Strategic Directions for Sakai and Data Interoperability

Charles Severance (csev@umich.edu), Joseph Hardin (hardin@umich.edu)

Sakai is emerging as the leading open source enterprise-class collaboration and learning environment. Sakai's initial purpose was to produce a single application but increasingly, Sakai will need to operate with other applications both within and across enterprises.

This document is intended to propose a roadmap as to where Sakai should go in terms of data interoperability both amongst applications running within Sakai and with applications interacting with Sakai. There will be other aspects of future Sakai development that are not covered here. This document takes a long-term view on the evolution of Sakai - it is complementary to the Sakai Requirements process, which captures short to medium term priorities. The tasks discussed in this document should not be seen as the "most important" tasks for Sakai. The document only covers data exchange aspects of Sakai.

Nothing in this document is set in stone - since Sakai is a community effort, everyone is a volunteer. However, by publishing this overview we hope to facilitate consensus and alignment of goals within the community over the longer-term future of Sakai.

Sakai is currently a Service-Oriented-Architecture. This work maintains and enhances the Service-Oriented aspects of Sakai and adds the ability to use Sakai as an set of objects. This work move be both Service Oriented *and* Object-Oriented.

Making Sakai Web 2.0

While "Web 2.0" is a vastly overused term, in some ways, this effort can be likened to building the Web 2.0 version of Sakai.

According to an article written by Tim O'Reilly [9] there are a number of core competencies that are the hallmarks of Web 2.0:

- Services not packaged software
- Architecture of Participation
- Cost-effective scalability
- Remixable data sources and data transformations
- Software above the level of a single device
- Harnessing collective intelligence

While it is difficult to summarize such a broad concept as Web 2.0 in a simple

statement - the following is an attempt:

- Web 1.0 was about building many stovepipe applications and using a common presentation technology (browser) allowing users to make use of literally millions of independent web-based applications.

- Web 2.0 is about finding ways to take the data stored in those applications and moving the data between those applications and allowing the data to be useful and usable outside of the original application.

You can almost view the trend towards Web Services as "Web 1.5". Web services allowed some data to be exchanged between cooperating applications - but usually required a specific negotiation between pairs of applications to establish the data exchange protocols.

Web 2.0 is about a far more general and generic ability to share data stored in one application and make use of the data in another application using generalized data protocols and representations. Web Services is one of the important aspects and is an important underpinning to many of the Web 2.0 trends.

Interestingly, if you look across seemingly divergent trends in the industry such as REST[10], RSS [11], and RDF[12] the common thread across all three efforts is about making an application's internal resources accessible to other applications as shown by the following quotes:

- *With REST, the Web is comprised of resources. A resource is any item of interest. When a URL is accessed, a representation of the resource is returned. [10]*

- *An RSS feed contains a list of items or entries, each of which is identified by a link. Each item can have any amount of other metadata associated with it as well. [11]*

- *The Resource Description Framework (RDF) is a language for representing information about resources in the World Wide Web. [12]*

The major difference is the approach used to serialize and then retrieve the data elements.

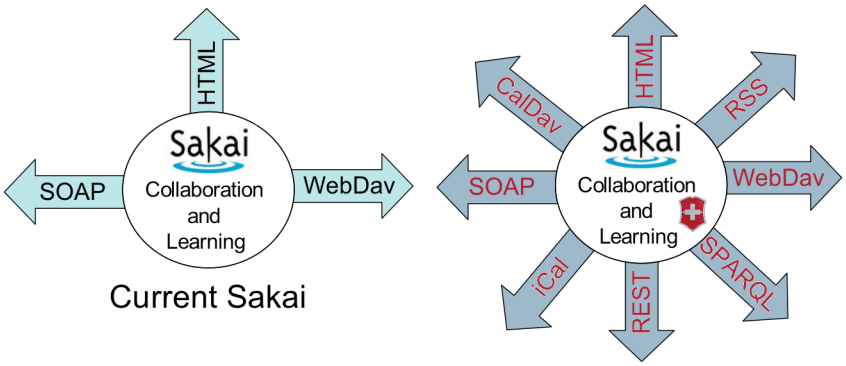
Sakai finds itself in a very enviable position when looking towards Web 2.0. Because Sakai has espoused and heavily uses a service-oriented architecture (SOA) from its inception, Sakai is well positioned to move into Web 2.0 technologies and protocols in a way to provide significant impact. Sakai already

has strong support for SOAP and WebDav so we could almost classify Sakai as "Web 1.5".

Introduction

A key objective that we will need to accomplish going forward is to free the knowledge captured in the system for use in new ways be other systems. The data stored within Sakai will need to be exchanged with many other applications. In order to do this, Sakai will need to implement a number of different protocols so as to exchange data with the widest possible range of applications.
The ultimate goal is to allow the end-users to make use of a far wider range of software elements. Without rich data interoperability and data portability, users can only access the data and information stored in Sakai using the Sakai software. If the data can be easily moved between applications, then users can make use their Sakai data with all of these applications.

Ultimately we need to evolve Sakai to the point where it is a veritable "Swiss-Army Knife" of data exchange protocols.



This is not as simple as just implementing the protocols as separate efforts. In effect these different protocols are simply different ways of rendering the data elements stored within Sakai. We must first build basic infrastructure and capabilities to properly model data held within Sakai and then this data once modeled can be exchanged using many different protocols.

This will take several steps:

- Accurately model Sakai's internal business objects so they can be fully represented in any needed data exchange protocol. Currently these business objects (Users, Sites, Resources, etc) are modeled using the Sakai APIs. Sakai's current web services are a convenient way to access these business objects via their APIs. The goal is to be able to directly retrieve the business objects themselves using various protocols instead of remotely calling Sakai's APIs to manipulate the objects.

