Integrating Cloud Services to Support the Formation of Informal Learning Groups

Adam Giemza[1], Marc Jansen[2] and H. Ulrich Hoppe[1]
[1]University of Duisburg-Essen, Duisburg, Germany
[2]University of Applied Sciences Ruhr West, Bottrop, Germany

Keywords: Cloud Services, Mobile Application, Group Formation, Learner Support.

Abstract: Cloud Services like Google Drive, Brainstormer, Doodle are meanwhile frequently used as tools for learning in various contexts. They provide storage facilities, production tools and particularly also coordination support. The management of these heterogeneous tools is a challenge for the individual users as well as for the usage in groups. This paper presents a mobile application to support the learners in the formation of informal learning groups and integrates heterogeneous cloud services to support group formation and further group work in a campus environment. Furthermore, this paper tackles the ambiguity of the term cloud application and tries to provide a definition and discusses the process of bringing the Brainstormer web application into the cloud for integration with the mobile application.

1 INTRODUCTION

Nowadays presence-based university studies are supported by a complex and heterogeneous information infrastructure: Course-specific information is typically embedded in learning management systems, with possibly different systems used even in one university. Administrative and organizational information about study programs, general student services etc. is often provided through other information channels. Students access, partially store and further manage this information on personal devices such as notebooks or smartphones. Generally available social media and cloud services may be used to share and further distribute such information. In this sense, a modern campus information environment is an example of a complex, heterogeneous and somewhat scattered infrastructure (cf. (Hanseth, 2010)).

From the given characterization we can directly construe the challenge of better integration. Integration can be addressed from two sides: from the sources (i.e. information and service providers) or from the user (student) perspective. Integration from the source would need to be based on strong premises of being able to change large parts of grown systems. So, in a modest and realistic approach we address the problem from the user perspective.

We have specialized the problem in the form of a typical application scenario: Especially students in their first semesters on campus face the challenge of finding peer groups to collaboratively work on assignments or prepare for exams. This includes the problem of group formation and basic support for group work. As a further specification, we have set a focus on delivering and managing information on personal mobile devices and on considering opportunities in space and time in a campus environment. This conceptual and technical challenge has been addressed in a still ongoing software development project with master level students from an interdisciplinary study program on interactive media and applied cognitive science.

2 RELATED WORK

Providing orientation support for freshmen on campus has been a theme of a number of mobile learning applications (Lucke, 2011), (Giemza et al., 2012), often also using a game-based approach. While these approaches focus on learning the university campus and the surroundings, our application focuses on getting to know new fellow students to form learning groups. The gamification aspect has a lower importance, as we believe that the students have an

Giemza A., Jansen M. and Ulrich Hoppe H. (2013).
Integrating Cloud Services to Support the Formation of Informal Learning Groups.
In *Proceedings of the 5th International Conference on Computer Supported Education*, pages 724-730
DOI: 10.5220/0004455307240730

intrinsic motivation (Ryan and Deci, 2000) to find learning groups for gaining knowledge from others in the collaborative learning process.

Jansen et al. (Jansen et al., 2005) describe a prototypical campus information system that integrates interactive public displays with personal mobile devices and supports personalized and location-aware information transfer. If such an infrastructure (including interactive displays distributed over the campus) were widely available and accessible this would have been an ideal basis for implementing the specific functions supporting group formation and support of group work. However, unfortunately we cannot rely on such premises.

The formation of learning groups has been studied from an intelligent systems perspective using quite sophisticated processing techniques (Largillier and Vassileva, 2012), (Isotani et al., 2009), (Hoppe, 1995). In our case, we do not rely on deep knowledge modeling and intelligent learner diagnosis as input to the group formation process, although this could be considered in future versions. Our starting point and current focus is on opportunistic usage of simple user profiles, general information and coincidences in time and space.

Jansen et al. (Jansen et al., 2013) have classified the usage of existing cloud services from an educational perspective. They distinguish different types of services, including archiving/repository services (such as Google Drive), communication and coordination services (such as Twitter or Doodle) and rich production services beyond simple text messages (such as MindMeister). Still, simple repository or communication services are predominant in many educational applications. A future potential is seen in the area of processing services and especially services for learning analytics.

3 Meet2Learn

Meet2Learn aims at supporting students to meet new fellow students with the goal to form learning groups for their joint lectures. Once the learning group has been formed, the system will act as an organizer to integrate learning materials and other artifacts worked out collaboratively from different cloud services.

