Ubiquitous Personal Learning Environment (UPLE)

http://dx.doi.org/10.3991/ijet.v7iS2.2322

Behnam Taraghi
Graz University of Technology, Graz, Austria

***Abstract*—Web 2.0 technologies opened up new perspectives in learning and teaching activities. Collaboration, communication and sharing between learners contribute to the self-regulated learning, a bottom-up approach. The market for smartphones and tablets are growing rapidly. They are being used more often in everyday life. This allows us to support self-regulated learning in a way that learning resources and applications are accessible any time and at any place. This publication focuses on the Personal Learning Environment (PLE) that was launched at Graz University of Technology in 2010. After a first prototype a complete redesign was carried out to fulfill a change towards learner-centered framework. Statistical data show a high increase of attractiveness of the whole system in general. As the next step a mobile version is integrated. A converter for browser-based learning apps within PLE to native smartphone apps leads to the Ubiquitous PLE, which is discussed in this paper in detail.**

***Index Terms*—e-learning, PLE, TEL, u-learning**

I. INTRODUCTION

Web 2.0 changed the online behavior of users on the World Wide Web (WWW) [1]. It was not merely a new technology. The former consumers are now also producers of the contents on web. This changed the way people interact on the web and opened up new possibilities in education known as e-learning 2.0 [2]. Since then many research studies have been carried out to evaluate these new possibilities [2] and show how they can be used in teaching and learning process, i.e. weblogs [3], podcasting [4], microblogs or social networks [5] and wikis [6].

Mobile learning, known also as m-learning, has become popular in Technology Enhanced Learning (TEL). Since 2000 first surveys demonstrated how the use of Personal Digital Assistants (PDAs) helps to increase the learning efforts [7]. The research in m-learning has gained more attention since the rapid growth of smartphones and mobile applications, driven by Apple's iPhone and Android mobile Operating System (OS). Nowadays many people are permanently online with their mobile devices, share and exchange their ideas across the WWW. This has led teachers as well as learners to use mobile phones in different contexts including teaching and learning purposes [8].

Due to their ubiquitous availability and pervasive use mobile technologies and the social web influences both our daily life as well as learning environments [9] [10]. For education it is quite challenging not to be overwhelmed by lots of different tools. The WWW offers various services like YouTube (for sharing videos), Flickr (for sharing photos), Slideshare (for sharing presentations), Scribd (for sharing documents), Mendeley (for sharing publications) or Delicious (for sharing bookmarks). Each of these services can be used for teaching and learning, but could not be integrated in common learning environments such as Learn Management Systems (LMS) that are widely used in many universities and high educational institutes. Despite this, LMS is a teacher driven environment, where teachers provide the students with contents relevant for a course in a structured way. The regulated integration of online services of the WWW would not help to assist self-regulated learning in LMS.

Learners should be given the freedom to use the WWW the way they want, use services and resources they need for their personal learning goals. Learners must decide themselves, which learning content fits best and which resource will help to increase their learning outcome. Bearing the rapidly growing number of applications and tools in mind that can be used for the described purposes above, it is quite challenging to mange these tools within an learning environment efficiently. Various studies on Web 2.0 usage amongst students [11] underline the fact that it is hard to keep an eye on these tools or even monitor them in an appropriate way. Some of the latest surveys in the area of TEL deal with mashups [12] and personalization as well as the possibilities to manage them. This led to the idea of Personal Learning Environment (PLE) as a concept [13]. The combination of different tiny applications, i.e. in form of widgets, within a framework and with strong relationship to learning aspects is called PLE. Following the idea that the learners themselves can manage these applications according to their needs a PLE is able to offer a new form of personalized learning [14].

This publication describes the already running prototype of a mashup-based PLE[1] at Graz University of Technology (TU Graz) [15]. The PLE has been redesigned in 2011, using metaphors such as apps and spaces, for a better learner-centered application and higher attractiveness. Therefore a general description will be given to show what a learning environment should look like to fulfill this requirement. First statistical data will be presented to show the extent of the impact the redesign has had on the attractiveness and usage of the PLE. Furthermore the very new mobile view of the PLE will be described. An applied approach is introduced that shows how the browser-based PLE widgets (similar to apps) can be converted to desktop widgets on Windows 7 and Mac OS dashboards to build a ubiquitous learning environment.

[1] https:/my.tugraz.at (last visit: 2012-08-30)

II. PLE VERSION 2.0

The main idea of using a PLE at TU Graz is described in details in [16]. The PLE at TU Graz is based on mashup of widgets. Widgets represent independent resources, services, and applications that are integrated into the PLE [17]. The architecture of the User Interface (UI) of the first PLE prototype (version 1.0) [15] consisted of a sidebar navigation element (see Fig. 1(1)) and several widget zones (see Fig. 1(2)). The widget labels were listed on the sidebar and served as navigation elements to find the widgets on the different widget zones. Widgets were positioned in a grid order in three columns on widget zones. The main idea behind the former UI architecture was to take the best of two worlds: the familiar traditional navigation-based UI and the unfamiliar UI, which consists only of widgets. Because of the fact the usage of smartphones working with apps has become popular among our potential users (young students), the widget-based UI is no more unfamiliar to them. Several usability tests were performed on the first prototype based on common methods of Human Computer Interaction (HCI). According to the test results it was decided to redesign the whole UI structure, described in [15], for a better performance and higher user friendliness.

