Collaborative Development of a PLE for Language Learning

doi:10.3991/ijet.v5s1.1196

D. Renzel[1], C. Höbelt[2], D. Dahrendorf[2], M. Friedrich[3], F. Mödritscher[4],
K. Verbert[5], S. Govaerts[5], M. Palmér[6] and E. Bogdanov[7]
[1] RWTH Aachen University, Aachen, Germany
[2] imc information multimedia communication AG, Saarbrücken
[3] Fraunhofer Institute for Applied Information Technology, St. Augustin
[4] Vienna University of Economics and Business, Vienna, Austria
[5] Katholieke Universiteit Leuven, Leuven, Belgium
[6] Uppsala University, Uppsala, Sweden
[7] École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland

***Abstract*—This paper provides a report on the experimental collaborative and distributed development of a prototypic Widget-based PLE. The development process is described and detailed taking into account the requirements of a language learning scenario. First results are presented, and developer experiences are discussed critically with a focus on the development process as well as problems with current Widget technologies and interoperability.**

***Index Terms*—Collaborative Software Development, Technology Enhanced Learning, Personal Learning Environment, Widget Interoperability.**

I. INTRODUCTION

Current endeavors in the domain of *Technology Enhanced Learning (TEL)* exhibit the need for increased openness and responsiveness of current learning environments. While older learning technology generations were often central and closed systems, often merely focusing on the management of learning processes, the next generation of Personal Learning Environments (PLEs) tackled by the ROLE project [1] concentrates on a highly distributed approach drawing on the combination of established open-standard Web technologies in order to enable the learner-side integration of services and tools from a plethora of heterogeneous sources into customized learning environments. One of the major goals of ROLE is to deliver an appropriate technical infrastructure for the establishment of such responsive open learning environments. Another goal of the project is to establish a community of open source and external developers outside the consortium contributing further tools and services based on this infrastructure. During the first project developer meeting, we thus agreed to work in parallel to the standard project plan towards a common goal, called the "Christmas Project" with the following objectives in mind:

- Create a first demonstrator of ROLE to visualize the potential of the project
- Enable the consortium to better define needs and derive technical specifications
- Experiment with promising combinations of Web technologies towards an integration infrastructure for PLEs
- Explore the feasibility of a collaborative and distributed development process scalable to a large community of independent developers

While the first two objectives addressed the consortium-internal collaboration, the last two clearly address a broader audience. Thus, this paper reports on the results of the Christmas Project with a focus on the collaborative distributed development process and a first integrated PLE prototype resulting from this process. It gives the reader an insight into the challenges we faced during our work regarding the development process and the technologies we experimented with. After drawing the conclusion that current technology and development processes are often still insufficient for a seamless independent development of interoperating learning services and tools, it outlines possible improvements.

The rest of this document is structured as follows. In Section II we describe the development process to give an insight of how our work was organized. In Section III we present details on the requirements elicited for an integrated PLE based upon a language learning scenario. In Section IV we present the individual partner contributions in more detail. Section V presents our experiences and a critical conclusion of our work regarding current issues regarding Widget technology. In Section VI we end with a short summary and give a short outlook to further work.

II. DEVELOPMENT PROCESS

Following the objective of establishing a community-oriented development process, we planned to explore such a process in a smaller scale within the consortium starting off with a community of nine partners across Europe, from both industry and academia and with different degrees of technical background. Since heavy-weight processes would in practice not be feasible and accepted with a large-scale developer community, we decided to keep the process as light-weight as possible, however borrowing concepts from standard processes such as Agile Development [1][2], e.g. short iteration cycles, shared code & documentation, continuous integration, regular developer

[1] http://www.role-project.eu/

communication, etc. However, given by the spatial distribution of the community, concepts requiring physical attendance emphasized in Agile approaches had to be replaced by communication technology in order to avoid roundtrip unrealistic in a larger scale community. Furthermore, such a distributed approach requires technical means for code and documentation sharing and an integration environment with low entry barriers. Table I briefly shows the rather light-weight abstract schedule we pursued during our experiments. It should be noted that this process can be iterated. However, the schedule shown here is likely to be subject to refinement or even replacement in next phases of the ROLE project.

In the following, we provide details on the first iteration conducted during the ROLE Christmas Project.

All of the participants dedicated themselves to contribute components for an integration framework, individual learning services, either implemented or as mockup to reach the common goal of delivering an integrated PLE prototype.

After the collection of all contributions intended by the partners, we sketched the ROLE Christmas Project Big Picture (cf. Figure 1). Given by the heterogeneity of the partners' plans for contributions, we agreed on a common scenario serving as a concrete use case for a ROLE PLE prototype based on a Widget approach. For that purpose we chose a language learning scenario described in detail in Section III.

As a basis for ongoing documentation we decided to setup a document to be edited collaboratively by all partners, starting with the Big Picture, an elaborate description of the scenario and a time plan. Every partner added a description of his contribution and how it would fit with the scenario. Thereby, we did not require a perfect match, but at least a high degree of relevance.

A ROLE XMPP Server was setup for direct communication. A ROLE developer chat room was configured to log all group conversations on the server side. Thus, everybody could easily keep track on previous discussions, which turned out to be a helpful feature. However, restrictive firewall policies enforced by various partner institutions sometimes hindered the use of XMPP – a valuable experience for future considerations regarding its use in our software (cf. Section IV.E.)

