Early Experiences with Responsive Open Learning Environments

Martin Wolpers, Martin Friedrich
(Fraunhofer FIT, Sankt Augustin, Germany, {martin.wolpers, martin.friedrich}@fit.fraunhofer.de)

Ruimin Shen, Carsten Ullrich
(Shanghai Jiao Tong University, Shanghai, China {rmshen, ullrich_c}@sjtu.edu.cn)

Ralf Klamma, Dominik Renzel
(RWTH Aachen, Aachen, Germany, {klamma, renzel}@dbis.rwth-aachen.de)

Abstract: Responsive open learning environments (ROLEs) are characterized through their openness for new configurations, new contents and new users, and through their responsiveness to learners' activities in respect to learning goals. Openness specifically encompasses the ability to include new learning material and new learning services. These can be combined either in a static fashion or dynamically, therefore allowing learners to create their own learning environments. Consequently, throughout the lifetime of a ROLE, new configurations will be created by learners, usually adapted to their needs, requirements and ideas. In this paper, we will describe first experiences using ROLEs for a course on foreign language learning at the Shanghai Jiao Tong University in China. Results of our trials are two-fold: on the one hand, widget, container and other enabling technologies are insufficiently mature to allow large-scale deployment. On the other hand, the results are encouraging as students which learn how to use ROLEs clearly benefit from their use.

Keywords: Personalized learning environment, Responsive open learning environment, language learning, inter-widget communication

1 Introduction

Responsive open learning environments (ROLEs) are characterized through their openness for new configurations, contents and users and through their responsiveness to learners' activities in respect to learning goals. Openness specifically encompasses the ability to include new learning material and services. These can be combined either in a static fashion or dynamically, therefore allowing learners to create their own learning environments. Consequently, throughout the lifetime of a ROLE, new configurations will be created by a learner, usually adapted to her needs, requirements and ideas. In addition, ROLEs respond to the actions the learners carry out within them. Such actions can be as simple as downloading a suggested reading and as complex as participating and interacting with others learners in a complete course about a certain topic. Responsiveness then requires the system to respond to these

activities, for example by incorporating previously downloaded documents into the suggestions for new readings (as kind of a recommender system) or by offering new services that better suits the needs of learners. For example, a learner might be suggested to use a simulation and conceptual mapping tool to understand a certain problem instead of reading theoretical literature as was provided before.
In the context of the research project ROLE (Responsive Open Learning Environments), the infrastructure is developed to enable the creation of individual open responsive learning environments. The ROLE project aims to enable learners to assemble and re-assemble their own learning environments which become personal learning environments (PLE) in due course. This is of particular relevance in the critical lifelong learning transition phases when inhomogeneous groups of learners are treated in a one-size-fits-all way since there is no way to respond to their individual strengths and weaknesses. Even worse, in such transition phases learners are typically required to become accustomed to working with an entirely new virtual learning environment (VLE). The ROLE project will enable the learner to easily construct and maintain her own PLE consisting of a mix of preferred learning tools, learning services, resources, etc. In this way the level of self-control and responsibility of learners will be strengthened, which is seen as a key motivation aspect and success factor of self-paced, formally instructed as well as informal/social learning. Especially in the ROLE testbed at Shanghai Jiao Ton University (SJTU) students are mostly adult learners who have limited knowledge of Web tools. The students often do not know or have difficulties using specific tools. Thus the personalization of their learning environment such as assembling tools they are familiar with is very important for them.