A. Ubiquitousness: Web View

The new UI relies on a full app-based architecture. It consists of a number of spaces, which are unlimited in width and height; they represent multiple personal desktops. The spaces are closely similar to the app environment on smartphone devices. Learners can add as many spaces to their PLE as they need. The concept of spaces for learning environment resembles the user´s real world behavior where the learner can use several desks for learning with different (in)dependent learning resources spread on and arranged in an arbitrary order by the user himself. Fig. 2 shows a user's space, where he has positioned several widgets arbitrarily. The widgets in this example are related to translations (3 widgets on the left), searching for courses and professors (widget in the middle), and searching for books in the library (widget on the right).

It is planned to extend the access on spaces so that it is possible to share spaces between multiple users in future. All users sharing the same shared space will have the same rights within the shared space. Additionally a real time interaction will be possible within shared spaces between users. If one user installs a new widget on a shared space the other users will use the widget in real time too. This new feature would open up many new scenarios for collaborative learning in PLE.

Widgets can be maximized to a full size view (see Fig. 3) for a better user experience or minimized if they are not needed within a session. Users can trigger several actions on each specific widget: configure the widget preferences, uninstall, rate and comment the widget, get in contact with the widget developer, and more. These actions are provided within the header of the each widget. They are faded in as soon as a widget is focused (see Fig. 2 the left widget).

The space management interface provides the possibility to have an overview over all spaces, the widgets within the spaces, and the arrangement of widgets in different spaces. Rearrangement of spaces and widgets as well as

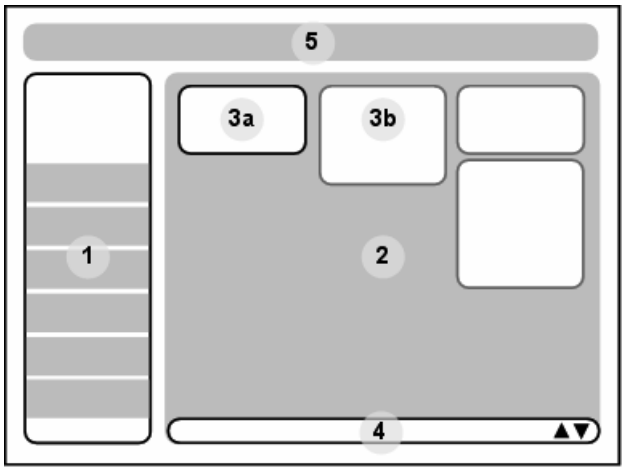

Figure 1. Former UI elements of the PLE (version 1.0)

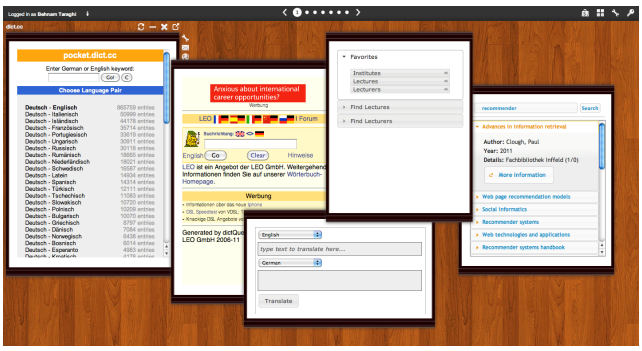

Figure 2. A user's space in PLE, filled with several widgets positioned arbitrarily by the user.

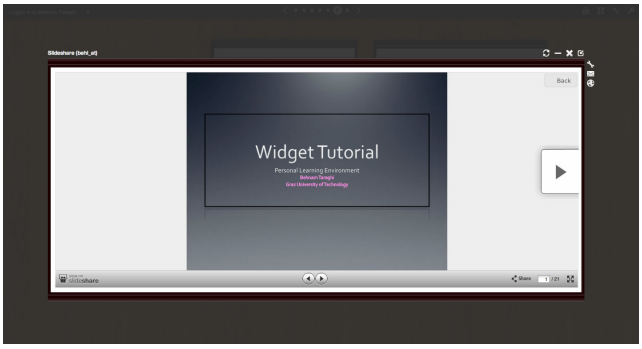

Figure 3. "Slideshare" widget in full size view

uninstalling widgets or deleting, adding spaces are possible within the space management. Additionally the space management can be used as a visual table of contents in order to find the space where a distinct widget is installed and navigate to it. Fig. 4 demonstrates the space management interface of a user, having 6 spaces. Clicking on the space number on the bottom of a space would navigate the user to the corresponding space. Furthermore, selecting a widget would navigate the user to the containing space and show the widget in full size view (see Fig. 3).