TABLE I.
ABSTRACT PROCESS SCHEDULE

Planning Phase (collaborative)
• Start with of story-based use case scenario • Extract (non-)functional requirements • Identify components • Structure & categorize components • Identify interfacing components • Agree on time schedule
Development Phase (distributed & independent)
• Develop & document components • Use own development environment • Communicate with other developers • Continuously share code & documentation • Continuously run latest version of components in integration environment

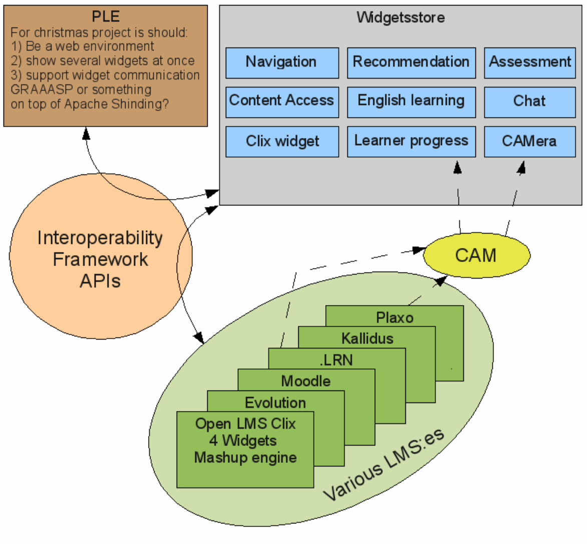


Figure 1. ROLE Christmas Project Big Picture

Furthermore, in order to maintain the source code of all partner contributions, we agreed on utilizing a git-based [3] repository at github.com for reasons of wide visibility and acceptance in the open source developer community. Besides SCM functionality, github provides an issue tracker, Wikis, repository statistics, etc. All of these features were frequently used during the development phase.

With regard to a development environment for the individual partner contributions, we defined the following simple policy:

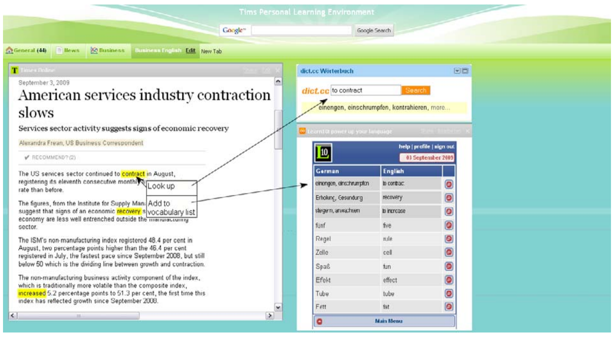
1. All partners setup development environments.
2. One partner maintains integration environment and regularly pulls from the ROLE github.

During a later physical meeting we discussed a few options and suggestions regarding the choice of technologies as a basis for development and integration environments. The following considerations were taken into account:

1. How to quickly move forward and succeed with Christmas Project on schedule.
2. Avoid using technologies that will hinder us or force us to start over completely later on.
3. Make as few decisions as possible at this point in time.

Since the prototype should be Widget-based, we discussed a pre-selection of promising technologies for software components such as Widget engines, containers, stores and repositories, inter-Widget communication mechanisms, protocols, etc. and started our developments after a first decision for one configuration.

All components of the software we used as technical foundation for our work promised to be platform and browser independent – unfortunately, this assumption was far from being fulfilled. It turned out that all regarded alternatives were still in an early experimental development stage. The consequence was an increased communication overhead among the partners for the purpose of finding, agreeing on and changing to more acceptable solutions. Due to the fact that we could never rely upon the software components of our development and integration environments, continuous integration was hardly possible, leading to the usual longer and error-prone final integration phase shortly before the prototype delivery deadline. The details of the technical problems we faced will be discussed in Section V. In the next section we present detailed requirements for the ROLE Christmas Project.



Tims Personal Learning Environment

September 3, 2009

American services industry contraction slows

Services sector activity suggests signs of economic recovery

Alexandra Frean, US Business Correspondent

✔ RECOMMEND? (2)

The US services sector continued to contract in August, registering its eleventh consecutive monthly ... rate than before.

The figures, from the Institute for Supply Man... suggest that signs of an economic recovery s... economy are less well entrenched outside the manufacturing sector.

The ISM's non-manufacturing index registered 48.4 per cent in August, two percentage points higher than the 46.4 per cent registered in July, the fastest pace since September 2008, but still below 50 which is the dividing line between growth and contraction.

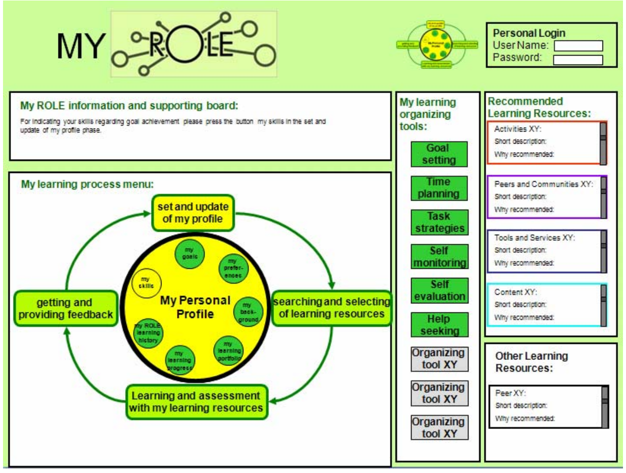
The non-manufacturing business activity component of the index, which is traditionally more volatile than the composite index, increased 5.2 percentage points to 51.3 per cent, the first time this index has reflected growth since September 2008.

to contract

einengen, einschrumpfen, kontrahieren, more...

German	English
einengen, einschrumpfen	to contract
Erholung, Gesundung	recovery
steigern, anwachsen	to increase
fünf	five
Regel	rule
Zelle	cell
Spaß	fun
Effekt	effect
Tube	tube
Fett	fat

Figure 2. PLE with three Learning Widgets



For indicating your skills regarding goal achievement please press the button my skills in the set and update of my profile phase.

Figure 3. Learner Navigation Tool Mockup

III. REQUIREMENTS

In this section we will present an overview of the requirements elicited for the realization of the Christmas Project. We start with the language learning scenario we agreed upon as well as the underlying psycho-pedagogical model. We then continue with technical requirements for the realization of our prototype and considerations on how to fulfill them.

A. Scenario

In our language learning scenario, the learner, Tim, is an employee at Travel Books that sells books and videos on travel destinations. He works in the sales department and has to go to international fairs and to speak with distributors, bookshops and other business partners. As most business communication is in English, Tim needs to improve his English skills, especially in Business English.

One part of his learning strategy is to read texts and to learn its vocabulary using his PLE. For that purpose he adds three widgets: a *Language Resource Browser*, a *Vocabulary Trainer* and a *Translator* widget. All of them are visible on one webpage.

In the Language Resource Browser, Tim searches for a text and starts reading it. Each time he misses a word he selects it and opens a context menu on it. The system then proposes him to either look it up in the Translator widget or send it to the Vocabulary Trainer widget (cf. Figure 2).

