International Journal of Computer Information Systems and Industrial Management Applications.
ISSN 2150-7988, Volume 5 (2013) pp. 714-721

Integration Mechanisms in e-learning Platforms

Francisco Huertas[1] and Antonio Navarro[2]

[1] DSIA, Facultad de Informática
Universidad Complutense de Madrid
C/ Prof. José García Santesmases, s/n. 28040, Madrid, España.
fhuertas@fdi.ucm.es

[2] DSIA, Facultad de Informática
Universidad Complutense de Madrid
C/ Prof. José García Santesmases, s/n. 28040, Madrid, España.
anavarro@fdi.ucm.es

***Abstract*: Nowadays, integration of e-learning platforms has become a key issue in e-learning. In order to facilitate this integration, most e-learning platforms depict their functionality in terms of APIs and/or web services. Usually, APIs expose the most important functions in platforms. However, the availability of web services in every platform is very heterogeneous. In addition, every platform follows its own philosophy when designing its services. This paper analyses three of the most successful e-learning platforms (Blackboard, Moodle and Sakai), identifying their APIs and web services, and comparing their readiness for the development of a virtual campus based on these services. The goal of the paper is to facilitate the integration of these platforms in an information technology infrastructure.**

***Keywords*:** platform integration; Blackboard, Moodle, Sakai**.**

I. Introduction

In recent years e-learning has had a significant impact in the educational context and it covers a wide set of applications and processes, such as Web-based learning, computer-based learning, virtual classrooms, and digital collaboration. It also includes the delivery of content via Internet, intranet/extranet (LAN/WAN), audio and videotape, satellite broadcasting, interactive TV, CD-ROM, and more [1].

e-learning's success has promoted the appearance of *virtual campuses*: "The virtual campus is a metaphor for the electronic teaching, learning and research environment created by the convergence of several relatively new technologies including, but not restricted to, the Internet, World Wide Web, computer-mediated communication, video conferencing, multimedia, groupware, video-on-demand, desktop publishing, intelligent tutoring systems, and virtual reality [2]. In more recent studies [3, 4, 5, 6] virtual campuses are understood, in a broader sense, as the integration of Information and Communication Technologies in universities at both educational and organizational levels.

Originally, virtual campuses were built on a single e-learning platform, or *Learning Management System* (LMS). However, at present, virtual campuses are evolving towards complex applications built on several e-learning platforms that have to be integrated [7, 8].

In this context, e-learning platforms have evolved in order to facilitate their integration with other applications. This evolution has two different approaches: (i) the inclusion of *Application Program Interfaces* (APIs) to make public the functionalities of the e-learning platform in terms of a code written in the same language in which the e-learning platform has been built; and (ii) web services that allows the integration of e-learning platforms with heterogeneous applications.

This paper, an extended version of [9], analyzes the need for integration of e-learning platforms, as well as the integration facilities provided by three of the most successful e-learning platforms in terms of their APIs and web services. Thus, Section 2 describes two projects that take advantage of the integration capabilities of e-learning platforms. Sections 3, 4 and 5 describe web services availability in Blackboard, Moodle and Sakai. Section 6 analyzes this availability, comparing web services functionalities with APIs functionalities. Finally, Section 7 presents conclusions and future work.

II. Need for integration of e-learning platforms

A. VCAA Project

The *Universidad Complutense de Madrid* (UCM) is an old university, founded in 1499, and is currently the largest non-open university in Spain. In the academic year 2010-2011 there were 83,700 students and 6,200 lecturers. In the academic year 2003-2004 the *UCM Virtual Campus* (UCM VC) [10] was set up. The main objective of the project was to place at students and lecturers' disposal all the support that modern information and communication technologies can provide to improve the quality of learning and research activity at the university [11]. The UCM Virtual Campus includes management of the students enrolled in courses and of the content of these courses, as well as facilitating cooperation and communication: work groups, chats, forums, etc. In the present 2011-12 academic year there are 81,000 active students and 4,000 lecturers in the Virtual Campus.