- Build support for making these business objects available using a wide range of data exchange protocols including SOAP, RSS, iCal, WebDAV, CalDAV, REST, RDF SQARQL and others. We should prioritize the protocols chosen based on those which will provide the best reuse of the information stored in Sakai. Broad support for RSS should be an early focus as should iCal support. Ultimately we would expect that RDF SPARQL will be the most general protocol for accessing Sakai business objects.

- Evolve Sakai's application services to not only model business objects within each application, but also model business objects across application areas. This will abstract references between different Sakai services so that it is possible for an external application to retrieve a set of business objects and then resolve references to other business objects within Sakai. This will move Sakai toward a semantic network of all of the business objects within Sakai. This semantic network of internally cooperating objects will be reflected to the outside world via the Sakai data interchange protocols. Using this semantic network, an external application will be able to retrieve a set of business objects and then successively retrieve business objects that are related to and referenced by the first set of business objects.

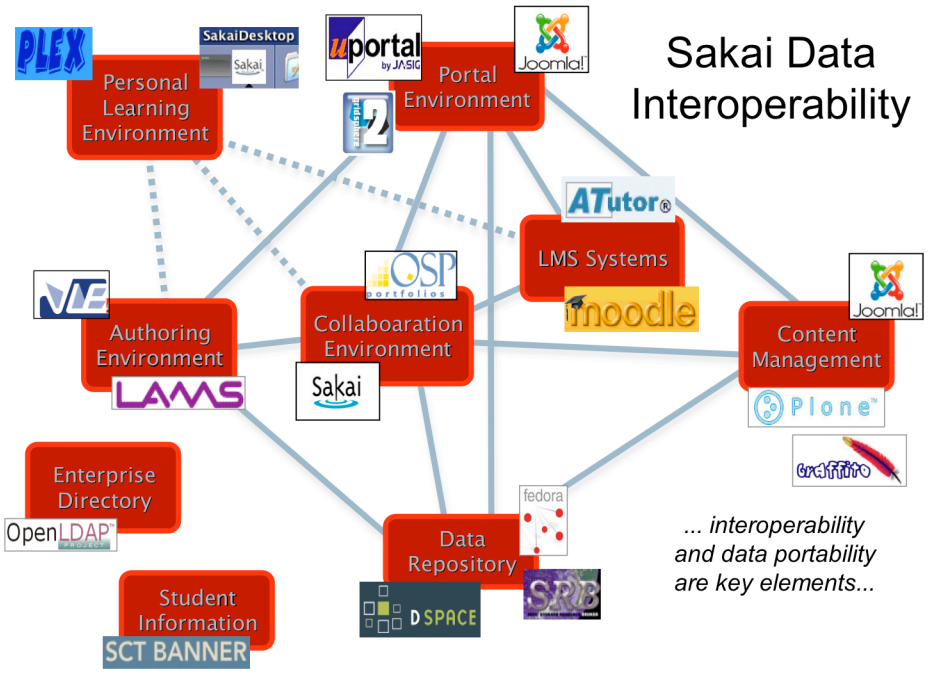
Using this approach, the data stored in Sakai can be reused in a wide variety of applications both existing and prospective. The Sakai community will need to work with the communities that support products with which we need to exchange data and validate data exchange.

Exploring the new Use Cases Enabled by this Work

This section relates Sakai to the other applications that may be encountered in a typical Enterprise and explores the use cases enabled by allowing these other applications to access the data stored in Sakai.

In some situations, there will exist a rich set of enterprise applications with which Sakai will need to integrate. In other cases, Sakai will be the core of a smaller enterprise and Sakai many pave the way for new enterprise applications such as a portal for these smaller organizations.

Sakai already is set up to integrate with Enterprise Directory and Student Information Systems using a service-oriented approach. Typically all of the Enterprise applications integrate with these directory and/or student information systems and so the relationships are not explicitly drawn for these two elements in the figure.



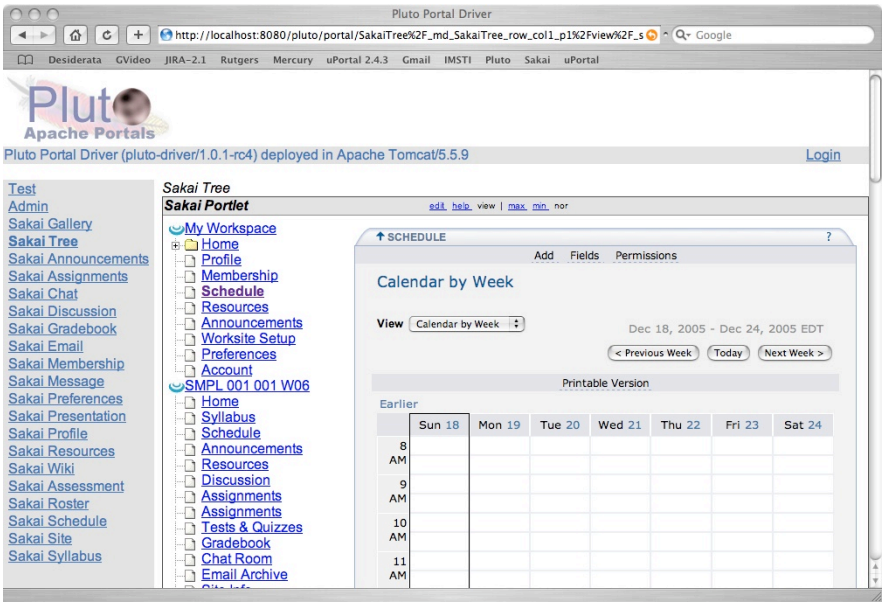
Ultimately as an organization assembles these applications they are creating what many call an "Enterprise Ecology" - this terminology is alluding to the idea that these applications operate in an environment with a rich set of data.

This section will cover each of the elements in this figure and cover use cases that would be enabled by this work.