So he adds words that he considers as important to the Vocabulary Trainer and others he only looks up.

After reading the text, he has gathered a list of words he considers important to be repeated in future using the Vocabulary Trainer widget.

In the next days he continues reading newspaper articles regularly and his Vocabulary Trainer widget obtains more and more words. In an analogous manner he uses the Language Resource Browser widget to work with other media types such as audio or video.

One day, he is learning with his Vocabulary Trainer, memorizing the words on the list and testing whether he knows them sufficiently well. He recognizes that he has problems to remember a certain word because he does not know the context anymore where it originally appeared.

Fortunately the Vocabulary Trainer always stores the link to the original text. So Tim clicks on the word and the original text appears in the Language Resource Browser widget and shows the sentence where the word was taken from.

Reading the sentence and thinking of the context facilitates him to memorize the vocabulary. Furthermore he improves his language proficiency by knowing situations where he can use the word.

The scenario can be extended by a group of learners, e.g. students that participate in an English Language course. The instructor sends them a list of newspaper articles that are available online. The students are asked to read and analyse them and to learn the vocabulary. As they come from the same background (high school English level), they decide to jointly create a vocabulary list using the Vocabulary Trainer widget. Whenever a student finds an unknown word in the newspaper article, he adds it to the joint vocabulary list. The Vocabulary Trainer widget will also display words added by other students who have the same level in the English language. Words that have been added more often are sorted higher. Each learner individually trains the list of words and the vocabulary widget keeps track of each student's individual vocabulary knowledge. The group's average knowledge (number of words learned) is displayed in the widget, together with the current student's knowledge. Nevertheless it should be possible to deactivate the group functionality if the learner only wants to learn for herself.

B. Psycho Pedagogical Model

In this section we shortly present the ROLE psycho-pedagogical model [4]. The central element of this model is a cyclic learning process model consisting of several learning phases related to learning activities. Following the connections between learning phases, activities and tools, a sequence of learning tools can be derived. Furthermore, two different kinds of learning tools are identified, first "normal" learning tools conveying domain knowledge, and second, meta-learning tools used for self-regulating the own learning process.

A navigation tool guiding the learner through a self-regulated learning process by recommending activities and tools was proposed. The partners created an interactive mock-up of such a navigation tool (cf. Figure 3) and presented it in the context of the scenario from the previous section.

The contribution of education professionals had a great impact on the technical developments and showed us once more that technical and conceptual work should be conducted hand-in-hand.

C. *Technical Requirements*

For the technical realization of a first ROLE PLE prototype, we decided to follow an approach of intercommunicating Widgets. The expectations from choosing such an approach were a relatively loose coupling between individual widgets contributed from different partners. It should be noted that we also experimented with such an approach, because in future project stages, developers outside the consortium should be enabled to work completely independent from other developers.

First, a technical infrastructure was needed as a basis for our prototype. Starting from the Big Picture, we identified requirements for the following components of such an infrastructure:

1. Widget Container/Engine
2. Widget Store/Repository
3. Widget Interoperability Mechanisms
4. Widget User Interface

As promising Widget Container/Engines, we considered experiments with Apache Shindig [5], the reference implementation of an OpenSocial [6] container currently under incubation at Apache. OpenSocial defines a common API for social applications across multiple web sites and also includes a specification for widgets – gadgets in OpenSocial terminology. As further alternative we considered SocialSite [7], based on Glassfish and Apache Shindig with the ability to run OpenSocial gadgets and have them backed by the same social graph. Furthermore, Apache Wookie [8] was taken into consideration as a solution for adding W3C Widgets [9] as well as OpenSocial and Google (Wave) Gadgets to web applications.

1) As solutions for a Widget store/repository, we considered the following four solutions:

- No store/repository, fixed list.
- Wookie
- Google Gadget directory
- ROLE widget store (remains to be built)

A fundamental requirement to Widget interoperability was the support of inter-widget communication. There should be no major configuration needed on the side of end users assembling widgets. Furthermore, it should be easy to build containers for the chosen widget technology. Integration into existing systems, like LMS should be possible. Preferably, standard containers should also support these technologies. The following technologies were taken into closer consideration as potential candidates:

- Gadget pubsub [6]
- OpenAjax Alliance hub2.0 pubsub [10]
- Open Application (draft) [11]
- XMPP [12],[13] XEP-060 Publish/Subscribe [14]
- HTML5 DnD[16]

2) Regarding a Widget user interface a couple of different approaches were considered, e.g. portal pages such as iGoogle, Wiki or LMS approaches, etc. However, we decided to keep things simple first using a fixed HTML page. An additional solution experimented with was the integration in Graaasp (*http://graaasp.epfl.ch*), a Web 2.0 contextual aggregator of people, spaces, assets and tools, in order to be able to make use of its built-in mechanisms for sharing, commenting, and recommendation.

For the first experiments, we decided to use a configuration of SocialSite as a container, no widget store, simple HTML page or Graaasp as user interface and OpenApplication based on Gadget pubsub for inter-widget communication. Furthermore, the previously described ROLE XMPP Server was used for experiments on remote inter-widget communication. For this first prototype, the primary focus was thus put on the development of communicating widgets.

IV. PARTNER CONTRIBUTIONS

In this section we will present the contributions of all partners for the Christmas Project prototype. The first two contributions are targeting at an integration framework focus on the communication between widgets. All remaining contributions consist of different types of widgets either especially for the language learning scenario or with rather general functionality.

A. *Inter-Widget Communication*

Uppsala University introduced the Open Application Event API to provide a generic solution to the requirement of inter-widget communication in the scenario. The most important aspect of the solution is that widgets need not be "hard-wired" against each other. Instead, they communicate using well-known data expressions, with the intention that widgets will understand the parts that are important to them.

The basic principle behind the event API is that all widgets are notified of all events. No specific subscription step is necessary. All widgets are given the opportunity to react to any event, which they may choose to do depending on event type, message type, message content, etc.

1) *Events types*

The event types include: *state, load, modify, save, select, unselect, startDrag* and *stopDrag*. A state event is different from the others in that it has no relevant resource, and therefore no resource is sent along. Instead, a state event indicates a change of state in the widget that sent it.