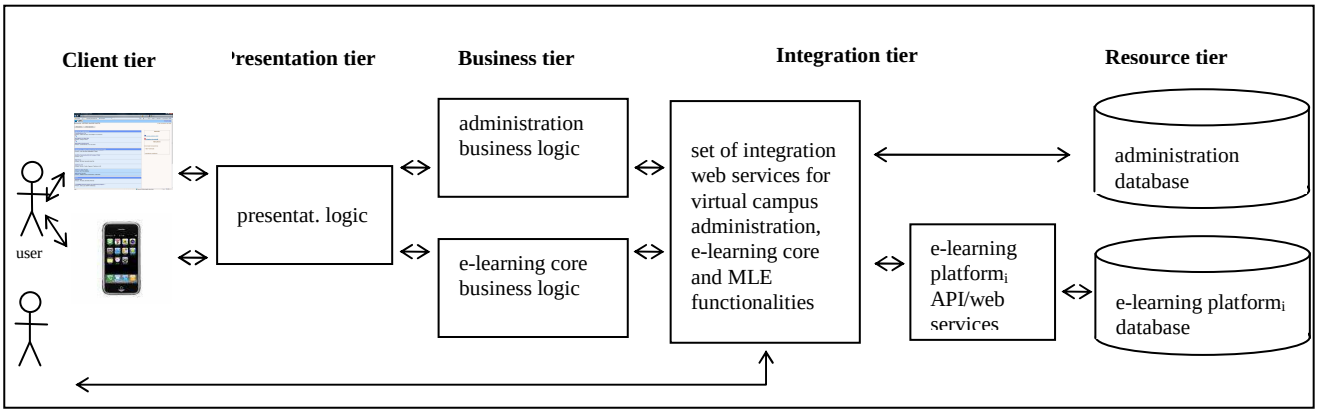

Figure 1. New integration architecture for virtual campuses [7]

Since its deployment, the UCM VC has had several software architectures for dealing with its e-learning and administrative facilities [7]. At present, the *Virtual Campus Advanced Architectures* (VCAA) project is designing new software architecture for virtual campuses based on *Service-Oriented Architecture* (SOA) [12].

According to this architecture, virtual campuses are built on an integration layer [13] described in terms of abstract interfaces. The e-learning platforms that implement these interfaces can be used to support core e-learning facilities and can be easily interchanged in these virtual campuses. Fig. 1 describes this architecture implementing the SOA architecture in terms of web services.

The first step for the development of these web services was the analysis of web services availability in e-learning platforms. Precisely, this paper describes such availability, as well as the availability of other integration devices such as APIs.

B. Campus Project and OKI

Due to the diversity of platforms and the differences among them, *Campus Project* [8] emerges as a developing community within the area of e-learning. Campus Project is focused on interoperability between systems, ensuring that developments are shared among its members.

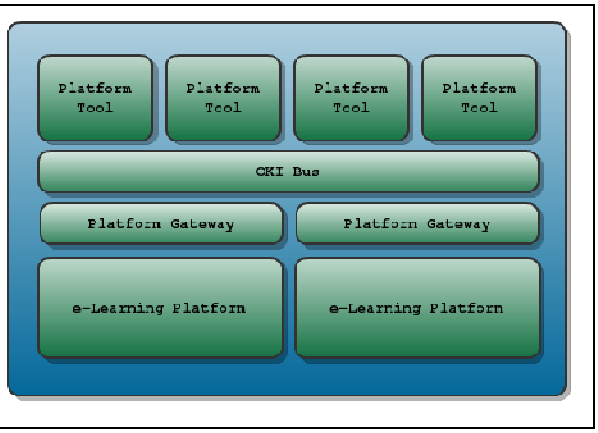

Figure 2. Architecture of Campus Project [14].

Campus Project is based on the assumption that the next step to achieving real interoperability is to adopt a SOA model. When these services implement a clearly-defined interface, it is possible to isolate the interaction mechanisms in a single layer, which provides control over the coupling between the two endpoints. If loose coupling is desired, the layer can be, for example, implemented using web services. Campus Project is this type of architecture (Fig. 2): heterogeneous tools developed in different programming languages that interact with a group of LMS services, but independently of the LMS. This type of architecture could be described as agnostic with respect to the learning tools and LMS used, and coincides with the vision of the VCAA Project. In order to implement this architecture, Campus Project adopts the *Open Knowledge Initiative* (OKI) proposed by *Massachusetts Institute of Technology* (MIT) [15].