The widget store is the place where learners can search for widgets within the PLE and install, rate, and comment the widgets. The widget store resembles the iPhone App Store or Android Market. The UI of the widget store looks like the app stores on the two mentioned smartphones very closely. Furthermore, it is possible to install several instances of a widget in different spaces. This feature is

useful in two cases. In the case the Inter-Widget Communication (IWC) mechanism is applied, different instances can be combined with different widgets. Through IWC two or multiple widgets can exchange data, trigger each other, and accomplish some tasks in serial or parallel. The second advantageous case is the one of shared spaces that were described above. The idea is that users can share an instance of a widget with others and work collaboratively on it within the PLE. Fig. 5 shows the actual widget store of the PLE. Widgets can be sorted on different criteria such as "most used", "top rated", "my widgets" and more. Widgets are classified in different categories. Users can search for widgets in each category as well. Fig. 6 shows the detail view of a widget in the widget space.

A notification module is integrated within the PLE that is triggered by the widgets on demand. Depending on the functionality, widgets notify the user about an actual status within their scope that may be of user's interest. As for an example, the RSS-Feed widget notifies the user about some new unread feeds; the email widget notifies user about some new incoming emails. The notification messages are shown up within the toolbar on the top of the PLE so that they are always visible, even if the notifying widget is not on the active space or minified.

From a technical point of view, the whole environment is based on client-server architecture. The PLE server offers an Application Programming Interface (API) for data retrieve by clients. The client (a web-browser) is developed in JavaScript and is responsible for building the whole UI structure. To increase the performance, web workers are applied in JavaScript. Web workers provide the possibility to run time-consuming JavaScript code in a parallel thread.

According to the statistics we could observe that the number of active users have been increased after the deployment of the new web interface. About 4 times more users have been using PLE since the update. The average frequency of logins has been gone up as well. This lies on a user-centered development, allowing users being engaged in the development, increasing attractiveness e.g. through fun theory and user centered design that will be explained in detail later on.

The new features of the PLE version 2.0 are not restricted only to a new UI. There are different widget specifications on the web. The most popular ones are OpenSocial[2] gadgets and W3C Wookie[3] widgets. PLE widgets are based on an old version of W3C that is not applied in any other widget container. PLE version 2.0 has been extended a short time ago to support OpenSocial gadgets and Wookie widgets. Through this extension the number of provided widgets in the PLE will increase faster in future.

B. Ubiquitousness: Mobile View

The growth of mobile devices such as smartphones and tablets as well as the increased availability of free wireless network access points extend the traditional e-learning into a new form of learning called ubiquitous learning, also known as u-learning. Zhan and Jin [18] define u-learning as a function of different parameters: u-Learning = {u-Environment, u-Contents, u-Behavior, u-Interface, u-Service}.

[2] http://opensocial.org/ (last visit: 2012-08-30)
[3] http://getwookie.org/ (last visit: 2012-08-30)

Figure 4. Space management interface of the PLE.

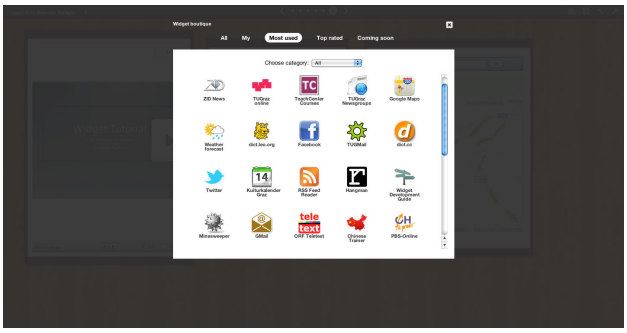

Figure 5. Widget store of the PLE.

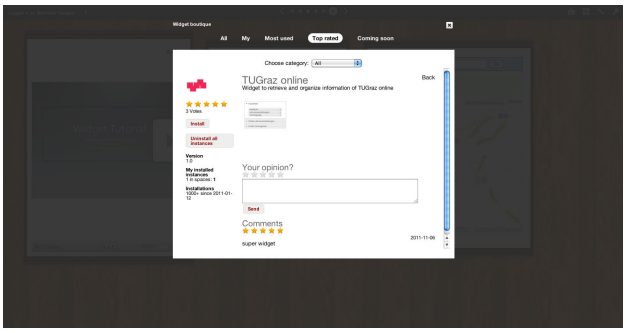

Figure 6. Detail view of a widget within the widget store

One of our main goals was to apply ubiquitous learning within the concept of mashup-based PLEs. The mobile view of PLE was the next step for a ubiquitous environment. As the PLE at TU Graz bases on a Service Oriented Architecture (SOA) only the view had to be refactored for mobile devices. Additionally the existing widgets need to be refactored in a way that they run on mobile devices smoothly. Although this process is generally time consuming, the widget development framework [19] used in PLE at TU Graz reduces the refactoring time to a great extent, as only the view part of widgets need to be refactored. The refactoring process of widgets is still ongoing.