2) *Event Structure*

Notifications of events are sent out as messages. The message consists of the relevant resource, if any, depending on the event. The message is wrapped in an envelope containing further event information.

event - the event type (see previous section).
type - The message type
message - The message, for example a resource.
uri - The message's URI, if any.
date - Timestamp of when the event occurred **sharing** - How the event is allowed to be used.

At the moment, the event API consists of a Gadgets PubSub channel, in which the messages are published and thereafter sent out to all widgets.

An Open Application compliant widget subscribes to the PubSub channel when the widget is loaded. The sharing property is intended to specify how the data may be used: on the same page, only on the user's machine, by a service under the user's control, by participants with access to the same widget instances, or that it may be transmitted to various services. Each level includes all the privileges of the previous levels.

3) Message Types

A number of message types are defined. These include: namespaced-properties, JDIL, JSON, URL, HTML, XML and MIME content. Of these, namespaced-properties is intended for simple RDF-like metadata with direct properties, and MIME content for unparsed text or binary data.

B. *Web 2.0 Platform for Collaborative Organization of Information and Tools*

The EPFL team developed a Web 2.0 platform, namely Graaasp, to helps users to collaboratively organize information and tools toward a given goal or activity. Tools in Graaasp are implemented as widgets. In addition to the standard add, remove, browse, group and share operations, Graaasp also supports tagging, rating and commenting. The widgets are imported/bought from a widget store and linked to the given Graaasp space dedicated to a learning activity.

Once the activity is configured the user can switch the view to play with instances of selected widgets. The widget instances are rendered in a widget container managed by a Wookie engine. Any widgets following the W3C widget specifications can be instantiated into Graaasp.

Based on both the created activity structure and the user ratings, a recommender system that would contextually recommend widgets to users is being developed. Similarly, Trust and Reputation algorithms for widgets are also considered.

C. *Monitoring of User Behaviour*

To provide recommendation and self-evaluation mechanisms Fraunhofer FIT developed a CAM [17] widget to unobtrusively monitor user behavior. Other widgets, like the vocabulary trainer widget (cf. next section), trigger events on different user actions which are then broadcasted to other widgets using Open Application. The CAM schema provides a standardized data format to store user activities and thus fosters widget interoperability since every widget could access these data.

Since the CAM widget is a simple subscriber widget, it listens to every event published from any other widget and collects them. The collected events are then transferred into the CAM schema and afterwards stored in a database. As all control should be with the user, she can decide between different storage modes. For the Christmas project we therefore specified three different storage modes which can be selected by the user (cf. Figure 5).

If the user decides to store her activities remotely, the CAM widget transfers the data to a central CAM repository, where all events of every user are stored. After creating a CAM instance, the CAM widget calls a Web service [18] passing the CAM information which is then stored in a database. The *local* storage mode uses the Gears plug-in [19] to store CAM information in a local database. The Gears plug-in is available for many platforms and supports all common browsers. It provides a SQLite [20] interface to easily create a database and store information inside of the local browser profile. An alternative approach to Gears is using HTML5 [16]. Since the specification of HTML5 is not finished yet and is currently not supported by every browser we have chosen Gears. If the user does not want her usage behavior stored, she can choose the storage mode *off*.

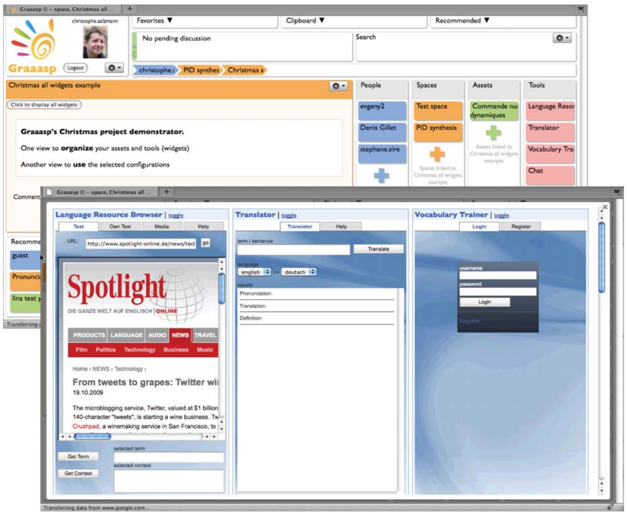
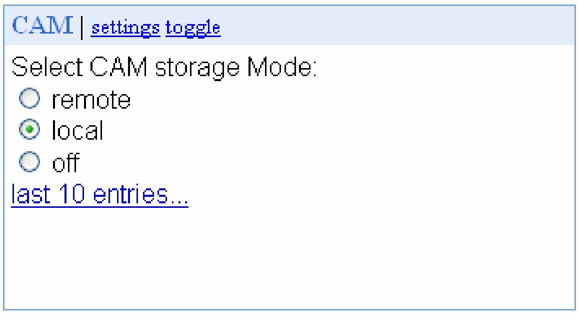


Figure 4. Graaasp views to organize the activity (on the back) and to play the instantiated widgets (on the front)



CAM | settings toggle

Select CAM storage Mode:

○ remote
● local
○ off

last 10 entries...

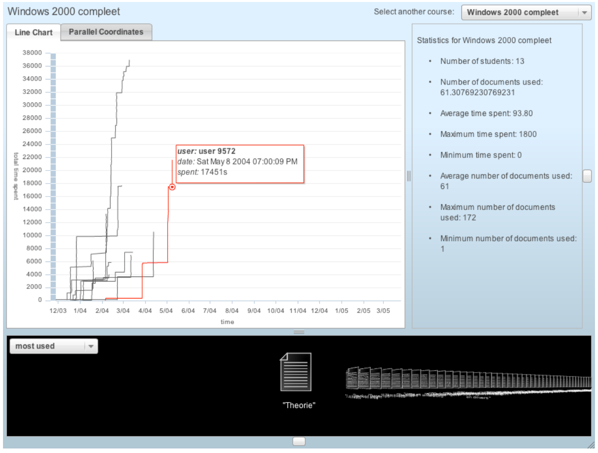
Figure 5. The CAM widget with three different storage modes

To generate recommendations based on CAM, these can either be based on the user's own previous behavior or the usage history of others can be taken into account as well. The different storage modes have effect on the generation of recommendations. In the *remote* mode the user can get recommendations, but also allows the system to use his information to generate recommendations to other users. The *local* storage mode only allows retrieving recommendations, and disallows the system to use her information to generate recommendations to other users. If the user does not want her usage behavior to be monitored she can neither get recommendation nor support the system to generate recommendations to other users.

