollection class (JavaDoc) such that it serves as a gateway to a repository. It would need to represent the protocol to be used (Fedora, SRA, JSTOR, whatever) and likely include at least some support for digital rights management.

A simple idea with a lot of consequences, both easy and hard. On the plus side, repository-based collections could either be tied to a search pattern on the repository,or tied to a specific remote collection (including the root). Done right, it would integrate seamlessly into existing Sakai tools, such as the ResourceTool, etc. On the minus side, ContentResource might also be affected, since the tools would list content that is no longer directly accesible via the current implementation. WebDAV integration might also be challenging.

In the intial stages, this would have to be an optional feature that could be enabled or disabled since complete integration may not come quickly. Still, one could have an instance of the ResourceTool tied to a specific repository (not mixed) that could be billed as not supporting WebDAV or other regular content hosting features.

ContentHostingService refactoring

Ian Boston of Cambridge University reports development of a plugin into ContentHostingService that enables an implementing class, once it has registered itself with the ContentHostingService to be consulted if it 'owns' a node in the content hosting tree. Once it 'owns' the node, it takes responsibility for providing ContentResources for that node and all child nodes.

Cambridge has used this to implement an IMS Global Learning Consortium Content Packaging specification (IMS-CP) plugin, that 'plays' IMS-CP files but one could just as easily use it to 'mount' a repository inside the ContentHostingService at an arbitrary position.

If the resource is read-only, or can't be accessed via DAV, then it is the responsibility of the plugin to act accordingly (ie, not return the content).

Cambridge is also looking to write a plugin that 'interprets' an XML file loaded into resources to invoke a UI. With Fedora, that XML file might be the connection settings to the repository.

Since the plugin patch is small (<50 lines), and the plugins can be implemented inside webapps (ie reloadable), Ian is hoping that it will be useful and can be included in the BaseContentService (JavaDoc) for 2.2. If it isn't, Cambridge will maintain it as a patch as it as they are going to be running it in production in July. See SAK-4457.

JSR-170 interface

JSR 170 is the Content Repository API for Java. It specifies a standard API to access content repositories in Java independent of implementation. (See also its follow-on: JSR 283 Content Repository API version 2.0). Neither Fedora nor Sakai have JSR-170 interfaces, although if they did it could be a point of integration. Implementation of JSR-170/283 would be a new implementation of the Sakai ContentHostingService API.

See JSR-170 for Ian Boston's description of a JSR-170 Implementation of ContentHostingService as a prototype against the then Trunk 2.2 ContentHostingService.

AFS interface

UC Davis people have home directories in AFS space, and their course management system also utilizes this space. UC Davis has a problem statement for integrating the user home directory AFS space into Sakai.

Alternate Integration Points

OKI Repository OSID as a Sakai Tool/Service

Sakai offers Repository integration through the O.K.I. Repository OSID. The Repository OSID is used in a small (but perhaps growing) number of places in Sakai as a secondary interface to external repositories — it does not go through the main ContentHostingService interface. If this bridge were wide enough to include all the functionality Sakai needed, these calls could go through OKI. (Additional information is on confluence, implementation is tracked as SAK-2327, and an example application is Twin Peaks.)

One implements a set of interface methods that talk to your repository and then applications (one of which is Sakai) can search, retrieve, and upload content to your repository, subject to permissions. Since the interface is standard (maintained by the IMS Global Learning Consortium), the calling application can use the same approach for all repositories. The caller is also insulated from connection, protocol, and some data format specifics.

The Fedora OKI Bridge under development by Anoop Kumar of Tufts for VUE release 1.5 (expected June 1, 2006) will support following:

- It will work with Fedora 2.0
- It will provide read-only support.
- It provides ability to perform simple and advanced search.
- It will support three types of RecordStructure(OSID).
- Dissemination RecordStructure holds all disseminations of object as its parts.
- Image Record Structure will provide thumbnail, medium and high res images.
- VUE Default Record Structure will provide a default view for digital object and dublin core metadata.

Sakai may be able to use the bridge as is or we can add additional record structures to support Sakai's needs.

Sakai's Repository OSID is being considered as an integration point with the DSpace Institutional Repository application.

UMichigan Fedora Tool

Beth Kirschner at the University of Michigan is developing a Fedora repository interface for XSLT-driven CRUD operations against Fedora objects. The current implementation is a prototype, supporting browse, search and editing capabilities. Authorization is not yet integrated with Sakai. For more information, see: source.sakaiproject.org¿README.txt.

(Note: this document was originally posted to the web as Fedora plus Sakai: A view from 30,000 feet. That version will not be updated; this version is considered authoritative.)