Campus Project is based on OKI, an open and extensible architecture that specifies how the components of LMSs communicate with each other. OKI is specified in the Open *Service Interface Definitions* (OSID) [16], a programmatic interface that describes OKI services.

Although OSID does not aim to provide SOA solutions, the presence of web services in the e-learning platforms can facilitate the implementation of the OSID interfaces in the Campus Project. Therefore, the definition of web services availability in main e-learning platforms can also facilitate the development of the Campus Project.

III. Blackboard Learn's Web Services

Blackboard Learn 9.x [17] is one of the most important e-learning platforms. More than fifty percent of the academic institutions use it as the main LMS [18]. The platform offers many features, and new functionalities can be deployed using its tool called *Building Blocks* [19].

Blackboard's web services prioritize functionality over usability and this makes it different from other e-learning platforms. Thus, Blackboard has the most complete implementation of web services. However, these web services do not include all the functionalities of the e-learning platform deployed as a web application.

A. Protocols Supported

Because Blackboard does not prioritize usability, it only implements one web service protocol, *Simple Object Access Protocol* (SOAP) [20]. However, this implementation is enough to support all the implemented functionalities. Therefore, no more protocols are needed.

B. Architecture of web services

Blackboard implements architecture similar to Sakai. This architecture is depicted in Fig. 3.

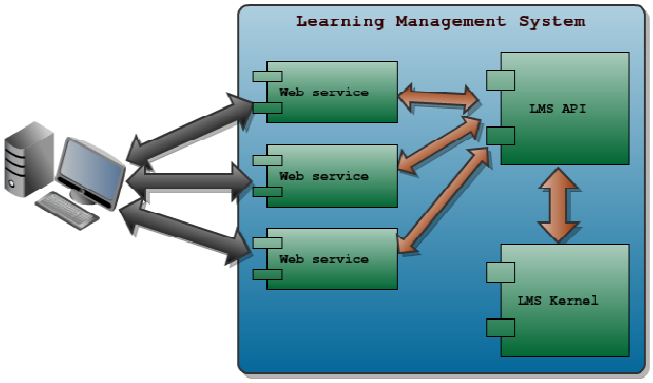

Figure 3. Architecture of web services in Blackboard and Sakai

Blackboard's services are grouped by functionality and resource. These services have basic login and management operations.

Blackboard's web services are:

- *Announcement*: This web service provides methods for creating, modifying and accessing announcements.
- *Calendar*: This web service provides methods for accessing and updating the calendar items in a calendar.
- *Content*: This web service provides methods for creating and accessing content items.
- *Context*: This web service provides the initial methods required for session creation. Therefore it needs to be invoked before any other web service can be used.
- *Course*: This web service interface provides methods for creating and accessing course items.
- *CourseMembership*: This web service provides methods relating to memberships of courses and groups.
- *Gradebook*: This web service provides methods for accessing grade books.
- *NotificationDistributorOperations*: This web service features web service methods for executing notification distributor operations in Blackboard Learn.
- *User*: This web service provides methods for accessing and updating the users, admin users and users' address book entries.
- *Util*: This web service provides secondary methods for accessing and updating global configuration.

C. Security level

To relate a session with web service layer, the LMS usually has a session identifier. This identifier identifies the user in the e-learning platform during a session. This is the principal unsafe point in most e-learning platforms, because if this session identifier is stolen, the user's session can be accessed by hackers.

The majority of LMSs do not implement security in the web service and end-users must implement security policies if needed. However, Blackboard can force the use of *Secure Sockets Layer* (SSL) [21] to access its web services.

IV. Web Services in Moodle

Moodle is currently the main open source e-learning platform and the second most widespread among LMSs [22]. Moodle is implemented in PHP [23], which makes it a highly accessible for any institution that wishes to use a simple LMS. In addition, there are a great many tools deployed by independent developers. Therefore, Moodle has extensive functionality.

Web services in Moodle are implemented following both usability and functionality philosophies. Therefore, Moodle has no static deployment of web services. These services are dynamically deployed and can be adapted to the users' requirements.

However, web service support in Moodle 2.0 is very limited.

A. Protocols Supported

One of the goals of Moodle's web services is usability. Therefore, Moodle implements different web services protocols:

- *XML-Remote Procedure Call* (XML-RPC) [24].
- *Action Message Format* (AMF) [25].
- *Representational State Transfer* (REST) [26].
- *Simple Object Access Protocol* (SOAP).