The mobile interface is introduced very briefly as follows: Fig. 7 (left) shows the first page of the mobile view after login. Users can access their spaces and already installed widgets under "My widget" (see Fig. 7 right side). Fig. 8 (left) demonstrates the widget store in mobile view. The detail view of the widget "Geolines" can be observed in Fig. 8 on the right side. The "Geolines" widget solves direct and inverse geodetic problems in 2d-cartesian and ellipsoidal coordinate systems. Fig. 9 (left) shows the widget "Geolines" running on a mobile client. As mentioned before, users can trigger several actions on each specific widget. Clicking on the "option" button on

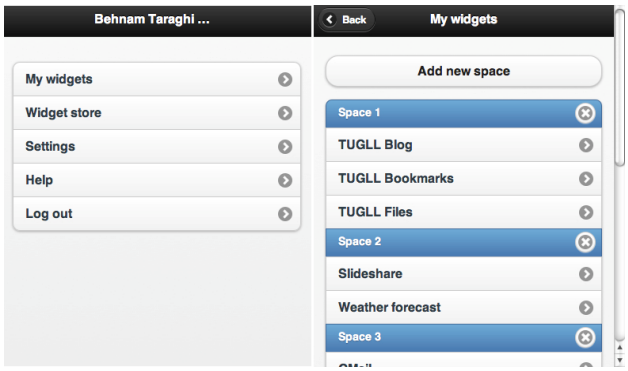

Figure 7. Mobile interface: left: start page, right: list of user's spaces and already installed widgets

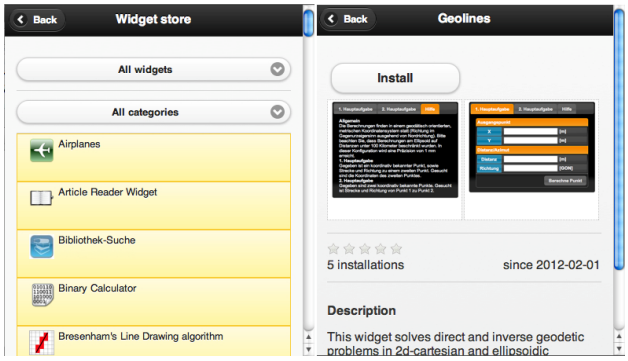

Figure 8. Mobile interface: left: widget store, right: detail view of a widget within the widget store

the top right corner (see Fig. 9 left) opens up a dialog window where the user can select the desired action (see Fig. 9 right side).

Next to the browser-based mobile view, it is possible to offer the PLE widgets as native apps for two known smartphones: iPhone and Android capable mobile phones. There are some free online tools that provide the possibility to convert HTML5 based application to native apps on smartphones. PhoneGap[4] is such a tool that is used for this purpose at TU Graz. An automatic conversion of PLE widgets to the native apps is challenging due to the restrictions on converters. The work on the conversion process is still ongoing. Once it is finalized, it will be possible to run widgets as standalone smartphone apps outside of the PLE. Fig. 10 shows the "calculator" widget that has been converted manually to an Android native app.

C. *Ubiquitousness: Desktop View*

As mentioned above, one of our main goals was to apply ubiquitous learning within the concept of mashup-based PLEs. Another step in this regard is done at TU Graz by converting PLE widgets to Windows desktop gadgets and Mac OS dashboard widgets. Microsoft Windows 7 and Apple Mac OS X both have their own widget engines and are used by 75% of users according to the last PLE statistics. The first step was an automatically porting of the PLE widgets to these platforms to be compatible to the specific devices of the users. This section aims to answer the questions how a widget following W3C specifications can be converted for Windows 7 and Mac OS X, and how this process can be automatized; what challenges and complications emerge from this process?

[4] http://phonegap.com/ (last visit: 2012-08-30)

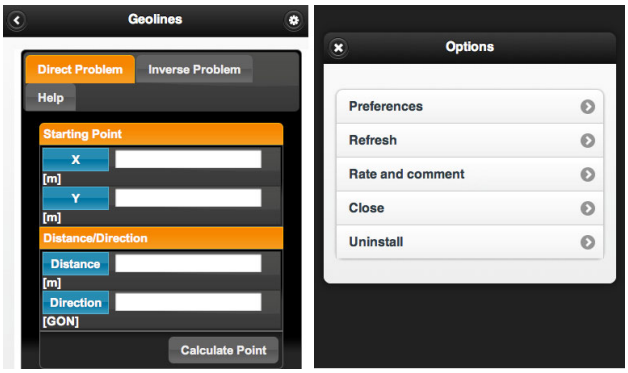

Figure 9. Mobile interface: left:"Geolines" widget running, right: actions the user can trigger on widget.