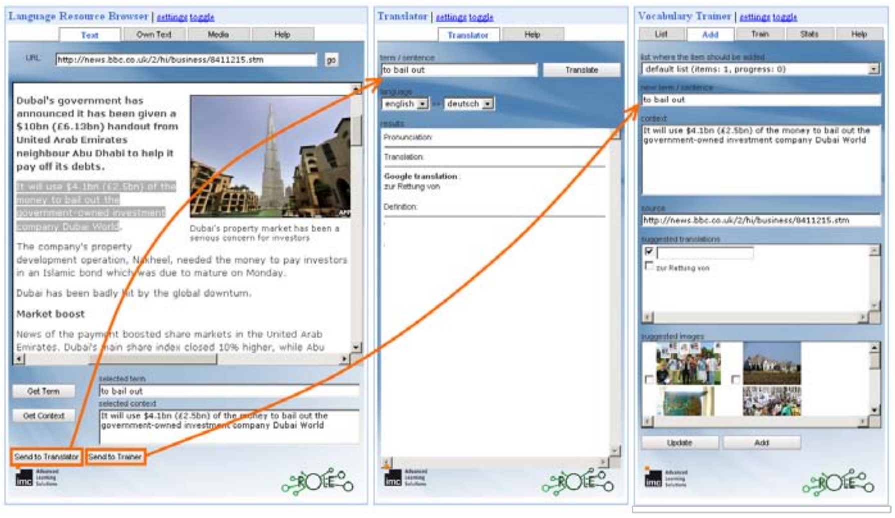
This paper describes a scenario regarding ROLE specific features and describes the technical implementation and evaluation in Chapter 2. The prototype supports language learning through a web based environment where learners use several intercommunicating widgets to learn a language. Chapter 3 describes first experiences using ROLE in a specific testbed. The results and insights gained from the application of the prototype in the testbed are summarized in Chapter 4.

2 Prototype description

To visualize the potential of the ROLE project and gather experience in using the technologies required within the project we built a first PLE prototype based upon a common language learning scenario derived from the SJTU testbed which is based on established open-standard web technologies. The resulting prototype helps to refine specifications and requirements and furthermore to validate the interoperability of the proposed technologies used within the ROLE project. During the development process of the ROLE prototype for language learning, we collected a set of valuable however negative experiences with the technical development.

2.1 Scenario

In our SJTU language learning scenario we assume that a learner wants to improve her business English skills. She wants to reach this goal by reading texts related to her working field and by learning the relevant vocabulary. Thus she assembles her PLE by adding learning tools (integrated via widget technology) which are related to the language learning task and fit best to her preferences. She decides to add three different widgets, a *Language Resource Browser* widget, a *Translator* widget and a *Vocabulary Trainer* widget, as shown in Figure 1.



URL: http://news.bbc.co.uk/2/hi/business/8411215.stm

Dubai's government has announced it has been given a $10bn (£6.13bn) handout from United Arab Emirates neighbour Abu Dhabi to help it pay off its debts.

It will use $4.1bn (£2.5bn) of the money to bail out the government-owned investment company Dubai World.

Dubai's property market has been a serious concern for investors

The company's property development operation, Nakheel, needed the money to pay investors in an Islamic bond which was due to mature on Monday.

Dubai has been badly hit by the global downturn.

Market boost

News of the payment boosted share markets in the United Arab Emirates. Dubai's main share index closed 10% higher, while Abu

selected term: to bail out

selected context: It will use $4.1bn (£2.5bn) of the money to bail out the government-owned investment company Dubai World

term / sentence: to bail out

language: english – deutsch

results

Pronunciation:

Translation:

Google translation:
zur Rettung von

Definition:

list where the item should be added: default list (items: 1, progress: 0)

new term / sentence: to bail out

context: It will use $4.1bn (£2.5bn) of the money to bail out the government-owned investment company Dubai World

source: http://news.bbc.co.uk/2/hi/business/8411215.stm

suggested translations
☑
☐ zur Rettung von

suggested images

Figure 1: Set of language learning widgets

The *Language Resource Browser* is used to display resources of different media types such as texts, videos or audio files. Furthermore, it offers the possibility to trigger different actions on words which the learner is missing. For example the learner can select missing vocabularies and perform a specific action. Since the learner decided to additionally use the *Translator* and *Vocabulary Trainer* widget she can look up the selected vocabulary using the *Translator* widget or send them to the *Vocabulary Trainer* widget which gathers a list of words she considers to be important in the future. The learner repeats this procedure during the next days with different resources and later decides to test herself by memorizing those words gathered by the *Vocabulary Trainer* widget. Every time she has problems with a word she can look up its context, such as the sentence where the word occurred, which helps her memorizing the word.

2.2 Technical Implementation

As described in the scenario we have chosen a widget-based approach to enable the learner assembling her own personal learning environment. This approach makes the development of new widgets independent from specific learning platforms and previous widget implementations.