Portal Environment

One of the more common points of integration for Sakai is with an enterprise portal such as uPortal[1] or others. Portals satisfy two basic sets of requirements. Organizations use portals to give their customers "one-stop-shopping" for the services provided across the organization. Some organizations allow significant end-user customization of their portal so that their users begin to use the portal as their "personal starting point" when working with the institution,

Sakai currently has two main ways that it integrates into portals: (1) simple iframe inclusion in the portal and (2) a set of JSR-168 portlets which act as proxies for Sakai functionality. These portlets use a combination of SOAP based web services and iFrames.



Sakai's JSR-168 Portlets

Probably the most important new portal functionality is to provide a rich set of RSS feeds. Nearly all portals have a highly configurable RSS reader. RSS can provide a very natural dashboard for Sakai. Sakai needs to provide RSS feeds at a server, site, and tool level. These RSS feeds must support authentication and produce user and context specific information in the feeds as appcopriate.

The next most important protocol for portals is to implement the iCal protocol. It will be necessary to provide iCal feeds for an individual, course, or project site.

Sakai also provides a WSRP 1.0 capability that is not yet production ready. To be production ready this requires additional investment in improving Sakai's support for WSRP[15] and then some investment in making the Sakai tools function better in the WSRP environment. This is a pretty significant task and unless a very strong use case appears which galvanizes a set of community resources into action, it appears that WSRP 1.0 may languish. Another problem with WSRP 1.0 is the fact that the Apache WSRP4J project has languished in Incubator mode for many years now with very few resources invested in the project. Without a solid reference implementation available, interoperability of WSRP implementations is very poor.

With the WSRP 2.0 and JSR-286 on the horizon, perhaps the time for WSRP 1.0 has passed. Sakai *will* need to support the new WSRP 2.0[2] and JSR-286[3] standards as well. These standards will support local and remote portlets in the next generation of standards-based portals. Both of these standards are currently under development and Sakai is participating in the JSR-286 expert group. By participating in these standards Sakai can get a heads-up as these standards

emerge into the market.

Data Repositories

Sakai already has efforts to integrate with data repositories. The TwinPeaks project (now Mellon-funded and called SakaiBrary) is exploring how to search repositories and include the results of those searches within Sakai. A Hewlett-funded project involving is investigating how to naturally move data from Sakai into EduCommons to begin to deliver on improving the workflow between a CLE such as Sakai and a the long term archival and serving of course materials in systems such as OCW and EduCommons. Other projects are looking at interoperation with institutional repositories such as Dspace.

In this section, we explore both the long term and short-term approaches for exchanging data with repositories.

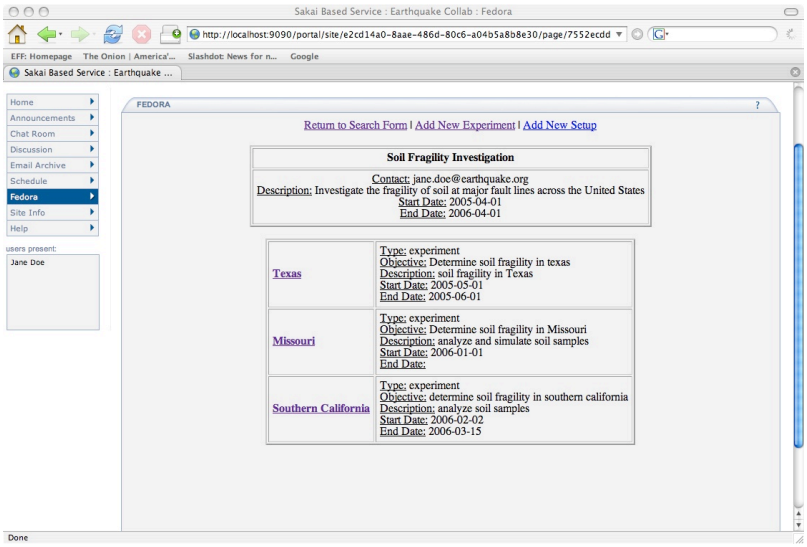
Long Term Approach to Repository Data Exchange

The long-term approach to repository exchange should be based on RDF. By supporting RDF export in Sakai and RDF import in data repositories, the repositories will be able to get 100% perfect copies of Sakai sites. Each repository may only understand a small fraction of the Sakai structures but it can store the entire structure. This allows Sakai data to move between repositories that support RDF with no loss of information. This satisfies an important use case where repository technology may need to change every 10-12 years or so.

Sakai should support two forms of RDF data export. The first should be a one-time export where all of the Sakai data objects are put into a single file (or files) and transferred to the repository in a single step for archiving and preservation. The second form of export should be "live" export (i.e. synchronization) where the repository will periodically retrieve new objects from Sakai by periodically looking through Sakai's internal structure and updating the copy of the information stored in the repository.

These efforts will require coordination between Sakai and Fedora, DSpace, and other repository projects. All of these projects have RDF on their roadmap.

Sakai already is doing some work exchanging data with Fedora:



This work is in the very early stages and is using the existing capabilities of Fedora - this work should naturally evolve to use RDF working closely with the Fedora team. The above work is targeted at using Fedora in an eResearch/eScience application.

IMS Common Cartridge

While RDF data exchange is very important for the long term, in the short to medium term, the IMS Common Cartridge format will be a natural exchange format for course material between Sakai and repositories. IMS CC is an XML schema-based standard and as such there is some information lost in translation. That said, IMS CC has the advantage of being a standard so we expect that other LMS systems will adopt it. This will allow repositories such as DSpace and Fedora to naturally interact with a wide range of LMS systems.

The IMS Common Cartridge [4] effort is defining a rich profile within IMS Content Packaging which will lay out precisely how to bundle QTI formatted test pools and questions, files, discussion threads, modules, SCORM content, etc into a single IMS Content Package file. This effort is well underway with cooperation between Sakai, Blackboard, WebCT, Angel, Pearson, and McGraw Hill. This standard will take several years of implementation experience and evolution before it is truly useful and in the hands of the publishers and the users.