In Meet2Learn learning groups are informal groupings organized by students themselves, in contrast to exercise groups offered beside a lecture. Here the learners take the initiative to organize and execute the learning group. Nevertheless they are formed around a lecture that builds the context and the content of the learning group. To support learners in creating and finding appropriate learning groups, the system needs to be aware of the lectures being offered at the University. This will prevent the users to create multiple learning groups concerning one and the same lecture only due to different naming by the learner while searching for a group or while creating it. Furthermore learners need to be represented in the system to provide grouping support based on a user profile. This profile needs to contain the course of studies, the number of semesters and finally the attended lectures. This will empower the system to provide recommendations based on user-based collaborative filtering methods (Resnick et al., 1994). Finally, universities provide various learning management systems (LMS) like Blackboard or Moodle, which can be integrated with the learning groups to import learning material from the lecture as well as to export collaboratively elaborated content back into the LMS. This content will be exported from the "production-type" cloud services like Google Drive or Brainstormer (Collide Brainstormer, 2013).

In the sequel we will explain the components of the Meet2Learn applications in more detail. The target groups for the first prototype of Meet2Learn are students at the University of Duisburg-Essen studying *Applied Computer Science* and *Applied Communication and Media Science*. Therefore the system imports the lectures from the university system called LSF and exports learning groups into a Moodle provided by the university as well.

3.1 Architecture Overview

The frontend of Meet2Learn has been developed as a native mobile Android application. This results in the benefit of a practically permanent availability to the students for checking the status of learning groups and receiving notifications about relevant information. The backend uses an agent architecture on top of a blackboard system (Weinbrenner et al., 2007). This allows for a flexible and extendible design with connection to heterogeneous services using multiple programming languages. The integration of cloud services takes place on the server side through agents as well as directly on the client side through the Android application (see **Figure 1**).

The application integrates the following types of cloud-based services on the smartphone: *Communication, Production* and *Repository*. Twitter and

Facebook are integrated as the first type of service. Hereby students can announce learning groups to their friends that are not yet using the Meet2Learn application. This will increase not only the number of users but also the probability that a learning group will take place with enough participants. Furthermore it will increase the extrinsic motivation (Brophy, 2004) to the learner, as he or she announced the intention to participate in a learning group to the public. Google Drive and Brainstormer are integrated as cloud-based production services. Here learners produce learning artifacts collaborative and link them to the learning group for sharing the results. Dropbox is integrated as a cloud-based repository service. Learners can link uploaded resources and provide direct access through the learning group. The client-based integration makes use of the rich connectivity and extendibility of Android through flexible built-in and third-party APIs.

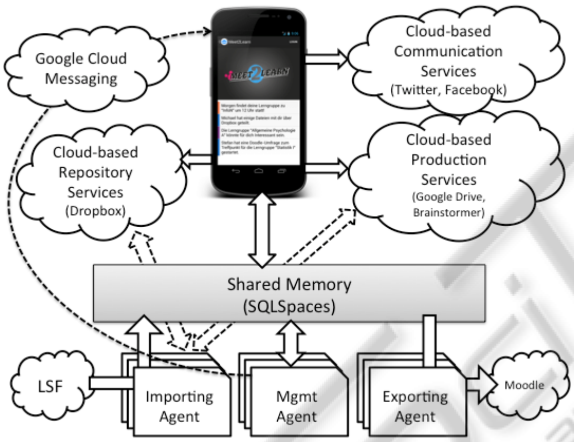

Figure 1: Meet2Learn Architecture Overview.

The server side uses an agent architecture utilizing the SQLSpaces (Weinbrenner et al., 2007) as a shared memory. The agents have different tasks assigned. In the lower left of **Figure 1** importing agents gather data from different heterogeneous sources and import them into the system. The current configuration integrates the study program from the central repository of the university (LSF). This data is used to support the students in adding lectures to their profiles and prevents multiple notations for one lecture. The management agents are responsible for processing data and for controlling the system. One particular feature is the notification of the learners about new events (e.g., a new learning group of interest will take place closely to the users current location). This is implemented by the native Android notification mechanism using the Google Cloud Messaging (GCM) service. Finally, exporting agents integrate learning groups into external systems, like the aforementioned learning management systems (LMS). We are planning to integrate the university's Moodle by exporting Meet2Learn learning groups into Moodle cohorts. This will allow learners to continue the work of a well-established learning group inside Moodle closely located to the lecture.

3.2 Mobile Application

The main entry point for the learner to the system is the Meet2Learn mobile application. Figure 2 shows three main views for the user. The first view (a) represents the main entry point to the application. Here the user finds a "news feed" containing the latest information about learning groups as well as notifications with recommendations and reminders to the learner. The messages are typed by a color tag on the left and can be therefore quickly grasped and identified. A click on the message opens a new view presenting more details about the event. The second screen (b) shows the group creation screen. The learner can configure a new group by setting multiple parameters like the name, time and place and publish this new group to the system. Fellow students can search this group and join it. Finally, the users are responsible themselves to participate physically in the informal learning group.

