Integrated Collaborative Environment (ICE)
for Teaching, Learning, Research & Work

White Paper & Project Proposal

Educational Partnerships & Learning Technologies
Computing & Communications

DRAFT
April, 2004

Table of Contents

1 Introduction

The Uniform Access computing environment provided to the UW community by Computing & Communications (C&C) has proven itself over the years as a remarkably robust and versatile setting for accomplishing a wide variety of computing tasks. Built upon this infrastructure, the Catalyst Web Tools, provided by Educational Partnerships & Learning Technologies (EPLT), have served the communication and collaboration needs of UW faculty, staff, researchers, and students since 1999. The UA clusters are used by tens of thousands of people on a regular basis for email, Web publishing, streaming media, file storage, and other computational tasks. Catalyst Tools use continues to climb. In 2003, 8,200 people set up almost 19,000 tools for use by others, students built over 12,000 new online portfolios, and total online learning activities doubled to 18 million. Almost every member of the UW community accesses these tools and resources through the MyUW portal.

Against this backdrop of pervasive technology use, however, many faculty, staff, researchers, and students are asking for a more integrated and personalized means to interact with the technology tools and resources. Currently, these Web-based resources are scattered among multiple sites with different user interfaces, making it difficult for users to know all the resources available to them and to coordinate their use. In many respects, these tools are underutilized because of this lack of integration.

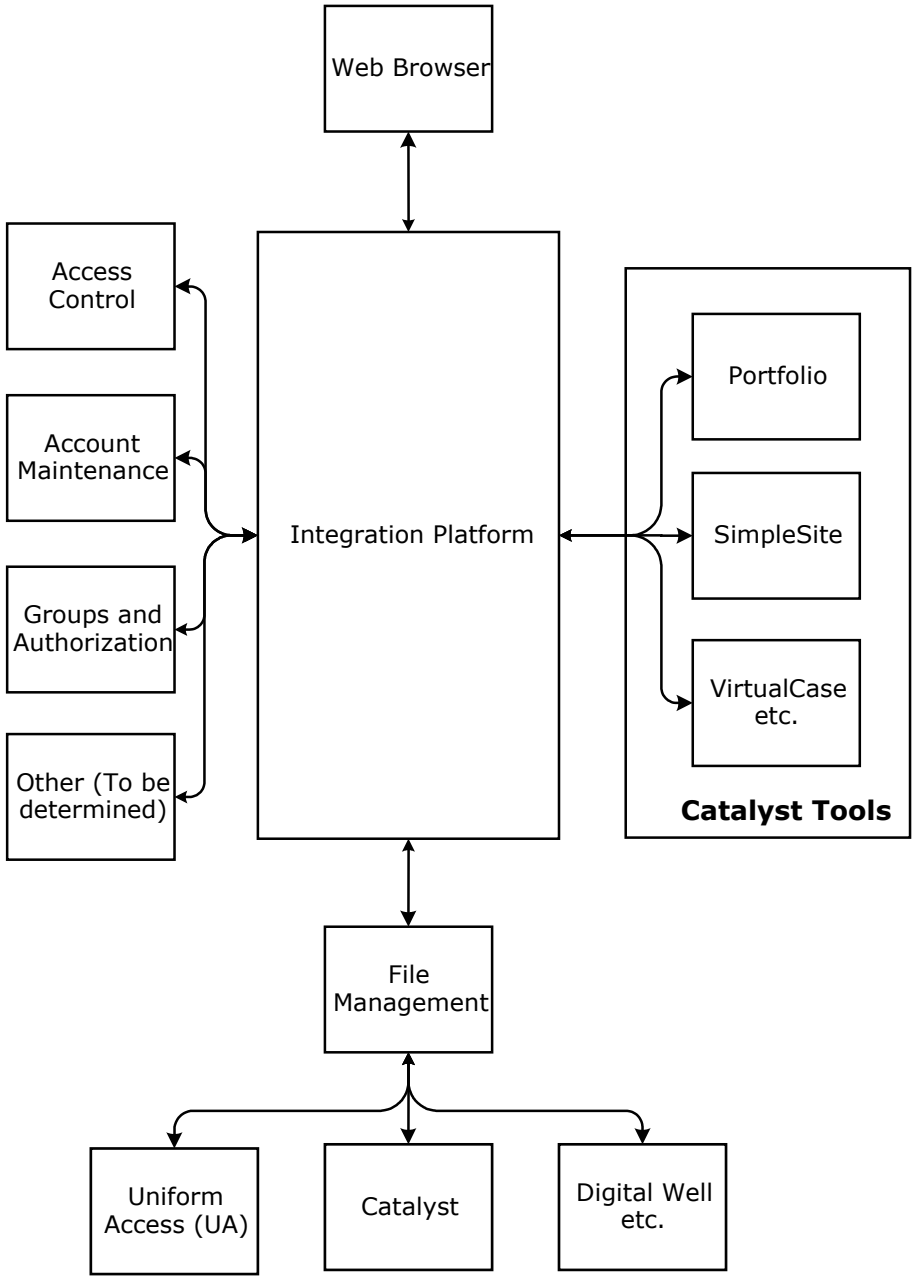
To extend the use of these computing resources further into the everyday teaching, learning, research, and work activities of the university, we propose the ***Integrated Collaborative Environment*** (ICE). ICE will incorporate Catalyst Web Tools, file management, Web publishing, access control, group management, and account management within a single, browser-based "workspace" with a common look and feel. ICE will take advantage of the current computing and networking environment at the UW, extend and refine the Catalyst Web Tools, and leverage national open source development efforts by partner universities and consortia to create a powerful online platform for UW faculty, students, researchers, and staff.