As useful as it will be, IMS CC will only provide a base level of data exchange when it comes to Sakai's broader Collaboration and Learning scope. Sakai will expand rapidly with new tools and new types of data that will far outpace the relatively slow process of IMS standardization and the XML-Schema based approaches to data interchange. To move to the point where repositories can import 100% of Sakai's internal stored data objects, we will need to adapt to RDF as a medium of exchange.

Extending Sakai's Repository Plugin Capability

Sakai currently supports the inclusion and searching of read-only repositories through the Digital Repository OSID. This will need to be extended to allow Sakai data to be written to external repositories as well. This would allow either the replication of Sakai resources or actual storage of Sakai resources in data repositories. To accomplish this a set of plugins will be developed for the Sakai ContentHostingService to allow these writable repositories. We will investigate support for both a writable version of the DR OSID as well as JSR-170[5] for this application.

Other Learning Management Systems

While we initially think that many sites will have a single learning management system, often there are capabilities present in one learning management system that are very unique. As an example, the LonCAPA[6] system is very well suited for testing and homework for Physics, Mathematics, and Chemistry. However LonCAPA is typically too complex for the average non-technical user wanting to build a simple quiz. Since LonCAPA is written in perl, it cannot run "inside" of the Sakai servlet container - but users will be greatly benefited if LonCAPA were available within Sakai.

Another use case would be where it would be useful to include a Sakai tool from within a commercial LMS such as Blackboard.

The primary guideline in this area is the IMS Tool Interoperability [4]. This standard was developed by a pioneering effort that included Sakai, Blackboard, WebCT, University of Wisconsin, Samigo, and QuestionMark. The first version of the standard is now complete but in not in use in any production environments at this time.

We need to invest in moving the IMS TI support in Sakai to production readiness. Sakai will soon support the inclusion of external tools using IMS TI in consumer mode and then we will need to work to allow other LMS systems to access Sakai tool instances using IMS TI producer mode. We expect that the commercial LMS systems will naturally implement IMS TI consumer mode, but they will not naturally implement IMS TI producer mode. We see Sakai as needing to take a lead to be the first LMS to implement IMS TI producer and then exploring the interaction between producer and consumer very deeply.

There are a number of important areas where Version 1.0 of Tools Interoperability needs improvements including improved discovery and provisioning mechanism, and a richer ability to return results from the included tools back to the LMS system. These improvements will only happen as IMS TI sees greater use. So far while the commercial vendors are fully in support of IMS TI standardization efforts they have not included IMS TI in any of their releases

so it may fall to Sakai to push IMS TI into production use.

Support for IMS Common Cartridge (described above) will enhance the data interoperability in the form of content exchange between Sakai and other Learning Management Systems. By providing a cross-LMS standard, IMS CC has the possibility of making it possible for repositories to provide base-level support for moving course material between a wide variety of vendors.

IMS CC also has the possibility of allowing publishers to produce a single course package that can be loaded into any learning management system. It is likely that the commercial LMS vendors will focus on IMS Common Cartridge as an import format. Sakai needs to invest in pushing to make use of IMS CC as an export and exchange format.

We expect that once IMS has well defined standards for tool interoperability and content exchange, we hope to move IMS in the directions of defining and building increasingly rich web services that will allow increasingly sophisticated online LMS to LMS interactions. Another area is to encourage IMS to begin to explore the use of RDF based standards in addition to XML Schema based standards. Remaining deeply involved in IMS is an important area for Sakai to help increase the capabilities available to the end-users regardless of the choice of their main LMS.

Content Management

Content Management systems such as Graffito, Joomla (i.e. Mambo), and Plone are increasingly becoming an important part of any organization. Sakai's main site (www.sakaiproject.org) uses Mambo rather than Sakai because Mambo is better suited for the document handling, public relations, marketing, and outreach uses.

Usually content management systems are centered on documents and workflow (much like an electronic magazine). These systems are designed to allow a marketing/web team to produce a rich and dynamic site without requiring continuous programmer intervention.

Sakai will need to interact with these systems in several ways. First Sakai will need to allow these systems to make use of Sakai tools such as calendar, resources, announcements, etc and rearrange the instances of these Sakai tools based on the needs of the site publishing team. In many cases Sakai worksites will have members who are nominated to maintain an associate public website or area within a public website. Integration with Sakai should enable this scenario to be as seamless as possible for the group/course. This will require support for SOAP web services, support for authorization exchange between the content systems and Sakai, support for the retrieval of individual tool instances and

placing them anywhere on the content management site.

A second important area of interaction will be to allow Sakai resources (files and data) to be usable within these content systems. Effectively it would be good if Sakai could be used to author and create documents and files and then these files would transparently be available to the content management system. This will require the use of SOAP web services, basic URL-style access to Sakai resources, and authorization support to allow the content management system to gain access to resources and files within Sakai.

Sakai already has a number of features that can be used to provide some initial support for these content managements systems. The first task will be to begin to work within one or more content management systems to properly make use of the existing Sakai capabilities. We will start working with Joomla because it is quite popular and PHP-based. The next system we will consider is PLONE because it is Python-based.

Part of the goal here is also to improve the overall connection between Sakai and non-Java systems. By building infrastructure and example code in these simpler languages, we hope to enable a large group of application developers who do not have Java programming experience. We expect that this will enable a whole new rich series of applications that work with Sakai.

Also there is a significant overlap between the needs of a portal and the needs of a content management system. Features such as RSS and iCal will also be useful to content management systems as well.

Authoring Environments

When comparing Sakai to commercial learning management systems and other open source learning management systems Sakai's most glaring shortcoming is the limited flexibility which a teacher has in organizing Sakai capabilities into a rich course site.