B. Architecture of web services

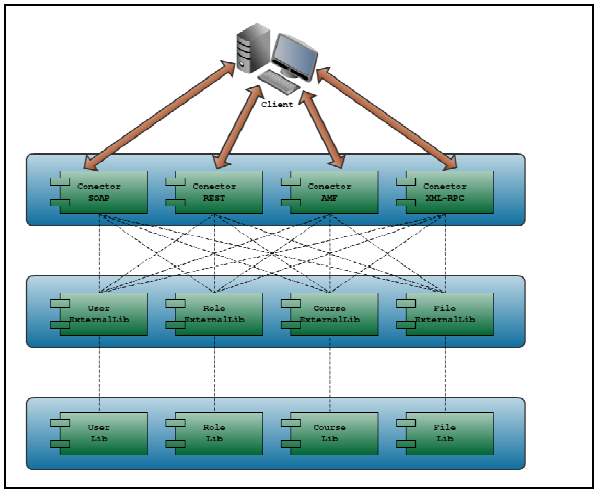

Figure 4. Moodle 2.0 Web service architecture

As Fig. 4 shows, Moodle web services have three tiers that are used to dynamically set up the services:

- *Library/API*. The web services are implemented on this API tier that performs the operations offered by these web services.
- *ExternalLib*. This is the set of the implemented operations that are used by Moodle's web services. It is an extension of available Moodle modules and, therefore, there is an *ExternalLib* for each module. Table 1 shows the *ExternalLibs* implemented in Moodle 2.0 and their operations.
- *Connectors*. Connectors have two missions: (i) they configure web services according to the user's demands and the operations of the *ExternalLib*; and (ii) they make the web service available. There is a connector per web service protocol (SOAP, REST, AMF and XML-RPC).

Resource	Operations
User	moodle_user_create_users moodle_user_delete_users moodle_user_get_users_by_id moodle_user_update_users
Role	moodle_role_assign moodle_role_unassign
Group	moodle_group_add_groupmembers moodle_group_create_groups moodle_group_delete_groupmembers moodle_group_delete_groups moodle_group_get_course_groups moodle_group_get_groupmembers moodle_group_get_groups
Course	moodle_course_create_courses moodle_course_get_courses moodle_enrol_get_enrolled_users
Resources	moodle_file_get_files

Table 1. Operations Supported by ExternalLib In Moodle 2.0

C. Access Control

In Moodle, when web services are created, the administrator of the e-learning platform determines their availability to external users. This feature is not as powerful as Blackboard's security policy, but it allows custom levels of access and operations per web service.

V. Web Services in Sakai

Sakai is a modern e-Learning platform promoted by several universities and other institutions [27].

Sakai implements enough web services to fulfill management of the LMS. Unlike Moodle, Sakai's web service architecture follows a classic model. According to this model, Sakai's web services offer all the functionality and cannot be changed without change development. This model offers a list of clearly defined services.

A. Protocols Supported

Sakai web services aim to offer a set of features capable of managing the platform and not adapt the services to all existing technologies. It uses two web services protocols. They offer all the functionality needed by the web services. These are:

- SOAP.
- REST.

B. Architecture of web services

Sakai's web service architecture is based on a classic model, as Fig. 3 shows. There is a set of web services that offer operations to interact with the e-learning platform. The operations of each web service manage similar information.
Sakai's web services architecture has two variants, according to the communication protocol:

1) SOAP Architecture: Sakai uses the *Apache Axis* framework [28] to implement web services. The web services implemented in Sakai are grouped by type of resource managed. They are [29]:

- *SakaiLogin*: These services are responsible for login facilities. Therefore, they need to be invoked before any other service.
- *SakaiPortalLogin*: These services are needed to help connections from Portal software such as uPortal [30].
- *SakaiScript*: This is a functionally-rich service that includes the main services needed for manipulating users, sites, memberships and permissions on sites.
- *SakaiSession*: This service returns the session information.
- *SakaiSigning*: This service enables external application to verify a user.
- *SakaiSite*: These services allow site handling. It is well worth mentioning that the methods with the word *DOM* [31] are returning strings in a specific *XML* format [32].