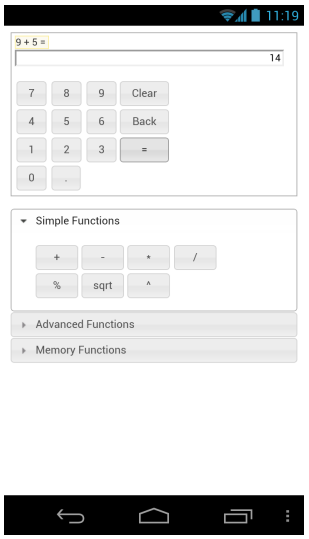

Figure 10. "Calculator" as an Android native app, manually converted from the corresponding PLE widget.

The W3C Widgets Family of Specifications[5] includes a set of specifications, which together standardize a widget as a whole. PLE widgets at TU Graz implement a part of these specifications. "Widgets packaging and configuration" as well as "Widgets Interface" are two necessary specifications to run a widget on the PLE. In order to realize an automatic conversion of W3C browser based widgets to desktop widgets these two specifications are taken into consideration.

"Widgets packaging and configuration" specification[6] standardizes a .zip-packaging format that includes some obligatory and none-obligatory elements. It describes also how internationalization and localization must be applied within the packaging format. The widget folder (.zip file) must contain two files at its root: A start file i.e. "index.html" and a manifest file called "config.xml" which contains all the metadata needed to initialize and run the widget on the PLE. The widget folder may contain non-required files for JavaScript, CSS, or images. "Widgets Interface" specification[7] defines an API for the widgets functionality. It describes the access to the metadata defined in a widgets configuration document, as well as persistently storing data and to receive events related to changes in the view state of a widget. The most used

[5] http://www.w3.org/2008/webapps/wiki/WidgetSpecs (last visit: 2012-08-30)
[6] http://www.w3.org/TR/widgets/(last visit: 2012-08-30)
[7] http://www.w3.org/TR/widgets-apis/(last visit: 2012-08-30)

methods that are used from widgets API are XMLHttpRequests (XHR) to retrieve data from local and remote resources, accessing user preferences (read and write), as well as accessing the default settings in the manifest file. Because of the same origin policy of browsers, widgets have no access to resources on remote domains through XHRs. The PLE provides the widgets with a web-based proxy to bypass this restriction. The widget API provides methods to resize the widgets during runtime, as well as setters- and getters-methods for metadata of widgets (such as title). The described specifications correspond to the actual version of the PLE widget engine. The PLE provides also the widgets with an event based IWC. Widgets can add listeners to events or fire events to notify others. Unicast, multicast, and broadcast communications are supported.

To export PLE widgets to other platforms the two W3C specifications mentioned above have to be considered initially. The packaging and configuration of the widget and the widget interface API have to be adapted to the target platform.

In case of Windows gadgets, the structure of the gadgets is similar to those in PLE widgets, but there are some differences that require a restructuring of the widget. The manifest file must be named "gadget.xml". The metadata contained in widgets' manifest file must be parsed and reformatted to the gadgets manifest file. Additionally, a CSS file called "gadget.css" must be created that defines the sizes of the gadget and the user settings dialog. A file called "gadget.js" must be created too. It sets "settings.html" as the HTML file for the settings dialog of the gadget. Furthermore it sets the events that must be triggered after closing the settings dialog and after the gadget loads completely. After creating these new files, they have to be included into the start file "index.html" of the widget. This is done by referencing "gadget.css" and "gadget.js". The widget API defined in "gadget.js" has to provide the same API that the widgets can access in the PLE. Remote XHRs are possible without a web-based proxy. Local resources within widget package cannot be accessed through XHRs. The Windows Scripting Object (WSO) can be used to access the file system and read the file contents within the widget package instead. It is not possible to set height and title of the widget for Windows gadgets. The title is only used in the gadget selector, and it is not possible to resize the widget dynamically during runtime.

In case of Mac OS dashboard widgets it is different. A minimal dashboard widget on Mac OS platforms requires four files within the widget package: an icon (Icon.png), a background image (Default.png), the manifest file (info.plist) and a HTML based start file. The manifest file contains different metadata about the widget. The HTML UI related to the user settings dialog (the back side of the widget) must be created in a separate "div" element in the body of the start file. The manifest file is called "info.plist"; it is an information property list document. The metadata and settings defined in config.xml of the widget must be parsed and converted into the plist-file. The HMTL structure of the user settings dialog must be generated dynamically with JavaScript and appended to the DOM of the start file while loading the widget. For this purpose the body of the start file has to be divided into two "div" elements, one for the front side and the second for the back side (setting dialog) of the widget. The required JavaScript functionality for this goal is implemented in "widget.js", which must be included into the start file of each widget similarly to the conversion mechanism, described for Windows gadgets. Furthermore a "widget.css" file is needed to position the Mac specific buttons and style the front side and back side of widgets according to the Mac style user guide. The Mac widget API is pretty much the same as those in PLE widgets. Local resources within the widget package can be accessed by XHR methods, as it is the case for the web-based PLE widgets. Remote XHRs are also possible without the need of any web-based proxy. Similarly to Windows gadgets, it is not possible to resize the widgets or change the title on Mac OS dashboard widgets during the runtime.