ICE will provide numerous benefits.

- Teaching: whether for campus based courses or online distance learning programs, ICE will make it easier to manage courses and conduct online pedagogical activities
- Learning: students will be able to use the communication and collaboration tools typically used only by faculty to form study groups and to structure their learning outside the classroom.
- Research: ICE will provide a means for any researcher to easily conduct surveys, organize and share data and information, and communicate with others about their work.
- Everyday Work: anyone with a UW NetID will be able to use the tools and resources within ICE to support their daily work activities.

To bring all of this about, an application integration platform (portal) must be developed or obtained to allow all the tools to integrate and function seamlessly. Once this platform is in place, significant work will be required on existing tools to fit them within the new platform. And finally, new tools must be developed to handle functionality that does not yet exist. The following diagram illustrates some of the components of ICE:

Integrated Collaborative Environment



2 Rationale for ICE

Current users of the UW computing infrastructure, resources, and tools often must go to several different locations to accomplish their goals, opening different Web browser windows, switching between applications, and moving amongst different accounts. Though usage is high, there are indications that usage would extend farther into the everyday teaching, learning, research, and work activities of the UW community if the experience were seamless and transparent, with an easy-to-use interface and a common look-and-feel. For example:

- Though over 100,890 UW NetIDs are eligible for Web publishing, only 18,450 have actually published content to the Web. The ICE framework will simplify Web account creation and publication so that users can effortlessly create Web pages for a variety of activities.

- In a recent survey of over 400 faculty, staff, and students, 79% of the respondents felt it somewhat, fairly, or very important to create a Web site, but only 8% reported no difficulty in doing so. Figuring out how the UW Web servers work, designing a Web site, and transferring files to the Web site were the top three difficulties. EPLT software developers will build SimpleSite 2.0 to fit within the ICE framework so that users will have access to a variety of Web templates and styles and can easily publish Web pages without having to know how UW Web servers work.