2) REST architecture: Sakai's RESTful services are more intuitively described than SOAP services. REST protocol fits in well with the most common application type: *CRUD* (Create, Read, Update, Delete) operations. Each web service manages a specific Sakai resource. They are: connection management, group, group membership, me, presence, search, site, site membership, user, files batch request and activity.

VI. Analysis

A. Web Services Availability in e-learning platforms

This section analyzes and compares the web services availability in the three e-learning platforms analyzed.

This analysis considers the main functionalities needed to use e-learning platforms (e.g. session management, user management, etc.), and analyzes them from the perspective of CRUD operations.

In this analysis four options can be selected for the availability of the CRUD operations for e-learning functionalities:

- Fully supported. CRUD operations are implemented. In addition, the operations that allow performance of the same functionalities as the web deployment of the e-learning platforms are also implemented.
- Supported. Only CRUD operations are implemented.
- Poorly supported. Only part of the CRUD operations is implemented.
- Not supported. None of the CRUD operations are implemented.

Table 2 summarizes this analysis.

According to this analysis, both Blackboard and Sakai are one step ahead of Moodle, although they do not fully implement all the e-learning functionalities used when interacting with e-learning platforms.

In most cases, only web services related to the user, course, announcements and session management are implemented. However those related to calendars, and communications tools (e.g. forums, mail, blog, etc.) are not supported by any e-learning platform.

This is an important drawback because our experience with the UCM VC tells us that communication tools are extensively used by both students and teachers.

Finally, the platforms offer the same operations to all protocols supported. Therefore, Blackboard Learn supports SOAP; Moodle supports SOAP, REST, XML-RPC and AMF; and Sakai supports SOAP and REST.

Web service	*e-learning platform*		
	Blackboard	***Moodle***	***Sakai***
Session	Fully supported	Fully Supported	Fully Supported
User	Supported	Supported	Supported
Role	Supported	Poorly Supported	Supported
Enroll	Fully Supported	Fully Supported	Fully Supported
Course	Supported	Poorly Supported	Supported
Resource	Supported	Poorly Supported	Not Supported
Announcement	Fully Supported	Not Supported	Not Supported
Forum	Not Supported	Not Supported	Not Supported
Calendar	Supported	Not Supported	Poorly Supported
Notifications	Not Supported	Not Supported	Not Supported
Internal Mail	Not Supported	Not Supported	Not Supported
External Mail	Not Supported	Not Supported	Not Supported
Blog/ Personal Web	Not Supported	Not Supported	Not Supported
Grades	Supported	Not Supported	Not Supported
External Mail	Not Supported	Not Supported	Not Supported
Blog / Personal Web	Not Supported	Not Supported	Not Supported
Grades	Supported	Not Supported	Not Supported

Table 2. Web Service Availability in e-learning Platforms

B. APIs and web services

This section compares web services with APIs in the three platforms: Blackboard, Moodle and Sakai. This analysis reviews APIs functionalities, and whether they can be accessed by external applications not deployed in the same machine where the e-learning platform runs, although some functions are only naturally used in the context of the platform's web tier (e.g. visual configuration of the user interface).

APIs can be used to extend the basic functionalities provided by the platform (e.g. a new plugin), or to expose the functionalities to external applications (e.g. a web service). However, web services are usually intended to expose the platform functionalities to external applications.

Therefore, APIs have more functionality than web services included in e-learning platforms. For example, functions to configure web application of the platform need no to be offered in the form of web services. Other example is the support classes including in API libraries, such as special data structures used to manage platform dataset (e.g. a set of cites). However, it can be useful to import these classes using an application library. This importation forces the native platform language support in the importing application.

For the sake of classification, API resources can be classified into four categories:

- *Category I*: Functions to manage the platform (e.g. visual configuration of user interface). These functionalities permit modify the platform, web user interface, general parameters, etc. These functions are not usually available as web services.
- *Category II*: Support classes (e.g. data structures used to manage a set of cites). These classes include special data structures, tools to manage dataset, etc. These functions are not usually available as web services but they can be used if the client application includes them.
- *Category III*: Functions used to access to persistent data (e.g. user registration). These classes give access to courses, announcements, etc. of the platform and manage persistent data. These functions are both available in APIs and as web services, because they enable interaction with the platform and its contents.