Sakai needs to improve both its internal authoring approaches as well as work with external authoring systems. Both are covered in this section.

Authoring using VUE

Discussions are already underway connecting Sakai with VUE. The integration work is the second phase of the currently funded VUE work. While the design of the interaction is not completed, some initial ideas are in place.

The exchange of data between Sakai and VUE will be in the form of RDF. There are two basic use cases which we will likely implement. The first use case is to

allow the storage and retrieval of VUE diagrams in Sakai.

The second use case is to make all of the Sakai objects available to VUE during its authoring and then playback phase. During the authoring phase we will likely make use of a combination of RSS with additional information included the RSS payload so VUE has full information about the Sakai objects referenced in the RSS. VUE will be able to retrieve related objects as well as the original objects retrieved in the RSS. This will allow VUE to navigate through the network of Sakai objects using RDF[12], RSS[11], and SPARQL[13].

During playback, VUE will need to exchange authentication information with Sakai as different users view the VUE networks. This will likely be done with a combination of existing Sakai web services and ability to serve HTML and resources.

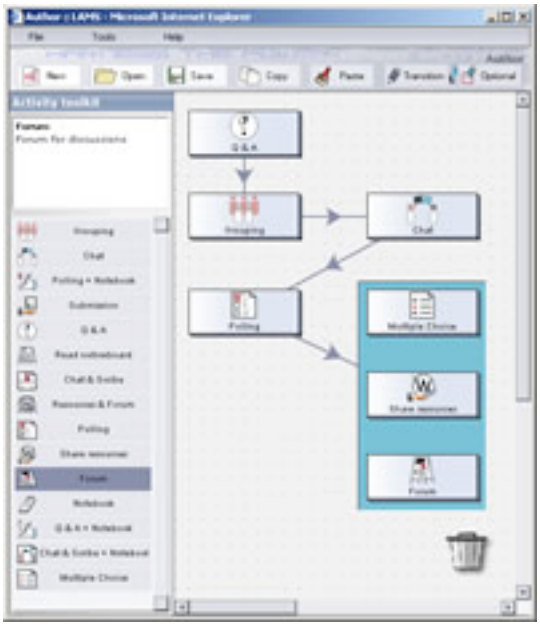
VUE and Sakai will continue to share approaches to Digital Repository integration using the Digital Repository OSIDs so that Sakai and VUE jointly benefit from advances made in this area.

Authoring Learning Design Sequences

Learning Design (LD) is an important teaching pattern that deals with organizing learning activities so that groups of learners can go through the activities together in a sequence controlled by the instructor.

A leading application in this space is the Learning Activity Management System (LAMS)[7]. LAMS allows an instructor to design sequences of activities which are in a precise order and where the instructor controls all aspects of the learner's transition from one activity to another. The instructor can track student activity in real time and alter the learning paths as the learning activity progresses.

A screen shot of the LAMS authoring environment is shown below:



LAMS is already integrated into Sakai at a very simple level[8]. The current integration simply connects a LAMS server with a Sakai server with automatic exchange of identity, role, and authorization information between the servers.

This integration is only a placeholder for a far more rich integration between LAMS and Sakai. Late in 2005 and early in 2006, there were a series of meetings between LAMS architects and Sakai architects (http://www.dr-chuck.com/media.php?id=59) to plan this integration.

Since there is really no clear standard that describes the interaction between a Learning Design authoring environment/runtime engine and a learning management system, LAMS has taken upon itself to build such a specification. The current draft of the LAMS specification is available at http://wiki.lamsfoundation.org/display/lams/Home - this will evolve over time but is a very strong start. The specification covers a wide range of aspects of the interaction from authoring and placement, to making printouts, controlling sequencing, etc.

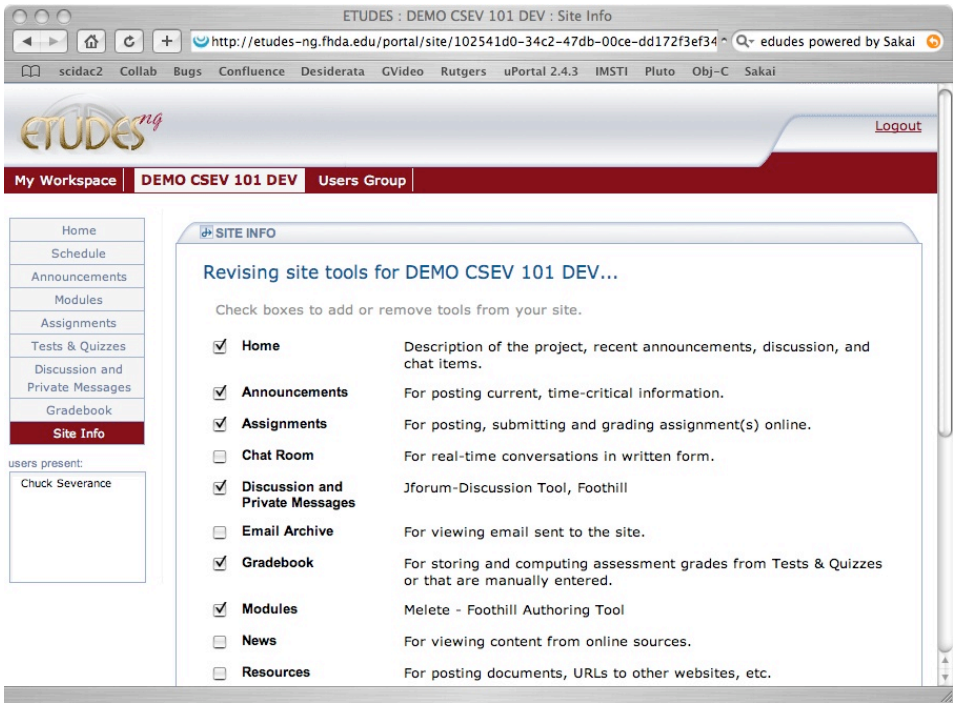
The LAMS 2.0 design effort greatly benefits form the experience of building a monolithic application (LAMS 1.0) and then re-factoring the code across a well-defined interface.