- In another recent survey of almost 300 faculty, staff, and students on electronic file management, 69% of the respondents reported moving files between computers every day or a few times a week, and 65% do this via email attachments which can be cumbersome. 83% find it necessary or very necessary to share files with others at UW, and 66% with others outside UW. Web-based access to files and access via a secure, encrypted connection is necessary or very necessary to 75% and 84%, respectively. The ICE framework will underpin development of a secure, Web-based file management system to meet these needs.

- At a recent retreat on digital scholarship hosted by the UW Libraries, many UW faculty who are deeply engaged in the creation of knowledge using digital means expressed a need for more "Catalyst-like" tools to support digital scholarship and collaboration. The ICE framework and portal technologies will make Catalyst Tools and other resources available in a Web-based "workspace" that lets faculty, staff, and students create customized, online collaborative spaces that can be shared with others both within and outside the UW community.

- Current users of Catalyst Tools frequently ask that the "workspace" user interface seen in Portfolio and Group Manger be extended to the other eight tools. Users also would like a "wrapper" that integrates the individual Catalyst Tools, making it easier to create them and link them to Web pages, whether for courses or for personal use. The portal technologies of ICE will allow EPLT software developers to achieve this level of integration, thus saving users from

opening several browser windows or cycling through multiple screens to accomplish their work.

In short, ICE will reduce these obstacles by building upon the strengths of the existing computing and networking environment at the UW, which is characterized by (1) distributed, powerful desktop computing, (2) widespread, increasingly fast networks, (3) pervasive UW NetID infrastructure, (4) universal access to the Web, and (5) ubiquitous communication and collaboration tools, including the Catalyst Web tools. Any UW user will access ICE through a Web browser where activities such as file management, Web publishing, access control, group management, account management, and the configuration of Catalyst Tools and other resources can all be accomplished within a single "workspace".

There is an active community working on open core technology standards and open source Web portals that will facilitate integration of disparate tools, content and resources. ICE will leverage these national open source development efforts by partner universities and consortia, thus bringing this value back into the UW community. The Open Knowledge Initiative will inform and in some cases furnish key pieces of the middleware architecture. The portal technologies will be shaped by the uPortal Consortium and recent developments in the Java Community Process. The Sakai project and work by the University of Michigan, Indiana University, MIT, Stanford, and nineteen other partner universities will influence the ICE framework and may lead to the incorporation of outside tools within ICE. Collaborative projects with Harvard and research and development with UCLA and UC Berkeley will produce Web Services for Remote Portlets that will help stitch together disparate tools and resources within ICE. Shibboleth middleware, an Internet 2 project, will facilitate inter-institutional sharing of Web resources subject to access controls.

3 Candidate Integration Platforms Overview

This section discusses the relevant standards and the current leaders in the open source Web portal arena.

3.1 Open Core Technology Standards

3.1.1 Portlet API (JSR 168)

Small units of Portal functionality are referred to as "portlets." JSR 168 is a recently ratified standard that specifies the API for portlets. Detailed information on JSR 168 can be found at http://jcp.org/en/jsr/detail?id=168.

The Apache Jakarta Project has developed a reference implementation of JSR 168 called Pluto. Pluto provides a portlet container in which portlets that adhere to JSR 168 can execute. Refer to http://jakarta.apache.org/pluto/ for more information.

The Portlet API is a key element of future versions of uPortal, as discussed below.

3.1.2 OKI/OSID

MIT's Open Knowledge Initiative (OKI) is developing a set of service-specific APIs called Open Service Interface Definitions (OSIDs) that cover the complete data and operations for common services such as authentication, authorization, filing, logging, hierarchy, and digital repository. The definitions separate what needs to be done from how it is done, which helps to reduce system complexity and enable integration and interoperability. Refer to http://web.mit.edu/oki/ and http://sourceforge.net/projects/okiproject for more information.

By adhering to an OSID, a system achieves an important level of abstraction and encapsulation for each service. Thus, when the behind-the-scenes implementation for a particular service changes, the invariant OSID API helps prevent other elements of the complex system from being affected.