The category II and category III are related because some classes of the category II manages the data obtained with the functionality offered by the category III.

Programming Language	*e-learning platform*		
	Blackboard	***Moodle***	***Sakai***
Native	Yes	Yes[2]	Yes
No native	Yes[1]	No	Yes[1]

[1]if the language supports java library
[2]with dependencies between API and the rest of classes of the Moodle engine

Table 3. API availability for external applications

In addition, in order to use the classes of the category II is needed that the platform API can be used by external applications. Table 3 shows the API availability for external applications, in the native programming language and in other programming languages.

Next sections compare the functionality offered by APIs and web services in each analyzed platform.

API Resources	Category	Web service availability
Announcement	II / III	Yes
Bookmark	II / III	No
Calendar	II / III	Yes
Category	I	n/a
Course	II / III	Yes
DataSource	I	n/a
DiscussionBoard	II / III	No
Portfolio	I	n/a
Filesystem	II / III	Partial[1]
Monitor	I	n/a
GradeBook	II / III	Yes
Navigation	I	n/a
Role	II / III	Yes[1]
User	II / III	Yes

[1]Inside content web service

Table 4. Comparative between API and web service in blackboard

1) Blackboard Learn's API and web services

Blackboard's API is characterized by offering a basic set of resources. This API does not have special classes (*category II*) associated with persistent data. In addition, the set of resources managed by this API is very basic. Moreover, this API has not

support for advanced resources like chat, internal mail, etc.
Blackboard has also a set of web services for most the management of persistent data. In addition, its API provides a set of special classes used to manage data returned by web services. These classes are different of those offered by the API. However, Blackboard does not have classes to manage permanent dataset. Additionally, this API is distributed in a JAR library (Java Archive), which can be imported by programming languages that support Java libraries.
Table 4 shows a relation between the resources offer by the API and web service availability.

2) Moodle's API and web services

Moodle's API is the complete API and provides the advance set of functionality. Table 5 shows the API elements and their classification according to the categories described in the beginning of the section. Moodle includes a set of functions called API Module which includes additional resources: assignment, chat, choice, data, feedback, folder, forum, glossary, imscp, label, lesson, page, quiz, resource, scorm, survey, URL and wiki. These resources are modules that can be included in a course.

API Resources	***Category***	***Web service Availability***
Access	III	No
Activity	II	No
Advanced grading	II	No
Backup	III	No
Blog	II / III	No
Calendar	II / III	No
Comment	II / III	No
Conditional activities	II	No
Course	II / III	Partial
Data definition	III	No
Data Manipulation	III	No
Events	II / III	No
File	II / III	Partial
Filter	IV	No
Form	II / III	No
Groups	II / III	Si
Grade	II / III	No
Logging	III	No
Message	II	No
Module	II / III	No
Navigation	II	No
Output	III	No
Page	II / III	No
Plagiarism	II / III	No
Preferences	II	No
Portfolio	III	No
Question	III	No
Rating	II / III	No
Repository	II	No
RSS	II / III	No
String	II / III	No
Tag	II	No
Time	II / III	No
Unit	III	No
User	II / III	Si

Table 5. Comparative between API and Web Services in Moodle

3) Sakai's API and web services

The functionality offered by the Sakai API is very complete. This API can manage all the platform's aspects: access to permanent data, web application configuration, etc. In addition, the Sakai's API has support classes that provide functionality to handle datasets. For example, the Citation classes offer functionalities to: handle a set of citations, configure the appearance of the citations, perform searches, share citations between users, use it in other platform modules, etc.

API Resources	***Category***	***Web service availability***
Announcement	II / III	No
Calendar	II / III	Partial[1]
Chat	III	No
Chefttool	I	n/a
Citation	II	n/a
Courier	I	n/a
Login	III	Yes
Gradebook	II / III	No
Group	II / III	Yes
MailArchive	III	No
Message	III	No
News	II / III	No
Podcast	II / III	No
Portal	III	Yes
Postem	III	No
Presence	II	n/a
Resetpass	III	No
Rights	II / III	No
Role	II / III	Yes
Section	II / III	No
SiteAssociation	I	n/a
SiteManage	II / III	Yes
Taggable	I	n/a
User	II / III	Yes
Warehouse	I	n/a

[1]Only support the copy the calendar between courses)

Table 6. Comparative between API and web services in Sakai

The use of the Sakai's API as library in external applications is more complex than the use of Blackboard's API because Sakai's API is not distributed as JAR library. However, the source code can be downloaded from the website and includes in an application project.