As we had no Vocabulary Trainer CAM data for the Christmas project, we transferred some PLE Monitoring data [21] into the CAM schema and provided a separate Web service which offers methods to generate recommendation and self evaluation statistics.

D. *Visualization of Monitored Activity*

KU Leuven developed a dashboard that enables students and teachers to monitor learning activities. In the dashboard students, can monitor the progress they made



Windows 2000 compleet

Select another course: Windows 2000 compleet

user: user 9572
date: Sat May 8 2004 07:00:09 PM
spent: 17451s

Statistics for Windows 2000 compleet

- Number of students: 13
- Number of documents used: 61.30769230769231
- Average time spent: 93.80
- Maximum time spent: 1800
- Minimum time spent: 0
- Average number of documents used: 61
- Maximum number of documents used: 172
- Minimum number of documents used: 1

Figure 6. Dashboard with line chart

on a certain course or task and compare themselves with other students working on the same task.

An important feature of the dashboard is learning material recommendation. Based on the learning material other students have used, who have progressed further, we can recommend interesting learning material to the student. A student can compare his activities with other students and a teacher can get a general overview of what is going on in the course, see if it meets expectations and detect potential problems.

Figure 9 illustrates the first version of the user interface for our application. The student can select the course and will be presented with 2 different charts, a course analytics overview and document recommendations. Every line in the chart in figure 1 is a student. The chart shows when the student worked (horizontal axis) and how long he worked on the course (vertical axis). The red line shows a selected student, we see that the student was very late in finishing the course and that he spent a lot of work in a small number of sessions. This view enables a student to compare his progress with that of his fellow students. Another visualization uses parallel coordinates [22]. It shows a set of metrics on parallel axes. A student is represented as a polyline with the vertices on every axis. The metrics are: the average and total time spent on the course, the number of documents used and the average time of the day that a student works. By visualizing these metrics next to each other one can grasp another view on the course activity and discover trends.

The widget is developed in Adobe Flex. To monitor user activities, we use CAM [17]. In order to test the tools, we used course data provided by U&I Learning [21]. Together with our partners at FIT, we experimented with the data to propose possible metrics and collaborated on a Web service providing methods to retrieve and calculate: a list of all courses, a list of recommended documents for a course, general statistics for a course, statistics of a student of a course and the student's attention metadata for a course. The widget can be easily deployed on top of another Web service that uses different attention metadata to provide the same statistics.

In a teacher modus, the student names may need to be anonymized. Privacy is an important issue when monitoring this kind of data. For the Christmas project, we want to simply anonymize the names of the students in the teacher view. This is not implemented yet, because the data from U&I was already anonymized.

Another design idea, not yet implemented, is a graph-based community visualization widget. This tool could allow students to find fellow students how have the same language proficiency as them to chat or collaborate with. The widget would communicate with the chat module of RWTH to provide chat functionality. We are currently implementing this widget.

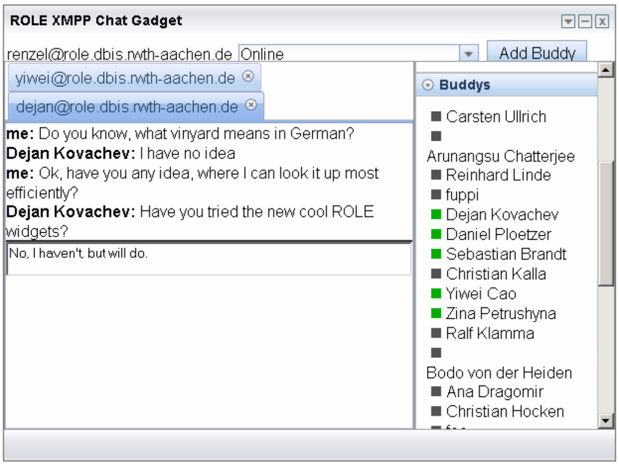
E. XMPP Chat Widget

The contribution of RWTH Aachen University is a Chat Widget providing a simple Instant Messaging (IM) client (cf. Figure 7) based on the XMPP [12][13] protocol. The widget offers interface elements for 1-on-1 conversations, the management of buddy lists and user presence information. In its default configuration, the widget connects to the ROLE XMPP Server. However, connections to arbitrary XMPP Servers are possible. Regarding the ROLE Christmas Project's language learning scenario, the XMPP Chat Widget contributes to synchronous communication between learners integrated in a PLE. The widget is as a rich source for communication events, e.g. updated presence information, incoming/outgoing message, etc. The integration of event publishing into the Open Application approach to be captured by CAM is planned for the near future.

Besides the added value of XMPP chats between learners and basic communication statistics, these two widgets demonstrate how inter-widget communication could be realized between remote widgets (via sending messages over XMPP) and between local widgets (e.g. via a Gadget pubsub). There already exist specifications on XMPP Extension Protocols (XEPs) for Publish/Subscribe [14] or Personal Eventing Protocol [15] mechanisms, which will be considered for a seamless remote/local inter-widget communication for future developments.

F. Language Learning Widgets

In order to fulfill the requirements of the language learning scenario, imc AG developed the English learning widgets: The Language Resource Browser widget, the Vocabulary Trainer widget and the Translator widget. These widgets demonstrate a reasonable use of the Inter-widget communication by sending and receiving term items described by a term and its context and source.