To achieve responsiveness and openness within ROLE we focussed on intercommunicating widgets which further decreases the configuration complexity of widgets for the end users. In a first step we used a Gadget pubsub channel [OpenSocial, 10] as a kind of message bus which enables widgets to publish or subscribe events using a unified message format. Each time a user action is performed inside a widget the event is published by broadcasting a message containing all information about the user action. This message is received by all other widgets connected to the same pubsub channel. The subscribing widgets then receive those messages and decide whether they react to the event or not. This means that, regarding the language learning scenario, the *Translator* widget is able to translate words selected by the learner inside the *Language Resource Browser* widget. By pressing the "Send to Translator" button a message containing the selected word is broadcasted and reacted to by the *Translator* widget by looking up the word in a dictionary and displaying its translation. Thus, all widgets have the opportunity to receive published events and choose whether and how to react depending on the event type, message type, message content, etc. At the current stage, inter-widget communication is limited to widgets in the same local browser instance only. To be more general in future, as a long term goal we aim to realize arbitrary forms of real-time communication between learners, remote inter-widget communication, interoperable data exchange, event broadcasting, etc., by employing the Extensible Messaging and Presence Protocol (XMPP) [Saint-Andre, 09] discussed in more detail in section 2.3.

To host and render the widgets we decided to use the Apache Shindig container [Shindig, 10] which is the reference implementation of OpenSocial [OpenSocial, 10]. OpenSocial provides a common API which enables applications to use social features across multiple websites and further includes a specification for widgets. By using Apache Shindig as container to host and render the widgets, we are able to integrate them into already existing learning management systems and open virtual learning systems including the pubsub feature. Unfortunately, this prerequisite is not valid for most publicly available containers such as iGoogle. A detailed evaluation of further considered widget container technologies such as SocialSite is done in section 2.3.

For further responsiveness of the learning environment in future we aim to provide self-evaluation and recommendation mechanisms. Therefore we developed a widget which monitors the user’s behaviour within her learning environment. Each user interaction inside a single widget triggering a publish event is captured to monitor the users behaviour over time. Thus, the monitoring widget receives every published message, transforms it into Contextualized Attention Metadata (CAM) [Wolpers, 07] and stores it in a central database. Having this information enables the generation of recommendations based on collaborative filtering algorithms [Linden, 03] and the provision of self evaluation mechanisms. For recommendations e.g. the current usage history of the learner can be taken into account and compared with the usage histories of other users. If other users own a usage history similar to the learner one can recommend those documents they have accessed after the current document of the learner. To provide self evaluation functionality CAM can be used to visualize and summarize the actions performed by the user per day, week or month and compare own with other users' activities. Further the CAM Information can be enhanced with manual annotations such as an evaluation of learning success which makes it possible to show the user's learning progress similar to [Wolpers, 09].

2.3 Development-related experiences

For the development of the first ROLE prototype we pursued a collaborative distributed development process on a rather small scale among consortium partners. The longer-term goal is to gain experience for a large-scale roll-out to open source and commercial developer communities outside the consortium, which is one of the main ROLE objectives. As we aim to build a web-based environment, in particular using widget technologies, we first scouted for a development environment suitable for hosting and rendering the sets of widgets and services contributed by the partners in a distributed manner. The goal was to organize our process according to the following policy:

- Every developing partner maintains a local development environment
- One partner maintains an additional integration environment
- All local development environments and the integration environment are equivalent

Especially the equivalence required in the last policy rule was not fulfilled for reasons discussed in the following. Although there exist multiple Open Source solutions for widget containers already, the biggest challenge was their immaturity, especially regarding inter-widget communication, one of the main requirements for our language learning PLE (cf. Section 2.1). During the development process we tested three different development environments, mainly consisting of container technologies for OpenSocial compliant widgets (resp. *gadgets* in OpenSocial terminology):

1. SocialSite (based on Shindig within Glassfish) [SocialSite, 10]
2. Apache Shindig [Shindig, 10]
3. OSDE (Eclipse Plugin with integrated Shindig) [OSDE, 10]

With all of the above systems we encountered at least one of the following problems:

- Lack of Forward Incompatibility
- Client Browser Dependence
- Server Platform Dependence
- Inaccessible Bugs in Generated Code
- Lack of Developer Support
- Incompatibility with External Libraries
- Instability of Draft Standard Specifications
- Missing/Changing Libraries for inter-widget Communication