3.1.3 Web Services for Remote Portlets (WSRP)

WSRP is a Web services standard that allows portals, other intermediary Web applications that aggregate content, and applications from disparate sources to "plug-n-play" or interoperate. It has recently been approved as an OASIS standard. It is built on top of SOAP (Simple Object Access Protocol), an XML protocol for sending and receiving messages via the Web, and WSDL (Web Service Description Language), an XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information. Both SOAP and WSDL are W3C Recommendations. WSRP is intended to be compatible with JSR 168.

Refer to http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsrp for more information.

3.2 Open-source Portal Technologies

3.2.1 uPortal

uPortal is an open source Java, J2EE, JSP, XML and XSLT portal framework. It is an active project of the Java in Administration Special Interest Group (JA-SIG) and more specifically the uPortal Consortium. A good summary of the uPortal project can be found at http://uportal.org. This site also provides many additional helpful links, including one to download the latest uPortal distribution. For a good guide to using uPortal, visit http://www.fair-portal.hull.ac.uk/downloads/uPortalGuide1.pdf.

uPortal is currently in version 2.2 and supports WSRP. Channels (implementing the IChannel interface) are the primary way to incorporate code-level content and tools into the portal in this version.

Version 2.3 is under active development and is scheduled to be released soon. Version 2.3 will incorporate Pluto, Apache's reference implementation of the portlet container for JSR 168. In this version, channels (implementing the IChannel interface) will still be the primary way to incorporate code-level content and tools into the portal. Portlet support will be provided by the Pluto portlet container but will be a second class citizen.

Version 3.0 completes the work begun in version 2.3. It is currently in development (incidentally requiring the majority of the code to be rewritten). Portlets will become the primary way to incorporate code-level content and tools into the portal. Channels (implementing the IChannel interface) will be provided for backwards compatibility.

Work on versions 2.3 and 3.0 is being driven by the Sakai Project (see below), of which the uPortal Consortium is a key member.

3.2.2 CHEF

The Comprehensive Collaborative Framework (CHEF) is an open source Java, Servlet-based portal developed by the University of Michigan. A project overview and other useful information can be found at http://www.chefproject.org. It is intended to provide a "flexible environment for supporting distance learning and collaborative work." Tools need a framework to provide a consistent way for invoking other tools and passing information among them. For example, a homework drop box tool developed at one university may need to invoke and pass a grade to a grade book tool developed at another. This framework provides a common place for these tools to interact with each other in a standard way. It also provides services like notification that cross tool boundaries. CHEF has developed such a portal-based framework that provides the environment for a large set of course management tools.

The University of Michigan is refactoring and rewriting elements of CHEF in order to incorporate them into the Sakai Project. The tool interoperability framework portion of CHEF will be a core piece of Sakai.

3.2.3 Sakai

http://www.sakaiproject.org/
Sakai is a collaborative effort between the University of Michigan, Indiana University, MIT, Stanford University and the JA-SIG uPortal Consortium. The goal of the project is to "join forces to integrate and synchronize the considerable educational software [of the partners] into a pre-integrated collection of open-source tools." Sakai will be built on uPortal 3 (JSR 168 compliant), OKI/OSID and CHEF's tool interoperability framework. Nineteen other universities, including the UW, participate in the development of Sakai, influencing its direction and contributing sharable applications that work with the Sakai framework.

4 Integration Platform Recommendation

This is an exciting, dynamic time in the open source Web portal space and associated technologies. The Sakai Project is moving ahead at breakneck speed and is poised to provide a solid, useful solution. However, as always, it is prudent to be conservative with unproven technologies and try not to be an early adopter. Fortunately, the UW has a robust set of communication and collaboration tools in place with Catalyst Web Tools and can take a staged, evolutionary approach to the integration platform:

- *Stage I:* Use uPortal 2.2 as the initial integration platform. Retrofit a set of existing applications with WSRP compliant Web services, or rewrite them in Java as channels, and integrate them.
- *Stage II:* Move to uPortal 2.3 when available/proven.
- *Stage III:* Write Java OSIDs and Java tools that use the Portlet API and integrate them. Provide WSRP compliant Web service interfaces to the tools where desired.
- *Stage IV:* Move to Sakai when available/proven.
- *Stage V:* Port existing non-Java tools to Java/Portlet API/OSIDs/WSRP when/where it makes sense.