Going forward to produce a rich integration between Sakai and LAMS will require effort both on the part of the LAMS team and the Sakai team. First LAMS will refactor its authoring and run-time engine to comply with their new specification. Then LAMS will refactor their internal tools (quiz, discussion, etc) to comply with the new specification. Once these steps are completed the LAMS and Sakai

teams will work together to bring Sakai tools into the LAMS authoring and run-time environment.

Authoring within Sakai using Melete

Sakai currently only provides a very simple ability to allow an instructor to configure how their course site is laid out. The following is a shot of the current Sakai tool for site setup from the Etudes NG system:



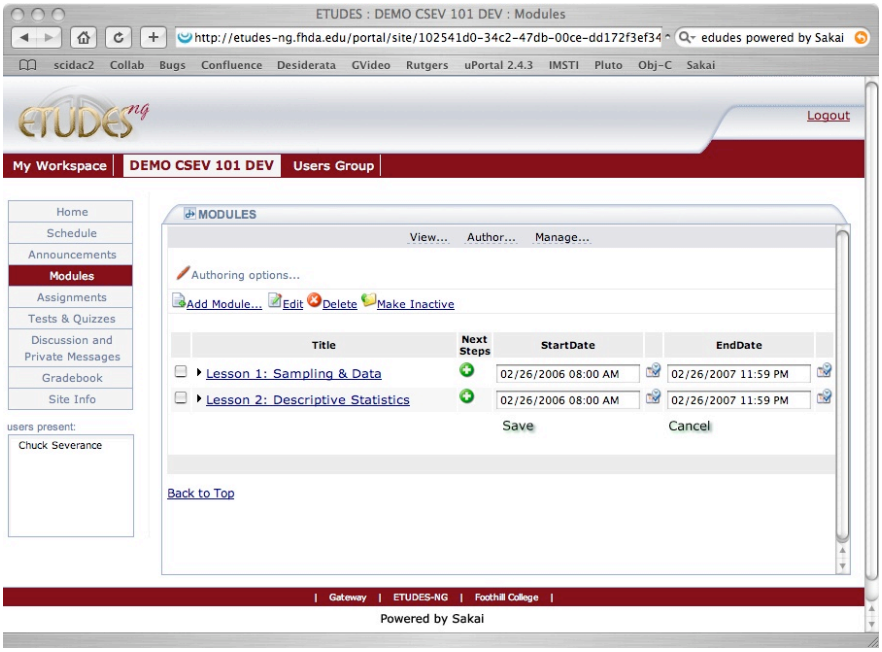
Many users find Sakai's simplicity an advantage in terms of less technical support required for large installations. A simple site setup tool works very well for basic use of a system in a learning environment and very well for ad hoc and project-style collaborations. For the light user the simple interface is ideal. But at the same time, there are instructors and instructional designers who want far more power in terms of site layout than Sakai provides.

While this tool is very easy to use with little or no training, there is a set of very important features needed to enable more sophisticated use of the system:

- The ability to have elements with release times
- The ability to organize and order the buttons in a course
- The ability to have a hierarchy of instructional objects (not just a flat list)
- The ability to have triggers between elements - where the successful completion of one element enables the use of another element
- The ability to place interactive elements anywhere in the course structure's

hierarchy

Sakai already has a tool that provides some of the needed capabilities. Melete was developed by Foothill College to fill this gap for their users. Melete currently operates as a single tool (not the entire site) and needs additional development to meet all of the user requirements in this area. The following is a screen shot of the Melete authoring interface:



Melete supports the ordering and timed release of elements but none of the other features. The green "plus sign" is the placeholder in the user interface where users can specify triggers and pre-requisites for modules. Notice that Melete operates within Sakai as a tool and is one of several buttons that Sakai provides.

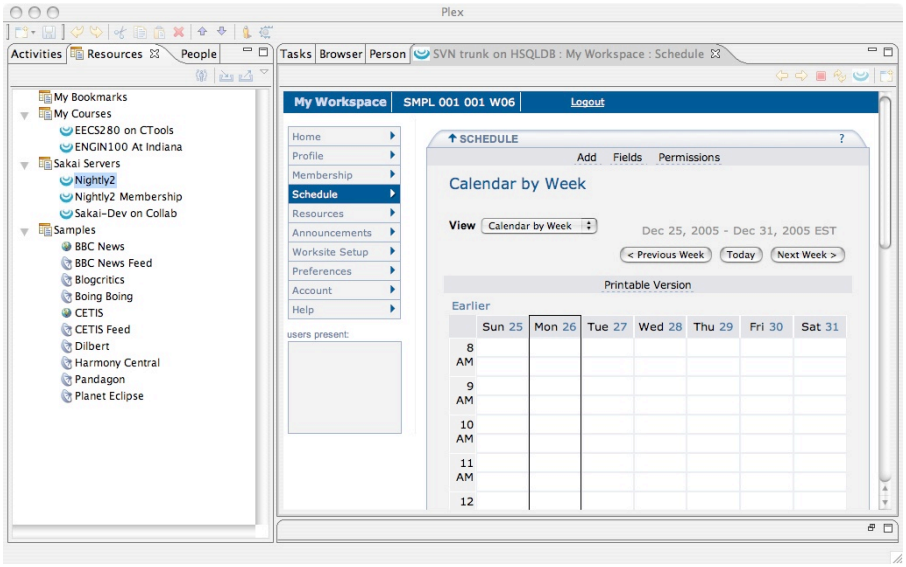
Foothill College is already working on the next generation of Melete with mentorship from the Sakai team at the University of Michigan. All of the mentioned requirements are on their task list. However, the last two requirements (triggers and flexible placement) requires the Sakai object bus (see Architecture discussion below) to be well supported to allow Melete to find, place, execute, and interact with the other elements of Sakai such as Samigo and the Discussion Tool. If we do not produce the object bus in a timely fashion, Melete will not be able to reach its full potential and meet the needs of the user community.