The first problem was related to container installation, in particular with SocialSite. First, the current SocialSite distribution was restricted to specific, already outdated versions of Glassfish and Shindig, and thus is not forward compatible - an essential property, when working with experimental systems likely to evolve in future. Given the diversity of devices and platforms used by the different developing partners, we quickly had to find out that platform and browser independence was not given at all. Although both Shindig and Glassfish are Java-based, and all partners were using the same installer version, some partner installations worked as expected, while others exhibited unexplainable container side errors. Another issue was browser incompatibility. Even with a working container, some browsers did not render widgets correctly, while others exhibited no problem at all. This problem is highly likely to be related to different implementations of JavaScript engines implemented in browsers. Another essential problem was the inaccessibility of bugs in JavaScript code. In many cases, problems occurred outside the source code under developer control. The reason was a malfunction in the code production performed by the container itself to render the widgets. Furthermore, error messages were cryptic and incomprehensible and thus did not provide any hint to the original location of an error. In conjunction with the above problems we had to experience that developer support by the SocialSite team was not available at all. At the time of writing this document, it seems quite obvious, that SocialSite is dead. After the initial experiments with SocialSite, we communicated possible alternatives. The alternative usage of Apache Shindig instead of SocialSite was also not successful for all ROLE developers at that point in time. However, due to the currently ongoing incubation process at Apache, we can expect to receive support and more mature versions from the Shindig team in future. The OpenSocial Development Environment (OSDE), which also uses a built-in Shindig test server, was considered as a helpful tool with respect to the development of single widgets, but proved to be impractical, when it comes to the development of multiple intercommunicating widgets. Further problems were related to the incompatibility of external JavaScript libraries with the Widget container, which resulted in strange code rewriting effects or cross-domain issues. With special regard to the publish/subscribe based inter-widget communication approach we are pursuing in ROLE, we experienced that at the current time Shindig is in a transition phase of changing from the pubsub feature included in the Google gadget API to the pubsub mechanism included in Open Ajax Alliance Hub 2.0 [OpenAjaxAlliance, 10]. Regarding all of the above issues, we can state that the collaborative distributed development process of a PLE was challenging due to the lack of a stable and reliable development environment.

With the long-term goal in mind to create a class of ROLE widgets supporting arbitrary forms of real-time communication between learners, remote inter-widget communication, interoperable data/metadata exchange, event broadcasting, etc., we put a special focus on the Extensible Messaging and Presence Protocol (XMPP) [Saint-Andre, 09]. Due to its rich set of built-in properties and features, its set of official XMPP Extension Protocols (XEPs) and the variety of application use cases in ROLE, we explored the protocol in conjunction with widget technologies and JavaScript libraries. Although there exist quite a number of XMPP libraries for JavaScript, we learned that many of them are still insufficiently mature for the realization of our goals and definitely need improvement. In particular, we experimented with a selection of libraries listed on the official XMPP website. A comparison of these libraries is presented in Table 1.

Library	**Connection Technique**	**Code/Documentation Maturity**	**UI Element Support**	**Supported Features & XEPs**
dojox.xmpp [Dojo, 10]	XMPP over BOSH	ok/weak	yes (basic)	Core: IM, Presence, Roster XEP-0004: Data Forms XEP-0030: Service Discovery XEP-0085: Chat State Notifications XEP-0206: XMPP over BOSH
xmpp4js [XMPP4JS, 10]	XMPP over BOSH	ok/ok	no	Core: IM, Presence, Roster XEP-0004: Data Forms XEP-0030: Service Discovery XEP-0049: Private XML Storage XEP-0077: In-Band Registration XEP-0085: Chat State Notifications XEP-0100: Gateway Interaction XEP-0206: XMPP over BOSH
strophe.js [Strophe, 10]	XMPP over BOSH	ok/weak	no	XEP-0206: XMPP over BOSH
js.io [JS.IO, 10]	(CSP) Comet Session Protocol	weak/weak	no	unknown

Table 1 Comparison of JavaScript Library Support for XMPP

Probably the most important notion was that libraries exhibited different levels of code and documentation maturity and different sets of core and extension protocol features implemented. The minimum level of functionality realized by all of the above libraries is the emulation of persistent, stateful, two-way connections to an XMPP server using comet techniques such as *Bidirectional-streams Over Synchronous HTTP*