The persistent data accessible using the web services is very limited compared with the persistent data accessible by this API, as table 6 depicts. This analysis is similar to the analysis performed in the previous section: Sakai's API supports much more functionality than Sakai's web services.

VII. Conclusion and future work

At present, taking into account the requirements of e-learning projects, integration capabilities are needed for most e-learning platforms. Web services enable transparent integration of e-learning platforms in environments such as virtual campuses.

However, according to the analysis carried out in this paper, current implementation of web services does not fulfill the requirements of advanced users. Thus, the implemented

functionality in terms of web services is less than half of the functionality offered by the web e-learning platforms.

On the contrary, APIs offer a good set of functionality, but they are intended to be used by applications written in the same language, binding the external application with the platform programming language. In addition, APIs do not have good libraries for facilitating resource manipulation, and therefore, resource manipulation becomes a complex task.

Blackboard is the most widely used e-learning platform and is also the most experienced. This is reflected in the implementation of its web services, Blackboard offers the greatest functionality implemented as web services, and most of persistent data stored in the application is accessible using these web services. These services have a classical implementation, similar to Sakai. However, Blackboard has increased modularity, and it also has a web service definition per resource or functionality. These features make Blackboard's services very understandable. Regarding, Blackboard's API, it is distributed as a JAR library, which facilitates its use in external applications, whenever the application supports Java libraries.

Moodle has a characteristic implementation of web services. Unlike other platforms, the web services deployment architecture is dynamic and the end-user creates it. This architecture has a set of operations implemented. These operations can be added to web services and determine all the functionality that they can have. This architecture adds an important personalization feature to Moodle's web services. However, Moodle does not offer enough web services to support the needs of a normal user, although it implements several communication protocols. Perhaps fewer protocols and more web services would be a more balanced approach. In addition, Moodle's architecture for web services is more complex than its counterparts' architecture. Regarding, Moodle´s API, its use in external applications is complex because there are dependencies between the Moodle API and the rest of classes that make up the e-learning platform.

Sakai has classic web services implementation architecture, classifying them according to their functionality. However Sakai's implementation of web services has a large drawback: all the functions are implemented in a single web service (i.e. a single "Web Service Description Language (WSDL)" [33] interface is provided). In addition, only the basic persistent data is accessible using web services. Regarding Sakai's API, it has useful tools that can be used in external applications and, although the API is not distributed as a JAR library, the source code can be exported as a library because Sakai is an open platform.

Comparing web services' interfaces, they are very heterogeneous. Therefore, they are unsuitable to be directly used in an architecture such as the one promoted by the VCAA Project. Thus, Moodle deployment architecture is completely different and it does not have a stable set of web services. Blackboard and Sakai, despite having similar architectural philosophies, structure their web services in different interfaces. Therefore, their integration with each other is not trivial.

Future work includes the development of a common set of interfaces for e-learning functionalities and their implementation in Blackboard, Moodle and Sakai. The final goal is the development of a virtual campus isolated from its underlying e-learning platform, as the VCAA Project promotes.

Acknowledgments

El Ministerio de Educación y Ciencia (TIN2009 - 14317 - C03 - 01), *La Comunidad Autónoma de Madrid* (S2009 / TIC - 1650) and *La Universidad Complutense de Madrid* (Group 921340) have supported this work.

References

[1] E. Kaplan-Leison. *ASTD Learning Circuits. Glossary*, 2001 http://www.learningcircuits.org/glossary

[2] G.C. Van Dusen. "The Virtual Campus: Technology and Reform in Higher Education", *ASHE-ERIC Higher Education Report*, 25 (5), 1997

[3] D.H. Allison, P.B. DeBlois and the EDUCAUSE Current Issues Committee (2008). "Current IT Issues Survey Report", *EDUCAUSE Quarterly*, 31 (2), 2008 http://www.educause.edu/ir/library/pdf/eqm0823.pdf