The demands from Melete will help form the requirements of the Sakai object bus and at the same time continuously validate the implementation of the object bus. Melete will mark the first instance of a new category of Sakai tools called "Resource Organizing Tools". Part of the effort in developing the Sakai Object

Bus will be to work closely with the Melete team to define and support this new type of Sakai tool.

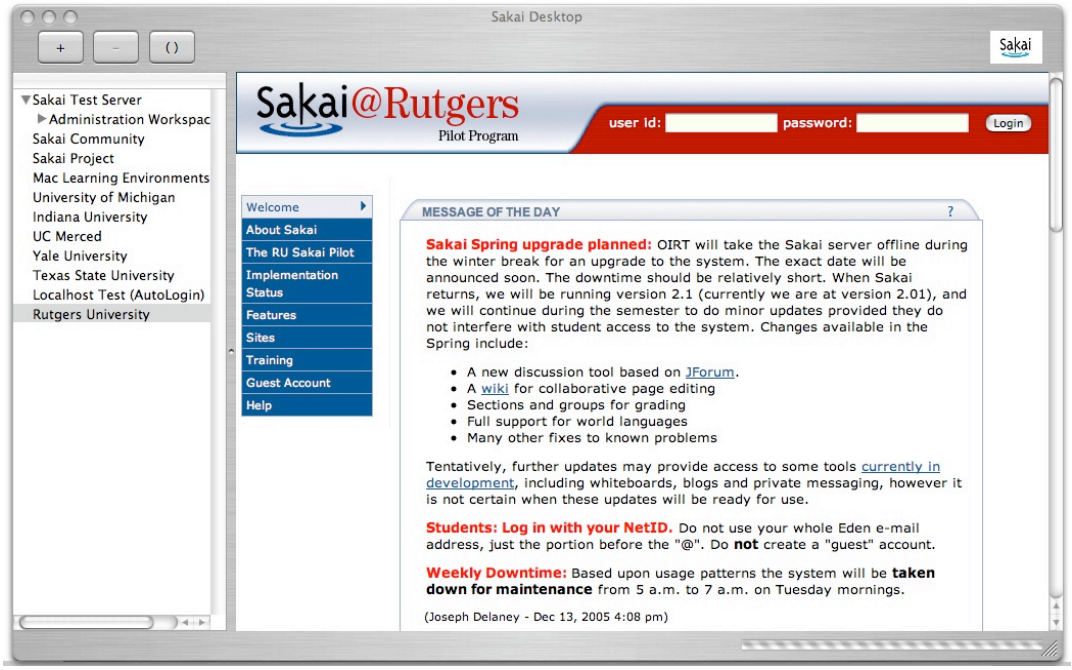
Personal Learning Environment

Personal learning environments such as PLEX[14] intend to produce an end-user experience similar to iTunes. The goal is to allow users to organize their collaboration and learning activities as they see fit rather than only relying on the organizational structure provided by their institution.



As users move beyond institutional boundaries and as users blend their teaching, learning, collaboration, and research it will be increasingly important to give users the ability to organize their own collections of materials across Sakai (and other servers).

The following is a simple Sakai Desktop application for Apple systems that begins to explore these areas:



These personal learning environments will be prototypes for some time but we feel that they have the potential to revolutionize how teaching and learning is done. We will not likely need to add anything special to Sakai specifically for personal learning environments. These environments will naturally make use of the features and capabilities that are built to interact with the other types of applications Sakai will work with.

The protocols that we expect to be most important to personal learning environments include: SOAP, iCal, RSS, HTML, and SPARQL.

New Sakai Capabilities

Semantic Services

In addition to the capabilities that allow Sakai to interoperate with other applications, we need to build a set of new capabilities within Sakai that will lay the groundwork for further data portability and interoperability.

There are a number of tools currently under development that need solid Semantic services available within Sakai. These tools include a Citation tool being developed as part of the SakaiBrary service and a learning outcomes tool being developed at Northwestern. Both of these tools are in the early phases of their development and are exploring their needs by implementing simple services that meet the needs of their particular tools.

Allowing these efforts to proceed independently is an excellent approach in the short term because these efforts are still at the early stages of exploring the requirements of Semantic tools in Sakai.

In time, as these efforts mature, Sakai will need to bring these efforts together and develop and release a solid set of general Semantic services inside of Sakai that can be used across a wide range of tools.

Off-line Sakai

Once Sakai objects are well modeled, and those objects are available in a serialized fashion, and it is possible to dynamically maintain synchronization between Sakai data elements stored in Sakai and data elements stored in a repository such as DSpace or Fedora, it is not a terribly great step to begin the design for off-line use of Sakai.