(BOSH) [Paterson, 08], often in conjunction with additional libraries for XMPP XML stanza construction and parsing. The next level of functionality was the implementation of XMPP core services, i.e. instant messaging, presence, and roster lists, which were only realized by dojox.xmpp and xmpp4js. Functionality beyond XMPP core services often only include very basic XEPs, e.g. for service discovery, server ping, data forms, etc., but does in all cases not include the implementation of central XEPs supporting powerful communication techniques such as multiuser chat, publish/subscribe, etc. Although code and documentation quality as well as the set of supported XEPs in xmpp4j seemed superior, we decided to start the experimental implementation of further XMPP XEPs based on dojox.xmpp, because it provides cross domain solutions, is integrated into the well-established dojo toolkit, and is designed to make use of the dojo widget framework for the provision of custom XMPP-powered UI elements. Currently, the XMPP Multiuser Chat XEP-0045 [Saint-Andre, 10] is under development in conjunction with a pubsub enabled XMPP multiuser chat gadget. As future work, further central XEPs will be implemented and evaluated in a whole class of XMPP-enabled Widgets.

Drawing the conclusions from our experiences, we can state that the technologies we experimented with were insufficiently mature for the deployment of a stable integrated prototype ROLE PLE assembled from a set of innovative tools realized using a combination of different bleeding-edge Web technologies. We finally managed to deploy our prototype in Graaasp [Bogdanov, 10] in a rather stable version, but still with a lot of open issues to be tackled in later development stages of the ROLE project. The evaluation of our prototype in one of the ROLE testbeds is described in the next section.

3 The Shanghai Jiao University Testbed

ROLE is based on the vision that responsive open learning environments will accompany learners throughout their learning career, from during formal education to learning at the workplace. Several testbeds enable the ROLE project to collect requirements in various settings. One major test bed takes place at the School of Continuing Education (SOCE) at Shanghai Jiao Tong University. SOCE is the online branch of SJTU. Online colleges/universities such as SOCE play a particular role in the Chinese education system. Foreseeing the enormous demand for higher education, the Chinese government decided in 1998 to establish a number of online institutions that were open to those students who did not pass the university entrance exams (in 2007, 10.1 million students applied for 5.67 million places) [People's Daily Online, 2007]. The ROLE SOCE testbed enables us to learn about PLEs from the viewpoints of “average” users, that is learners who are not highly technically literate, who have limited time due to jobs and families.

3.1 Test-bed Description

The students on the SOCE (the situation is similar at the other online institutions) are mostly adult learners who have a job. They study two to three years for associate or bachelor degree courses. The courses are similar to regular university courses with the difference that teaching takes place in the evenings and weekends. Each class lasts for three hours. Our blended classrooms are based on the Standard Natural Classroom model [Shen, 08], in which students can either attend the classes in person or attend via the live Web broadcast. Lectures can also be downloaded for asynchronous watching. During the lectures, online students can communicate with the teacher via short messages. The SOCE software system records, encodes and broadcasts each lecture in real-time. Target devices/destinations include desktop/laptop PC via ADSL broadband connections, IPTV devices via the Shanghai Educational IPTV channel, universities in western China via two satellite connections and mobile devices via GPRS. SOCE produces about 6GB of content every day. The current SOCE learning management system supports student-student communication via forums. Teachers can create exercises/homework to be completed by the students.

Some of the gaps in the current learning processes are representative for most Chinese educational settings. Students are degree/certificate oriented: the primary goal is to receive a formal acknowledgment of the mastery of the study subject, not the mastery of the subject. Group work is hard to perform as it is seldomly performed at school. In consequence, the practical application of learned knowledge is difficult for Chinese students. For instance, in language learning, the students have extreme difficulties communicating with native speakers. They are insecure, shy, and make frequent mistakes. Furthermore, in our lectures we over and over make the experience that the students do not know, have difficulties, or simply do not use existing tools such as online dictionaries, pronunciation tools or microblogging sites. Reasons given include lack of time, interest and motivation. In ROLE, we experiment with different scenarios to address these problems.

We experimented with an Open Learning Environment and a variety of tools (video conferencing, micro-blogging, translation services, text-to-speech, etc) over two semesters in the courses English Listening and Speaking, German I/II, French I/II and Introduction to Computer Science. About 50-100 students attended the language learning courses, and about 1.200 students took part in the Computer Science class. From our experience we knew that the students at SOCE have limited knowledge about Web tools (RSS is virtually unknown), only limited time at their disposal, and limited technical expertise. Furthermore, in the Confucian culture of China learning is still very teacher-centered [Zhang, 07], and students are not used to actively contribute within a class. We therefore decided to build PLEs according to the suggestions of the teachers and make these pre-built PLEs accessible to the students. The teachers presented the PLEs in class, and showed example usages. In order to encourage usage, the students were assigned homework that required using the PLEs. We then observed the students' usage of the PLEs and collected feedback from the teachers and students by interviews and questionnaires.