This approach, if adopted, could have a lot of potential benefits, including:

- Incorporate open source technologies, tools, and applications into the UW framework when it makes sense.
- Manage risk by allowing other community members to be the first adopters of a given technology before we move to it.
- Minimize "throw away" code because uPortal versions 2.2 and 2.3 are already planned to be forward compatible, as are WSRP 1.0, 1.1 and 2.0.
- Utilize and improve existing UW tools and applications, which can be modified over time to fit in the new framework.
- Avoid an "all or nothing" approach, meaning we can pick and choose from the technologies and phases without committing to a full integration while remaining able to easily react to change.

5 Candidate Tools for Integration within ICE

1) Catalyst Tools
   a) Portfolio
   b) SimpleSite
   c) VirtualCase
   d) EPost
   e) E-submit
   f) Peer Review
   g) QuickPoll
   h) UMail
   i) WebQ
   j) Group Manager

2) File Management Resources
   a) File management tools (list, create, move, rename, delete) for the user's UA unix file systems, web publishing directories, email folders, and streaming media files
   b) File viewing for supplemental accounts user owns
   c) Digital Well tools

3) Directory information

Teaching-specific customization

1) Teaching Class Schedule
2) Class information management
   a) Catalog description
   b) Instructor class description
   c) Class Web site address
3) Online course resources
   a) Electronic class list management
   b) Class Web site management
   c) Course materials management

4) Web publishing management and tools
   a) File upload and transfer
   b) Streaming media management

5) Supplemental Account resources and management
   a) List of "owned" supplemental accounts
   b) Email
   c) Web

6) Uniform Access computing resources
   a) UW NetID management
   b) Email forwarding
   c) Password management
   d) Add or change services
   e) Request other services

7) Potential: "Project management" tools
   a) Project description
   b) Project Web site management
   c) Discussion group management
   d) Project team/mailing list management
   e) Project materials management

8) Email

9) Calendar

10) Mailman (listproc) manager

6 Relationship with MyUW

MyUW is the UW's enterprise portal, providing access to a wide range of information and applications across the breadth of the entire institution. Parts of ICE's web-based workspace will be shown as channels within MyUW, most likely on the Teaching and Student tabs. These channels will also act as means of navigating to the ICE workspace. It is possible that ICE may also act as a platform for piloting specific technology directions or software protocols that may later be incorporated within future iterations of MyUW.

7 Project Approach

7.1 Phase I

1. Choose an integration platform, which includes thoroughly investigating the viability of the staged approach discussed previously.
2. Identify tools/apps currently at the UW that will be integrated.
3. Determine and assemble a steering committee and project team.

4. Produce a design document/functional specification detailing the resources to be integrated into the first version of ICE and how it will be done.
5. Produce a project plan that includes resource requirements and a timeline for achieving the first integration iteration.

7.2 Phase II

1. Develop a complete application HTML mock-up.
2. Implement the Phase I plan.
3. Make recommendations for future phases.

7.3 Phase III

1. Produce a method/API specification for the common look and feel as well as instructions for making the resources available to the common workspace.
2. Identify additional UW tools and applications currently and integrate them.

7.4 Future work

Future phases may consist of integrating additional tools and applications.

8 Resources Required

8.1 Hardware

8.1.1 Development Environment

A shared machine with the operating system and software discussed below.

8.1.2 Production Web Servers

Two redundant, load-balanced machines running Linux (flavor TBD).

8.1.3 Production Database Servers

Two machines, one primary and one secondary, running MySQL, SQL Server, or Oracle (TBD).