Table 1 shows the significant differences between three widget containers.

TABLE I.
IMPORTANT DIFFERENCES BETWEEN THREE WIDGET CONTAINERS (PLE, WINDOWS 7 DESKTOP, AND MAC OS DASHBOARD)

	PLE	**Windows 7 Desktop**	**Mac OS Dashboard**
Manifest	config.xml	gadgets.xml	info.plist
Start file	Any name	Any name	Any name
Settings	In manifest	In "settings.html"	In start file
Widget API	widget.xxx	System.Gadget.xxx	widget.xxx
Icon	Any name	Any name	Icon.png
Background image	No	No	Default.png
XHRs (local)	jQuery	ActiveXObject (Scripting.FileSystemObject)	jQuery
XHRs (remote)	jQuery (via Proxy)	jQuery	jQuery

The automatic conversion work is still ongoing and has not been finalized yet. Additional tests are necessary to guarantee the flawless conversion of all existing and future PLE widgets. The desktop widgets have some restrictions. For instance, the event based IWC mechanism fails on desktop widgets and cannot be supported therefore.

The first converter prototype consists of the following three main parts to realize a minimal functioning widget conversion for either Windows Desktop or Mac OS dashboard as widget containers:

- Parsing and adapting the manifest file for each container according to the container's specification.
- Creation of a HTML based dialog for editing the user preferences by the user for each container.
- Handling and adapting the XHR calls according to the specifications of each container.

Fig. 11 shows the "Hangman" widget in the PLE (left), the corresponding converted widgets for Windows 7 Desktop (middle), and Mac OS Dashboard (right). The "Hangman" widget represents the well-known hangman game that is used in the context of game based learning.

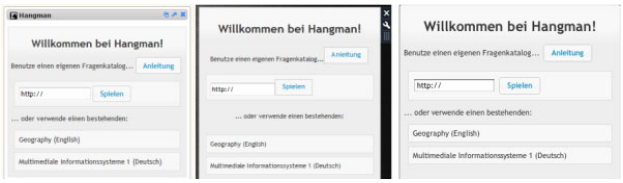

Figure 11. "Hangman" as PLE widget (left), Windows gadget (middle), and Mac OS widget (right)

III. PLE STATISTICS

In order to improve the PLE we needed to consider different parameters that influence the attractiveness and effectiveness of the whole system in general as well as individual widgets. To meet this goal a tracking module was implemented to measure quantitatively how often the widgets are used and by how many users. The measurement was operationalized by means of tracking individual and overall usage of widgets. In order to measure the usage of widgets a hidden module in the background tracked the users' active widgets.

Currently there are about 4000 registered users on the system. The analyzed track data are purely quantitative. From the number of users, who have installed and actively used a certain widget, we are able to determine the top 10 mostly used widgets out of the 75 provided. The first top 6 widgets are related to TU Graz services:

- ZID News (actual news published by the Information Technology Services (ITS) at TU Graz), over 2000 installations
- TUGMail (student e-Mail), over 1000 installations
- TUGraz online (administration and search platform for institutes, courses, and teachers at TU Graz), over 1000 installations
- Pruefungsplaner (a widget to search for examination dates and to plan an individual schedule for doing exams), over 200 installations
- TeachCenter courses (a widget representing the LMS of TU Graz), over 1000 installations
- TUGraz Newsgroups (to read and post threads in newsgroups), over 1000 installations

Within the top10 we find further:

- Weather forecast, over 200 installations
- Culture calendar of Graz, over 200 installations
- Google maps, over 200 installations and
- Widget development guide (a widget that provides some instructions for the users who are interested to enhance the PLE by developing widgets), over 100 installations.

From an educational point of view these students' choices make perfectly sense as these TU Graz services are well known and frequently used even without the PLE.

Translation services (dict.leo.org and dicht.cc), TU Graz calendar, and Facebook widgets follow the top 10 list mentioned above.

Interestingly, the most used widgets are not necessarily the most installed ones. For example, "Pruefungsplaner" widget ranked as the forth most used one is being used by about 200 users actively.