The prototype described in Section 2 was not finalized at the time we started our experiments at the SJTU testbed. Therefore, we used a slightly different version based on the portal Liferay. While we did not offer advanced features such as inter-widget communication, the basic functionality was the same and therefore enabled us to collect information about the usage of an OLE in a higher education context.

3.2 Testbed related experiences

Here, we will briefly report on our experiences with using the PLE in the SJTU testbed for language learning support. The integration of an OLE into a course is a complex task that does not work like clockwork, but requires careful introduction, integration into the classroom and a clearly perceived value by the students.

For the students it is not sufficient to present a single example of how to use the OLE in class. In itself, this will not enable and motivate students sufficiently to work with the embedded tools. In our experiments only a minority of students (about 10%) actually made use of the services and did the associated tasks (however, even "regular" homework to be done via the school LMS is only done by about 50% of the students). We hope to improve these figures by increasing the incentive to use the PLEs. Measures will include extra points for the final grade, but also better communication of the value of a PLE. In particular, students need to understand how the tasks and services work which can help them to achieve their goals. Each OLE usage needs to be broken down into the individual steps. For instance, the task of doing a spoken self-introduction can involve the steps of writing the introduction in the native tongue, translating it, polishing it, using a text-to-speech tool to listen to it, a recording to practice one's own pronunciation, and finally recording and publishing it. As a teacher, demonstrating this whole sequence only once or twice overtaxes the student. Each single step needs to be shown and done by the students several times. The single steps as well as the combination of services should be assigned as homework, giving the students an initiative for practicing. Breaking down the usage of an OLE helps students understand how to use it. Even more important is that the students understand why they should use a PLE. Students need to see the value of performing additional tasks which are not directly related to language learning.

For western students the access to the internet is almost as common as watching TV. Applying a web based PLE to other countries requires taking into account country-specific restrictions. For example, as SJTU is located in China the access to quite a large number of Web sites including social networks such as Twitter, Facebook and Friendfeed as well as many Web services is blocked. Furthermore, many institutions such as schools or companies restrict the access of sites deemed to be inappropriate for their scholars.

Furthermore, we experienced new directions for technical research which should be taken into account for the development of the next ROLE prototypes. This includes the necessity to create accounts for some services which require a login or the necessity to have a single sign on feature since logging in several times leads to frustration of the students. Finally, even though logging data was collected in the SJTU testbed, students did not raise privacy concerns. On the contrary, repeatedly, students uttered concerns that their contributions in the employed tools might not be noticed by the teacher.

4 Conclusions and Outlook

Responsive Open Learning Environments create enormous challenges on the psycho-pedagogical as well as on the technical side. Some of the challenges were sketched in this paper. The ROLE project has chosen a spiral "organic" development and deployment process to cope with the challenges mentioned above. We started with very small scale projects only covering partial requirements of the five testbeds of ROLE. This was necessary to understand the different development cultures among the developers at the different ROLE sites and also the language of the end users stating their requirements - an experience that will surely help for the larger roll-out planned. Indeed, beside those challenges the interplay between end users of personal learning environments and developers of enabling technologies and products is crucial. It is not only a problem that the current generation of developers do not recognize the time horizons covered in the development of personal learning environments, thus making industrial scale standards deployment inevitable, but also open source developers and company developers do not understand each other. At the moment, flexibility and openness is more on the side of the company developers.

In the near future, the development process is widened to cover more than one testbed. This is due to the necessary learning experience for developers that results developed in the context of one scenario are not necessarily transferable to another scenario. As the ROLE project is targeting the transitions of learning, the future scenarios involve learners' shifts of interests and learning goals during the scenarios like leaving the university with some degree of self-regulated learning and entering a company where learning goals are only valid in the light of the company's strategy.

Acknowledgements

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement no 231396 (ROLE project).

References

[Achmed, 99] Ahmed, M., Karmouch, A., Abu-Hakima S.: Key Frame Extraction and Indexing for Multimedia Databases, Vision Interface 1999, Trois-Rivières, Canada, 19-21 May 1999.