8.1.4 UW Infrastructure

WebLogin servers
24 x 7 operations support and monitoring
Network
Load balancer
Catalyst Web servers
Catalyst file servers
Uniform Access resources

8.2 Software

8.2.1 Open Source (free)

Java/J2EE and JDBC drivers
Perl 5
uPortal 2.2, initially, then versions 2.3 and 3.0 for later stages of the project
Apache Tomcat servlet container

Apache Pluto (for portlets)
Apache Java Server Faces (JSF)
WSRP implementation
MySQL database (TBD)

8.2.2 Commercial (buy)

SQL Server or Oracle database (TBD)
Java IDE for Linux-based Java development

8.2.3 UW Infrastructure

Pubcookie/Shibboleth
Catalyst applications
Uniform Access applications
Process and tools for code releases to production system

8.3 Staffing

C&C and EPLT staff will implement an integration platform and portal technologies and then weave existing Catalyst Web Tools and other Web-based resources into an easy-to-use package where all the work happens within a single Web browser. The following list describes staffing requirements:

8.3.1 Advisory Committee

Role:

- Guide the project direction and review implementation.
- Sounding board for project ideas and directions.

Level: As required

8.3.2 Project Manager

Role:

- Manage and run all aspects of the project.
- Work with all project staffing groups and individuals.
- Track and monitor project progress.

Level: .5 FTE

8.3.3 Technical Architect

Role:

- Design the software/development environment for ICE.
- Work with the application engineers.

Level: .5 FTE

8.3.4 Application Engineers

Role:

- Design, code, and implement the ICE software environment.
- Design, code, and implement the applications that are accessible from ICE.
- Provide on-going application support.

Level: 2-3 FTE (on-going)

8.3.5 User Interface (UI) Designer

Role:

- Design user interface for ICE environment and applications.
- Work with Usability Specialist to conduct usability studies on the UI.

Level: .5 FTE

8.3.6 Usability Specialist

Role:

- Coordinate, plan and conduct usability studies on ICE.
- Work with User Interface Designer to conduct usability studies on the UI.
- Get direction on what applications to integrate into ICE.

Level: .5 FTE

8.3.7 Technical Writer

Role:

- Provide portal content and write help pages.

Level: .25 FTE

8.3.8 System Engineers

Role:

- System administrators
- DBAs
- Network engineers
- Operations

Level: 1-2 FTE

8.3.9 USER-type groups

Role:

- Provide input on UI, application functionality and campus needs.
- Direct the applications to be included in ICE.

Level: As required.

9 Estimated Costs

We estimate the one-time hardware and software costs at $150,000. One-time costs for the staff needed to build ICE are estimated to range from 5.25-7.25 FTE at $590,000-$850,000. Ongoing costs for hardware and software together with the 2.5 FTE needed for application support, database administration, and backup services are estimated to range from $280,000-$350,000.

10 Assumptions

1. Integration will be accomplished with an existing application integration platform, preferably one using open source technologies.
2. Catalyst, MyUW, Digital Well, ITI, Client Services and other dependent groups and projects will have resources to support the project.
3. The scope for ICE includes integration of commonly used Catalyst Web Tools, Uniform Access computing resources, supplemental account resources, and file management resources. Integration of administrative applications and

data warehousing and other UW administrative work products is specifically outside the scope of this project.
4. The choice of development and production environment will be dictated primarily by 1) the chosen application integration platform (portal) and 2) existing UW projects and tools.
5. Depending on the choice of development language, resources will be made available for necessary staff training.

11 Issues and questions

The main outstanding issues and questions are as follows:

1. How many tools will be integrated?
2. How are applications that are integrated into ICE used standalone ("a la carte"), if at all?
3. What will our development language of choice be? Perl? Java? Other?
4. For existing applications (e.g. Catalyst), it would be beneficial to retrofit them to use WSRP. Is there a Perl implementation of WSRP that we could use to do this?