[4] R.M. Epper and M. Garn. "The Virtual University in America: Lessons from Research and Experience", *EDUCAUSE Centre for Applied Research* (ECAR) Research Bulletin, 2004 (2), 2004 http://www.educause.edu/ir/library/pdf/ERB0402.pdf

[5] PLS RAMBOLL. *Studies in the Context of the E-learning Initiative: Virtual Models of European Universities*, 2004 http://www.elearningeuropa.info/extras/pdf/virtual_models.pdf

[6] K.C. Green. *The 2005 National Survey of Information Technology in U.S. Higher Education: Growing Campus Concern about IT Security; Slow Progress on IT Disaster*, 2005

[7] A. Navarro, J. Cristóbal, A. Fernández-Valmayor, C. Fernández, H. Hernanz, S. Guillomía, and F. Buendía. "Towards a New Generation of Virtual Campuses", *AICT 2010*, 2010

[8] CampusProyect. http://www.campusproject.org

[9] F. Huertas, and A. Navarro. Web Services Availability in e-learning Platforms. *Seventh International Conference on Next Generation Web Services Practices (NWeSP'11)*, pp. 170-175, 2011

[10] UCM Virtual Campus. https://www.ucm.es/campusvirtual

[11] A. Navarro and A. Fernández-Valmayor. "Conceptualization of Hybrid Websites", *Internet Research*, 17, pp. 207-228, 3007

[12] T. Erl. *SOA Principles of Service Design*. Prentice Hall, 2007

[13] D. Alur, J. Crupi and D. Malks. *Core J2EE Patterns: Best Practices and Design Strategies*. 2nd edition. Prentice Hall/Sun Microsystems Press, 2003

[14] F. Santanach, A. Bertran, C. Ors and M. Gener. "Campus Project. Guide for Application Developers", *Campus Project*, February 2010

[15] G. Collier and R. Robson. "What is the Open Knowledge Initiative?", Massachusetts Institute of Technology (MIT), September 2002

[16] "Open Service Interface Definitions – Full Documentation V2.0", Massachusetts Institute of Technology, 2004

[17] Blackboard Learn. http://www.blackboard.com/Platforms/Learn/Overview.aspx
[18] "Managing Online Education". *The Campus Computing Project*, November 2010. http://www.campuscomputing.net/sites/www.campuscomputing.net/files/ManagingOnlineEd2010-ExecSummaryGraphics_1.pdf
[19] G. Ritter. "Blackboard, Building Blocks and Libraries". 2003 http://library.blackboard.com/docs/BuildingBlocks/Bb_datasheet_Building_Blocks_Libraries.pdf
[20] N. Mitra and Y. Lafon. "SOAP Version 1.2 Part 0: Primer"; W3C, 27 April 2007
[21] T. Dierks and E. Rescorla. "The Transport Layer Security (TLS) Protocol Version 1.2". RFC 5246, IETF, August 2008, Updated by RFCs 5746, 5878
[22] Campus Computing Project. Study of the role of eLearning. http://www.campuscomputing.net/
[23] PHP: Hypertext Processor. http://www.php.net
[24] D. Winer. "XML-RPC Specification", XML-RPC.com, Jun 2099
[25] "AMF 3 Specification", Adobe Systems Inc, 2006.
[26] L. Richardson and S. Ruby. *RESTful Web Services*, O'Reilly, May-2007
[27] S. Lonn. "Sakai 2009 Multi-Institutional Survey Initiative (MISI)", *Sakai project*, 2009
[28] "Web Services – Axis", *Apache Web Service Project, Last Version 1.4*, 2006.
[29] A.M. Berg and M. Korcuska. "Sakai Web Services: Connecting to the Enterprise"; *Open Source* , June 2009
[30] J.A. Lewis, "Sakai and uPortal integration". 3 July 2008 https://confluence.sakaiproject.org/download/attachments/39616558/Sakai+uPortal+Integration.pdf
[31] P. Le Hégaret, R. Whitmer and L. Wood. "Document Object Model (DOM)". *W3C DOM IG*, 2005
[32] M. Murata, S. St. Laurent and Kohn. "D.: XML Media Types"; *RFC 3023, IETF*, 2001
[33] E. Christensen, F. Curbera, G. Meredith and S. Weerawarana. "Web Services Description Language (WSDL) 1.1", *W3C*, 2001