ROLE XMPP Chat Gadget

renzel@role.dbis.rwth-aachen.de Online

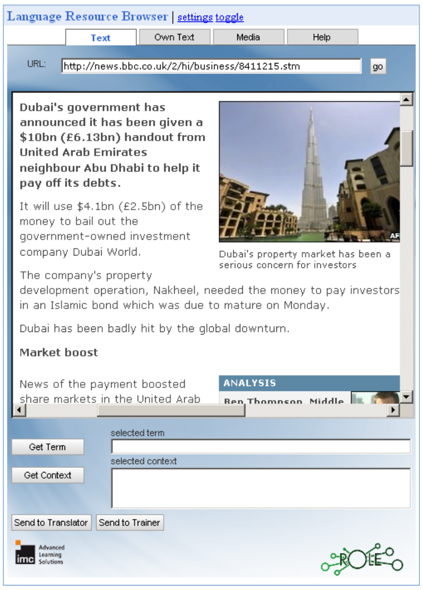
yiwei@role.dbis.rwth-aachen.de
dejan@role.dbis.rwth-aachen.de

me: Do you know, what vinyard means in German?
Dejan Kovachev: I have no idea
me: Ok, have you any idea, where I can look it up most efficiently?
Dejan Kovachev: Have you tried the new cool ROLE widgets?

No, I haven't, but will do.

■ Carsten Ullrich
■ Arunangsu Chatterjee
■ Reinhard Linde
■ fuppi
🟩 Dejan Kovachev
🟩 Daniel Ploetzer
🟩 Sebastian Brandt
■ Christian Kalla
🟩 Yiwei Cao
🟩 Zina Petrushyna
■ Ralf Klamma
■ Bodo von der Heiden
■ Ana Dragomir
■ Christian Hocken

Figure 7. ROLE XMPP Chat Widget



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

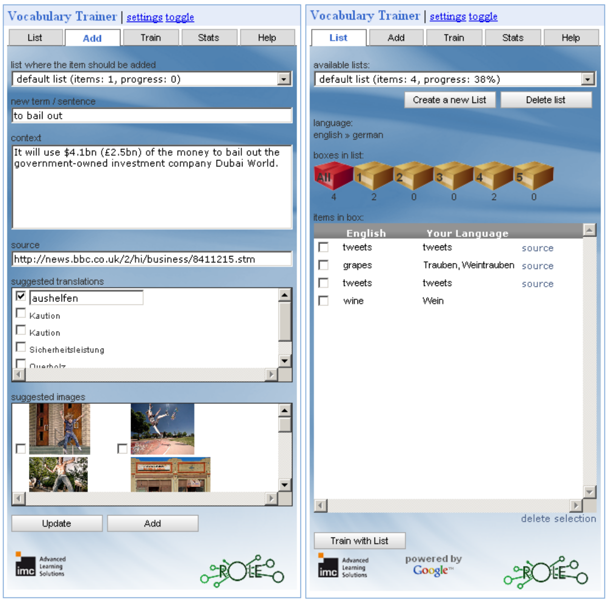
ANALYSIS

Ben Thompson, Middle

selected term

selected context

Figure 8. Language Resource Browser Widget



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

☑ aushelfen
□ Kaution
□ Kaution
□ Sicherheitsleistung
□ Querholz

suggested images

available lists: default list (items: 4, progress: 38%)

language: english » german

boxes in list:

All	1	2	3	4	5
4	2	0	0	2	0

items in box:

	English	Your Language	
□	tweets	tweets	source
□	grapes	Trauben, Weintrauben	source
□	tweets	tweets	source
□	wine	Wein	

Figure 9. Vocabulary Trainer Widget

1) Language Resource Browser Widget

The Language Resource Browser widget (cf. Figure 8) allows user to consume media and send term items to other widgets processing the information. Examples of such widgets are the Translator Widget where the term will be translated or the Vocabulary Trainer where the user can add this term to a vocabulary list. At the moment the widget offers three different tabs. The “Text” tab works like a web browser. It displays a page to a given URL in an iframe where the user can select the term and context. The source of such a term item will be the URL from the page.

In the second tab "Own Text" the user can add her own text taken from an online or offline resource. The third tab provides support to browse for different media such as video and audio. While watching or listening to the media, the user can enter a term in a field. The source of such a term item will be the URL from the media and the context will be defined as “Media Context”.

2) Translator Widget

The Translator widget allows a user to translate terms or sentences. It translates either a term which was entered from the user or a received term item. We combined different Web services (i.e. Wikipedia, Google Dictionary, DICT.ORG, Google Translate) for the translation process

At the moment only English to German is supported, but the language pool could be extended to all languages supported by the services above.

3) Vocabulary Trainer widget

The Vocabulary Trainer (cf. Figure 9) widget is implementing a slightly modified Leitner system [23]. A vocabulary list consists of five different buckets. If an item is added is will be put in the first bucket. If the user is training a list and knows the right translation the item will be moved to the next bucket and else it will be moved to the previous bucket.

The information is stored on a central server and accessed using REST Web services. Each user has a unique login and authentication is done by basic access authentication over REST. For translation the same Web services are used as in Translator widget, and Flickr is used to suggest pictures for terms. Vocabulary items are stored in a list which can be managed by the user.

The widget has four functionalities represented by four tabs: "Add", "List", "Train" and "Stats".

The "Add" tab allows users to manually insert a new term/sentence, the context of that term and its source. In combination with the Language Resource Browser the sent term item appears automatically in the Vocabulary Trainer widget.

The "List" tab provides an overview of the stored lists and vocabulary items. The user can create/delete lists and inspect the content of the different buckets.

The "Train" tab gives the learner the possibility to practice her stored vocabulary. After choosing a bucket that she wants to train a term from this bucket and its context will be displayed. The user can get help by viewing the source of that item or viewing the image to that item (if there exists one). The fourth tab "Stats" shows statistics of the training. It displays a global score and a score for each list.

G. Federated Search & Language Processing

In traditional educational scenarios, teachers typically provide appropriate learning materials and give feedback on student essays. Vienna University of Economics & Business contributed two widgets which are useful for these purposes and indicate their application for language learning.

ObjectSpot, a widget for federated search of academic papers in different digital libraries, has its origins in the iCamp project (http://www.icamp.eu). This search client allows plugging in different digital repositories, whereby the documents are retrieved via the Simple Query Interface (SQI) [24] standard. The central core of this widget is the ranking algorithm which has been developed over several iterations and mixes in the search results from the repositories on the fly [25]. In practice, this widget is useful for both learners and teachers to retrieve the most appropriate literature for a specific knowledge domain. By default, the most important digital libraries containing academic papers, e.g. ACM, IEEE, Google Scholar, CiteSeer, EBSCO, etc., are included.