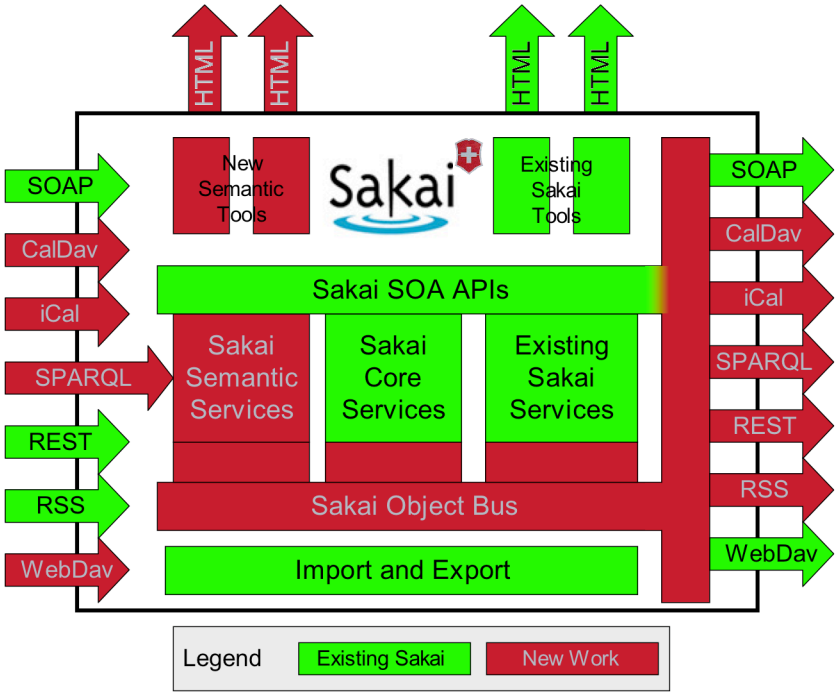
The powerful and desirable use case would be to pull your entire Sakai site onto your laptop complete with all of the assignments, files, etc. Then you could get on a plane and grade the assignments. At the end of the plane flight you could re-connect to a network and "synchronize" your off-line Sakai with the on-line Sakai. An initial vision for this work is presented in [19].

This is very common in productivity applications and is supported in Blackboard. This is a highly desirable feature and will become feasible as more of Sakai's internal data is well modeled. Once the basic infrastructure described in this document is complete, we could begin early phases of design and prototype work for an offline Sakai.

Architectural Approach

This is a large and dynamic effort. Because we will be driving towards a large number of successful integrations across the enterprise, we will need a clear architectural approach to make sure that all the data interoperability efforts use a common capabilities wherever feasible.

The following (rather busy) figure summarizes how to approach the problem to maximize reuse and make sure that we build common infrastructure and then make use of the common infrastructure in adding our data interoperability features.



The Sakai Object Bus is the technical component that enables much of the activity described in this paper. Currently Sakai already has a basic object bus (Sakai Entity Model) but this needs significant enhancement. The current Entity bus is aimed at exchanging objects within Sakai between Sakai Services operating in one instance of Sakai. To fully realize true data interoperability, the object bus will need significant enhancement to make sure that all objects are fully serializable and that the serialization is very generic.

A key is that serialization must not be part of the bus itself. Imagine a calendar item in the Sakai Calendar service - to render this for RSS requires one serialization, while rendering it in RDF responding to a SPARQL query requires a very different serialization. So the object bus must provide objects that can be serialized in many ways. We expect that the medium of exchange across the object bus will be Java objects and sets of properties. This allows the object bus to be used both with Sakai (between services) and between Sakai and an external consuming applications.

For each of the new protocols Sakai will support for this effort, we will need to implement the protocol and map between the needs of the protocol and the Sakai object bus and Sakai APIs. It is important to note that some of these protocols are read-only and others are read-write. As an example WebDav and CalDav are both read-write protocols. Currently to support read-write protocols, the WebDav implementation uses the Sakai APIs rather than the Sakai Entity bus. One of the challenges of this effort will be to add the capability to support read-write protocols across the Object bus (i.e. *without* using the Sakai APIs). Today

the Entity bus supports read-write activity but primarily focused on the off-line import and export of entire Sakai sites. As part of this effort, we will add on-line read-write capabilities to these objects in a generic way. Once this is completed, we will refactor the Sakai import and export processes to use this more general capability.

Another important aspect of this project is the Sakai Semantic services. As Sakai increasingly begins to consume data from external applications, we will need to have ways to store this information and then augment that information in value added ways - this allows Sakai to become as much a consumer of these protocols as a producer of the protocols. Sakai already consumes several protocols (RSS, SOAP, and REST) - the primary addition for this project will be consuming RDF using the getData and SPARQL protocols. Often support for consuming these protocols will be very tool-based - someone will be writing a new tool (such as an RSS viewer) and will need to build a new service that consumes external information.

Summary

The work described here represents a long-term investment in Sakai's ability to be a strong player in an increasingly multi-application enterprise environment. By enabling Sakai to "open its data" to all applications with an interest in Sakai's data, we hope to develop a pattern that will be an example for other enterprise class applications to open their data as well using these techniques.

We hope that we will set a strong example for commercial ad non-commercial applications across the enterprise. We hope this example encourages them to share their data and capabilities in increasingly portable ways. By implementing the features in this document, we set a rather high bar in terms of data interoperability. To reach these levels, we need a substantial investment in Sakai's infrastructure and quickly.

While there is very little in this document that directly affects the end-user feature set in the existing Sakai tools, we need to start quickly on improving these infrastructure issues so that we can get the necessary framework in place as soon as possible so that new tools that are developed work in the improved environment.

References

[1] http://www.uportal.org/

[2] http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsrp

[3] http://www.jcp.org/en/jsr/detail?id=286

[4] http://www.imsglobal.org/

[5] http://www.jcp.org/en/jsr/detail?id=170

[6] http://loncapa.msu.edu/

[7] http://lamsfoundation.org/

[8] http://lamsfoundation.org/integration/sakai2/

[9] Design Patterns and Business Models for the Next Generation of Software, Tim O'Reilly, 09/30/2005, http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html

[10] Roger L. Costello, Building Web Services the REST way, http://www.xfront.com/REST-Web-Services.html

[11] RSS Tutorial, http://www.mnot.net/rss/tutorial/

[12] RDF Tutorial, http://www.w3.org/TR/rdf-primer/

[13] http://www.w3.org/TR/rdf-sparql-query/

[14] http://www.cetis.ac.uk/members/ple

[15] http://www.oasis-open.org/committees/wsrp/

[16] http://bugs.sakaiproject.org/confluence/download/attachments/17504/nomadic-sakai.pdf?version=1