The quantitative data obviously show that the widgets that represent a known or very useful service are used most often. "Pruefungsplaner" widget is not a university service, hence not known to all students, but it can be very useful for students. As a result, it is used very often. On the other hand the widgets that are not really relevant for any specific use case or learning goal are not often used. For example, the "Hangman" widget has been installed by about 200 users, but is ranked on the 29th position in the list of the mostly used widgets.

To let the third party applications analyze the PLE statistics data, the quantitative tracked information have been made public a short time ago using a Twitter channel. As soon as a user uses a widget in PLE a tweet is automatically sent on the PLE Twitter channel[8]. The tweet includes information about the user who has used the widget, the widget, and the date and time the widget has been used. User information in tweets is made anonymous. Up to now more than 1500 tweets can be retrieved from the PLE Twitter channel.

IV. DISCUSSION

The crucial factor of the successfulness of a new service is its attractiveness and usefulness for all potential users. In case of PLE, based on mashup of widgets, a number of initial widgets must be provided which are needed by probably all users. For instance, university wide services such as e-Mails, LMS, social networks such as Twitter and Facebook etc. are used daily by a majority of students. User centered development is the key strategy before launching a PLE.

One of the main qualities of Web 2.0 applications is that the users are not only the consumers but also the contributors. Similarly a further concept of PLE is that the users with basic knowledge in web programming should be able to enhance the framework. Users are not only consumers or contributors, but can also be active developers. After a short introduction users can develop their own widgets and share them with others. Users come across the missing resources and applications that are required by or can be useful for them more probably than the administrators. They develop widgets of their interest and enhance the PLE thereby. The students have developed the majority of the widgets running on the PLE. The number of widgets is increasing each year.

In order to make the PLE more attractive to users, the UI is built pretty much resembling to the mobile apps environments. Despite the UI, students are allowed to share their own developed widgets within the PLE. The widgets must not be necessarily associated with learning. The "Weather forecast" and "Minesweeper" widgets are two examples of such widgets that follow the so-called fun theory: the funnier the environment is, the more often it will be used.

The overall new design concept, web view as well as mobile view, strongly relies on common and approved design structures. For example, the provided spaces are similar to the space concept of Apple's Operating System (Mac OS). Furthermore each widget can be compared with an application, shortly called app, on common mobile phones; too the widget store strongly reminds of app stores (iTunes and Android Market). First usability studies

[8] http://twitter.com/PLETUGraz (last visit: 2012-08-30)

underline the intuitionally character of the system; users are able to work with the environment within seconds without any further explanation.

A further development of the PLE is the introduction of shared spaces. Each space will be shareable with any other user. Shared spaces will allow collaboration between teachers and learners as well as learners and learners. The most interesting didactical approach will not only be the freedom of course design and integration of any resource, but also the equality of user rights and possibilities between teachers and learners within shared spaces.

V. CONCLUSION

The resemblance of the PLE UI to mobile app stores has attracted the users a lot. The new version of the PLE web interface was launched in October 2011. Since then the number of active users have increased to over 200 users per day. That is an increase of 400% in comparison to the former version. It must be mentioned that these figures are still not satisfying much. Once the needs and interests of the majority of our students are covered by appropriate widgets, we can hope to state a high increase of daily active users. To meet this goal, a lot of more learning objects and widgets need to be developed for each major field of study. We hope to provide more PLE widgets through the recently new extension, namely support for Wookie widgets and OpenSocial gadgets.

Spaces help learners to classify their learning resources (widgets) as they are used to in their learning environments in real world. It is planned to provide the possibility to forward spaces as a bundle of widgets to another user. In this way for instance a teacher can create a space as a bundle of widgets that may be interesting for a specific course and forward it to the course´s students.

Furthermore, it is possible to embed widgets through an iframe in any web page. Similarly to web applications such as Slideshare or YouTube, users can copy the iframe code from the PLE and paste it into their own personal web page. In this way PLE widgets can be integrated into the user's own personal web interface, where the user is actually most often online. Learners do not have to enter the PLE to use widgets, they can use the widgets wherever they like.

The opening of PLE statistics data through the PLE Twitter channel provides the possibility for the analysis of usage data by any client.

We believe that the efforts we have already done will take us to a Ubiquitous PLE (UPLE) that can be used anywhere (within the PLE itself or in user's personal web page), anyhow (mobile, desktop, browser-based), and on anytime by each learner in a non-collaborative and (in future) collaborative way. In the near future, learners can decide on their own which widgets to use on which platform in order to support their daily learning activities. Making the widgets platform independent supports the required flexibility that is often desired by users.

Once the number of widgets increases, a recommender mechanism would be necessary to provide as the next step. Furthermore practical experiences must be gathered with the new version. Additionally it must be evaluated how learning and teaching can occur in this novel learning environment.

REFERENCES

[1] T. O'Reilly, Web 2.0 (2006) Stuck on a name or hooked on value? Dr Dobbs Journal, 31, 7, p. 10.