The screenshot shows the results for the query term 'open responsive learning environments'. At the top, the search term and the state of the repositories are displayed. At the bottom, the user can navigate through the pages. On the left hand side, a user can 'lock' appropriate results, thus giving explicit relevance feedback and recommendations for others. This mechanism can also be used to export search results through a feed-based API if another tool is attached to ObjectSpot. Pressing the option button, a user can configure SQI-enabled digital repositories for her search client, visualize the location of the repositories on a map, get statistics on the quality of the repositories, export the results as RSS-feed, get recommendations for a query term, or plug another tool to ObjectSpot.

Conceptalyzer comprises a language processing widget which builds upon a LSA-based Web services developed within the LTfLL project (http://www.ltfll-project.org).

This widget analyzes online resources (e.g. Wikipedia articles or RSS feeds) in terms of the concepts behind the text and visualizes them according to their relevance. Application areas of this widget comprise learner positioning, monitoring one's conceptual development. It can be also helpful for teachers in preparing learning materials or grading students [26].

The screenshot shows the result of the analysis of a Wikipedia article, whereby the relevant terms are visualized in the form of a 'concept cloud'. The size of a term indicates its relevance for the article while the color links to the text corpora used to train the LSA function.

V. DISCUSSION OF WIDGET TECHNOLOGIES

During the development process of the ROLE Christmas Prototype PLE, we collected a set of valuable however negative experiences to be shared with other developers working with the technologies we attempted to combine. These experiences will be discussed in this section.

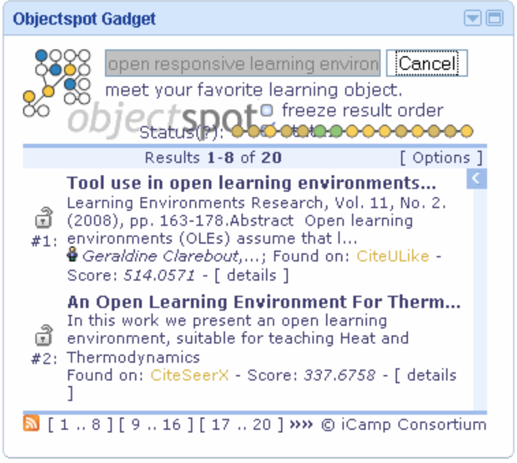
One of the most surprising experiences from the developer perspective was the immature state of many widget technologies, especially with regard to inter-widget communication, one of our main requirements for the language learning PLE. During the development process we tested the following three OpenSocial compliant Gadget development environments:

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

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
- Incompatibility with External Libraries
- Lack of Developer Support

The first problem was related to the installation of a gadget container, in particular with SocialSite. First, the current SocialSite distribution is restricted to specific, already outdated versions of Glassfish and Shindig, and thus is not forward compatible - an essential property, when working with experimental systems. Given the diversity of devices and platforms available to the developers, we quickly had to find out that platform and browser independence was not given at all. Container-side or/and browser-side errors were the result. The most 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. Furthermore, error messages were cryptic and incomprehensible and thus did not provide any hint to the original location of an error. Furthermore, we lost a lot of time communicating possible alternatives. An excursion to the usage of Apache Shindig instead of SocialSite was also not successful for all of us. Further problems were related to the incompatibility of external JavaScript libraries with the Widget container, which again resulted in strange code rewriting effects. Especially with regard to JavaScript library support for XMPP, we had to experience that libraries were not far enough for the realization of our goals and definitely need improvement. Finally, we had to experience that the 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.


Objectspot Gadget

open responsive learning environ [Cancel]

meet your favorite learning object.

objectspot □ freeze result order

Status(?):

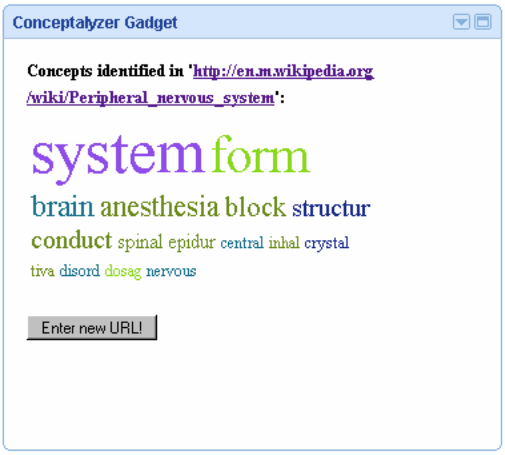
Results **1-8** of **20** [Options]

#1: **Tool use in open learning environments...**
Learning Environments Research, Vol. 11, No. 2. (2008), pp. 163-178.Abstract Open learning environments (OLEs) assume that l...
Geraldine Clarebout,...; Found on: CiteULike - Score: *514.0571* - [details]

#2: **An Open Learning Environment For Therm...**
In this work we present an open learning environment, suitable for teaching Heat and Thermodynamics
Found on: CiteSeerX - Score: *337.6758* - [details]

[1 .. 8] [9 .. 16] [17 .. 20] »»» © iCamp Consortium

Figure 10. Objectspot – Federated Search Widget


Conceptalyzer Gadget

Concepts identified in 'http://en.m.wikipedia.org/wiki/Peripheral_nervous_system':

system form
brain anesthesia block structur
conduct spinal epidur central inhal crystal
tiva disord dosag nervous

[Enter new URL!]

Figure 11. Conceptalyzer – Language Processing Widget

We finally managed to deploy our prototype in Graaasp in a rather stable version, but still with a lot of open issues to be tackled in later development stages of the ROLE project.

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 assembled from a set of innovative tools realized using different technologies. For further collaborative distributed development experiments we agreed on short, but regular biweekly meetings in order to get aware of occurring problems earlier. The agenda will be inspired by action items of W3C meetings.