[Bogdanov, 10] Bogdanov, E., Salzmann, C., El Helou, S., Sire, S., and Gillet, D.: Graaasp: A Web 2.0 Research Platform for Contextual Recommendation with Aggregated Data. In ACM Conference on Human Factors in Computing Systems (HCI) - Work-in-Progress, 2010.

[Chakrabarti, 99] Chakrabarti, K., Mehrotra, S.: The Hybrid Tree: An Index Structure for High Dimensional Feature Spaces, In Proc. Int. Conf. on Data Engineering, February 1999, 440-447 http://citeseer.nj.nec.com/chakrabarti99hybrid.html.

[Cocoon, 02] Cocoon XML publishing framework, 2002, http://xml.apache.org/cocoon/.

[Dojo, 10] Dojo Toolkit. Unbeatable JavaScript Tools. http://www.dojotoolkit.org/. Last visit March 2010.

[Glassfish, 10] Glassfish Open Source Application Server. https://glassfish.dev.java.net/. Last visit March 2010.

[Hunter, 00] Hunter, J.: Proposal for the Integration of DublinCore and MPEG-7, 2000.

[JS.IO, 10] js.io. Real-time Web Applications. http://js.io/. Last visit March 2010.

[Linden, 03] Linden, G., Smith, B., York, J.: Amazon.com Recommendations - Item-to-Item Collaborative Filtering, IEEE Internet Computing 7 (1), 2003

[OpenAjaxAlliance, 10] OpenAjax Hub 2.0 Specification. http://www.openajax.org/member/wiki/OpenAjax_Hub_2.0_Specification. Last visit March 2010.

[OpenSocial, 10] OpenSocial. A common Web API for social applications. http://code.google.com/apis/opensocial/. Last visit January 2010.

[OSDE, 10] OpenSocial Development Environment (OSDE). An IDE development tool for OpenSocial applications. http://code.google.com/p/opensocial-development-environment/. Last visit March 2010.

[Paterson, 08] Paterson, I., Saint-Andre, P.: XEP-0206: XMPP Over BOSH. Draft Specification. October, 2008. http://xmpp.org/extensions/xep-0206.html. Last visit March 2010.

[People's Daily Online, 07] People's Daily Online: More than 9.5 million Chinese compete in world's largest examination. 2007: http://english.peopledaily.com.cn/200706/07/eng20070607_381831.html. Last visit November 2007

[Saint-Andre, 09] Saint-Andre, P., Smith, K., and Tronçon, R.: XMPP: The Definitive Guide - Building Real-Time Applications with Jabber Technologies. O'Reilly Media. April 2009.

[Saint-Andre, 10] Saint-Andre, P.: XEP-0045: Multi-User Chat. Draft Specification. January 2010. http://xmpp.org/extensions/xep-0045.html. Last visit March 2010.

[Shen, 08] Shen, L., Shen, "The Pervasive Learning Platform of a Shanghai Online College - A Large-Scale Test-Bed for Hybrid Learning Hybrid Learning and Education", In R. Fong, J.; Kwan, R. & Wang, F. L. (ed.), First International Conference, ICHL 2008, Hong Kong, China, August 13-15, 2008, Proceedings, Springer, 2008, 5169, 178-189

[Strophe, 10] Strophe. A family of libraries for writing XMPP clients. http://code.stanziq.com/strophe/. Last visit March 2010.

[Shindig, 10] Apache Shindig. OpenSocial container reference implementation. http://incubator.apache.org/shindig/. Last visit January 2010.

[SocialSite, 10] Project SocialSite. https://socialsite.dev.java.net/. Last visit March 2010.

[Wolpers, 07] M. Wolpers, J. Najjar, K. Verbert, and E. Duval, „Tracking Actual Usage: the Attention Metadata Approach," in Educational Technology & Society, vol. 10, no. 3, pp. 106-121, 2007.

[Wolpers, 09] Wolpers, M., Memmel, M., Giretti, A.: Metadata in Architecture Education - First Evaluation Results of the MACE System, In Proceedings of the 4th European Conference on Technology Enhanced Learning: learning in the Synergy of Multiple Disciplines, 2009

[XMPP4JS, 10] Xmpp4Js. An XMPP client library written in pure Javascript. http://xmpp4js.sourceforge.net/. Last visit March 2010.

[Zhang, 07] Zhang, J. A cultural look at information and communication technologies in Eastern education, In Educational Technology Research and Development, 2007, 55, 301-314