[2] S. Downes, E-learning 2.0. ACM e-Learn Magazine, 2005, 10.

[3] J. Farmer, A. Bartlett-Bragg, Blogs @ anywhere: High fidelity online communication. In: Proceeding of ASCILITE 2005: Balance, Fidelity, Mobility: maintaining the momentum?, 2005, pp. 197--203.

[4] N. Towned, Podcasting in Higher Education, Media Onlinefocus 22, British Universities Film & Video Council, 2005, http://www.bufvc.ac.uk/publications/mediaonlineissues/moF22_vf61.pdf

[5] M. Ebner, H. Maurer, Can Microblogs and Weblogs change traditional scientific writing?. In: Proceedings of E-Learn 2008, pp. 768--776. Las Vegas.

[6] N. Augar, R. Raitman, W. Zhou, Teaching and learning online with wikis. In: Atkinson, R., McBeath, C., Jonas-Dwyer, D., Phillips, R. (eds.) Beyond the comfort zone: In: Proceedings of the 21st ASCILITE Conference, December 5-8, 2004, pp. 95--104. Perth, Australia

[7] A. Kukulska-Hulme, J. Traxler, Mobile teaching and learning. In A. Kukulska-Hulme & J. Traxler (Eds.), Mobile learning – a handbook for educators and trainers, 2005, pp. 25-44, London-NewYork: Routledge.

[8] M. Ebner, N. Scerbakov, C. Stickel, H. Maurer, Mobile Information Access in Higher Education. Proceeding of E-Learn 2008 (2008), p. 77-782 E-Learn – World Conference on E-Learning in Corporate, Government, Healthcare, & Higher Education.

[9] A. Holzinger, A. K. Nischelwitzer, M. D. Kickmeier-Rust, (2006) Pervasive E-Education supports Life Long Learning: Some Examples of X-Media Learning Objects, http://www.wccee2006.org/papers/445.pdf

[10] R. Klamma, M. A. Chatti, E. Duval, H. Hummel, E. T. Hvannberg, M. Kravcik, E. Law, A. Naeve, P. Scott, (2007) Social software for life-long learning. Educational Technology & Society, 10, 3, pp. 72-83

[11] M. Ebner, W. Nagler, Has Web2.0 Reached the Educated Top?. World Conference on Educational Multimedia, Hypermedia and Telecommunications; 2010 (2010), p. 4001-4010

[12] R. Tuchinda, P. Szekely, C.A Knoblock, Building Mashups by example. In: Proceedings of the 13th international conference on intelligent user interfaces. ACM, 2008, Gran Canaria, Spain.

[13] S. Schaffert, M. Kalz, Persoenliche Lernumgebungen: Grundlagen, Moeglichkeiten und Herausforderungen eines neuen Konzepts. In K. Wilbers & A. Hohenstein (eds.), Handbuch E-Learning. Expertenwissen aus Wissenschaft und Praxis - Strategien, Instrumente, Fallstudien. (Gruppe 5, Nr. 5.16, pp. 1-24). 2009, Koeln, Deutscher Wirtschaftsdienst (Wolters Kluwer Deutschland).

[14] F. Wild, F. Moedritscher, S. Sigurdason, Designing for Change: Mash-Up Personal Learning Environments, 2008.

[15] B. Taraghi, M. Ebner, G. Till, H. Mühlburger, Personal Learning Environment – A Conceptual Study, International Conference on Interactive Computer Aided Learning (ICL 2009), Villach, Austria.

[16] M. Ebner, B. Taraghi, Personal Learning Environment for Higher Education – A First Prototype. In: World Conference on Educational Multimedia, Hypermedia and Telecommunications, 2010, p. 1158 – 1166.

[17] B. Taraghi, M. Ebner, S. Schaffert, Personal Learning Environments for Higher Education: A Mashup Based Widget Concept. In Proceedings of the Second International Workshop on Mashup Personal Learning Environments (MUPPLE09), 2009, In conjunction with the 4th European Conference on Technology-Enhanced Learning (ECTEL'09).

[18] G. Zhan, Q. Jin, Research on Collaborative Service Solution in Ubiquitous Learning Environment, 6th International Conference on Parallel and Distributed Computing. Applications and Technologies (PDCAT'05), 2005, 804-806.

[19] B. Taraghi, M. Ebner, A Simple MVC Framework for Widget Development. In: Proceedings of the 3rd International Workshop on Mashup Personal Learning Environments (MUPPLE10), 2010, In conjunction with the 5th European Conference on Technology-Enhanced Learning (ECTEL'10).

AUTHOR

Behnam Taraghi is a junior researcher and a PHD student at department of Social Learning – Information Technology Services at Graz University of Technology, Münzgrabenstrasse 35A, A-8010 Graz, Austria (e-mail: b.taraghi@tugraz.at).

Manuscript received 17 October 2012. Published as resubmitted by the author 7 November 2012.