VI. CONCLUSION & OUTLOOK

In this paper we provided a report of the collaborative distributed development of the ROLE Christmas Project resulting in a prototype of a Widget-based PLE for language learning. We first described the development process conducted among nine different partners from both academia and industry, with varying technical backgrounds, motivations and interests regarding the whole project. We pointed out that regular communication and the clear definition of goals and a schedule was inevitable during the whole process. Furthermore, we listed useful technical means of collaboration such as communication media, shared documentation, shared code repositories, etc. Furthermore, we had to draw the conclusion that our approach did not work as expected, rising the necessity for an improved approach better suited for collaborative distributed development of Widget-based PLEs. In a section on requirements we presented our use case scenario and gave an insight into the psycho-pedagogical model behind. In that context we pointed out that conceptual and technical work must happen together. We elicited technical requirements to a basic infrastructure for distributed PLE development and presented a selection of technologies for its realization as foundation for our experiments. We then provided an overview of the innovations resulting from individual partner contributions, ranging from integration technologies to scenario-dependent and independent learning service widgets. Finally, we critically discussed the outcome of the ROLE Christmas project and reported a set of technical issues hinting to the conclusion, that Widget technology is not mature and stable enough to enable distributed collaborative PLE development without hassle at this point in time. However, we worked out the requirements and associated problems for distributed implementation of widget based PLEs and collected a lot of valuable experience that will shape future endeavors. In an upcoming consolidation phase, the developer team will stabilize current results, improve and align the development process and then continue work towards a number of bundles for the implementation of the ROLE test bed scenarios.

ACKNOWLEDGMENT

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

[1] K. Beck, M. Beedle, A. van Bennekum, A. Cockburn, W. Cunningham, M.Fowler et al. *Manifesto for Agile Software Development.* 2001. http://www.agilemanifesto.org/

[2] K. Beck and C. Andres. *Extreme Programming Explained. Embrace Change*. 2nd Edition, Addison Wesley, December 2004, ISBN 0-321-27865-8

[3] git. The fast version control system. http://git-scm.com/. Last visited: Jan 2010.

[4] A. Nussbaumer, K. Fruhmann, U. Kirschenmann, P. Ferdinand, A. Kiefel, A. Naeve . ROLE Deliverable 6.1: Common psycho-pedagogical framework. Unpublished.

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

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

[7] Glassfish – Project SocialSite. https://socialsite.dev.java.net/. Last visit January 2010.

[8] Apache Wookie. http://incubator.apache.org/wookie/. Last visit January 2010.

[9] Marcos Cáceres . *Widget Packaging and Configuration*. W3C Candidate Recommendation. W3C. December 2009.

[10] Open Ajax Alliance. Standardizing Ajax Development. http://www.openajax.org. Last visit January 2010.

[11] Open Application. http://code.google.com/p/open-app/. Last visit January 2010.

[12] P. Saint-Andre, ed . *Extensible Messaging and Presence Protocol (XMPP): Core*, IETF proposed standard, RFC 3920, Oct. 2004; http://www.ietf.org/rfc/rfc3920.txt.

[13] P. Saint-Andre, ed . *Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence*, IETF proposed standard, RFC 3921, 2004; http://www.ietf.org/rfc/rfc3921.txt

[14] P. Millard, P. Saint-Andre, Ralph Meijer, ed. *XEP-060 Publish-Subscribe*. Draft Standard in progress, Dec. 2009; http://xmpp.org/extensions/xep-0060.html

[15] P. Saint-Andre, K. Smith, ed. *XEP-163 Personal Eventing Protocol*. Draft Standard in progress, Oct. 2009; http://xmpp.org/extensions/xep-0163.html

[16] I. Hickson and D. Hyatt. *HTML5: A vocabulary and associated APIs for HTML and XHTML*. Editor's Draft. W3C. Jan. 2010; http://dev.w3.org/html5/spec/spec

[17] 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.

[18] Apache Axis2: Java Next Generation Web Services. Last updated: 23 Oct 2009; http://ws.apache.org/axis2

[19] Gears. Improving Your Web Browser; http://gears.google.com/. Last visit January 2010.

[20] SQLite; http://www.sqlite.org. Last visit January 2010.

[21] U&I Learning; http://www2.learning-service.com/portal/uni/. Last visit January 2010.

[22] A. Inselberg, "N- Dimensional Coordinates," in *IEEE Pattern Analysis & Machine Intelligence (PAMI), "Picture Data Description & Management"*, Asilomar, California, 1980, 136.

[23] S. Leitner: *So lernt man lernen.* 13. Auflage. Verlag Herder, Freiburg 1995, ISBN 3-451-05060-9

[24] B. Simon, D. Massart, F. van Assche, S. Ternier, and E . Duval. Simple Query Interface Specification. Public Draft. April 2005; http://ariadne.cs.kuleuven.be/lomi/index.php/LorInteroperability

[25] R. Koblischke, *Federated Ranking: Evaluating Result Merging Algorithms for Distributed Retrieval*, master's thesis, Vienna University of Economics and Business, Vienna, 2010.

[26] F. Wild, B . Hoisl , and G. Burek, "Positioning for Conceptual Development using Latent Semantic Analysis," in *Proceedings of the EACL Workshop on GEMS*, Athens, Greece, pp. 41-48, 2009.

AUTHORS

D. Renzel (renzel@dbis.rwth-aachen.de) is with the Chair for Computer Science 5, RWTH Aachen University, Aachen, Germany.

C. Höbelt (christina.hoebelt@im-c.de) is with imc information multimedia communication AG, Saarbrücken, Germany.

D. Dahrendorf (daniel.dahrendorf@im-c.de) is with imc information multimedia communication AG, Saarbrücken, Germany.

M. Friedrich (martin.friedrich@fit.fraunhofer.de) is with the Fraunhofer Institute for Applied Information Technology, Sankt Augustin, Germany.

F. Mödritscher (fmoedrit@wu.ac.at) is with the Vienna University of Economics and Business, Vienna, Austria.

K. Verbert (katrien.verbert@cs.kuleuven.be) is with the Katholieke Universiteit Leuven, Leuven, Belgium.

S. Govaerts (sten.govaerts@cs.kuleuven.be) is with the Katholieke Universiteit Leuven, Leuven, Belgium.

M. Palmér (matthias@nada.kth.se) is with the Uppsala University, Uppsala, Sweden.

Evgeny Bogdanov (evgeny.bogdanov@epfl.ch) is with the École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland.

This article was modified from a presentation at the International Conference of Interactive Computer Aided Learning ICL2009, September 2009 in Villach, Austria. Submitted 11 September 2009. Published as resubmitted by the authors on 17 January 2010.