The last screen (c) shows the details a learning group. The learner can see the title, the participants, the next meeting and finally the resources shared within the learning group using the aforementioned cloud services (here Dropbox and Google Drive). For this purpose Meet2Learn registers itself as a recipient of "share events" of other applications on the Android phone. A user can therefore open the Google Drive app and share the link to a resource with the Meet2Learn app. This will prompt the user to select the correct learning group to share the document with. All participants of this group will receive a notification that a new resource has been shared with them in the shared group. The view (c) also implements the standard share button (top right) that allows the users to publish a learning group on cloud-based communication services like Twitter and Facebook. In this example, the user can share the information about the learning group for the lecture "InfoN" by clicking the share button. The system will present a list of possible (cloud) services to use to process the message. If the user selects Twitter, the system will open the common Twitter dialog with a pre-filled message about the group and

Figure 2: Meet2Learn App - (a) News feed and notifications (b) Group creation view (c) Group details view.

a link to a web page presenting some more details about the learning group. Then the user can send the tweet and share this to his or her followers. This allows promoting the learning group to other learners not using Meet2Learn and thus allows new users to join the Meet2Learn community.

4 CLOUDIFICATION OF WEB APPS: THE BRAINSTORMER EXAMPLE

In the previous sections we have mentioned Brainstormer (Collide Brainstormer, 2013) as a rich production cloud-based service for collaborative brainstormings on assignments. In fact, we have developed Brainstormer as a Web Application in first place and later deployed to the Open PaaS (Platform as a Service) provider Cloudfoundry[1]. We will use this example and our experiences to discuss what "cloudification" actually means beyond mere web applications and what possible added values can be gained.

There is still no distinct definition or description of the requirements for Cloud Applications. Therefore, this section provides at first our view of a possible definition of the notion of Cloud Application. Based on this definition and on the Brainstormer example, we provide a technical description of steps that are necessary for moving an already existing application into a Cloud Computing environment.

4.1 A Possible Definition for Cloud Applications

Having a look at one of the major definitions of Cloud Computing (Mell and Grace, 2009), a distinction in three different level of abstraction can be found:

1. IaaS - Infrastructure as a Service: Virtual provision of computing power and/or memory. A prominent example of an IaaS service is the Amazon WS service.
2. PaaS – Platform as a Service: Provision of a runtime environment, like application servers, databases etc. Cloudfoundry is an example for a (open) PaaS provider.
3. SaaS – Software as a Service: Provision of usually browser based applications that can directly be used. Here, Google Docs or the Customer Relationship Management software of salesforce.com serves as examples.

Obviously, the only layer the end-user actually interacts with, is the SaaS layer. Therefore, Cloud Applications will be deployed to this level of abstraction. Nevertheless, in order to provide Cloud Computing key factors to the Cloud Application, usually, the SaaS layer will be deployed on top of a PaaS layer running a flexible IaaS environment. Therefore, the first thing to mention for a Cloud Application is, that it needs to run in a Cloud Computing environment, as explained above.

Furthermore, (Mell and Grace, 2009) provides a list of essential characteristics an application needs to provide in order for being called a Cloud Application. First of all, the application needs to provide an *on-demand self-service*, which basically means that the necessary computational resources are provided to the application without human interaction. In the context of learning scenarios, this would mean that a Cloud Application would allow the learner to use the application independent of time and space, with almost unlimited resources. The second aspect men-

[1] Cloudfoundry - http://www.cloudfoundry.com/ Last visited: Feb. 2013

tioned is the *broad network access*, meaning that the application itself can be accessed via standardized tools and/or off-the-shelf applications (Pettersson and Vogel, 2012). The aspect of *resource pooling* is not that much of interest with respect to the SaaS layer since the provision of according resources is usually ensured by deploying the SaaS layer on top of a PaaS and especially an IaaS layer. Therefore, this characteristic does not provide any new impacts for a Cloud Application within the context of learning scenarios. The fourth aspect, the *rapid elasticity*, yields to the high degree of scalability that Cloud Applications need to ensure. Here, with respect to learning scenarios, this characteristic does not provide new requirements, since, similar to the aspect of resource pooling, this aspect would be answered by a deployment to a PaaS and/or a IaaS infrastructure. Taking a broader view to the question about scalability issues for applications in the context of learning scenarios (Giemza et al., 2012), the question of scalability provides an interesting topic of current research. The last characteristic mentioned is the characteristic of *service measuring*. Here, for Cloud Computing environments, a special need for measuring the service consumption is necessary, especially since the resource provided for the service in question are highly flexible. Again, with respect to learning scenarios, this characteristic is not of high interest as it would usually also be handled on the PaaS and IaaS level and/or there is no special business model for learning applications so far.

Therefore, a Cloud Application in the context of learning scenarios, might be defined as a highly scalable application, both from a technical and a collaboration point of view, that is available via standardized interfaces and with almost unlimited resources.

4.2 Necessary Steps for Moving a Learning Application to the Cloud

In order to fulfill the requirements discussed in the last subsection, there are a couple of steps necessary in order to extend an already existing application to a Cloud Application. We will discuss these steps in general and also in the context of the Brainstormer example.

First of all, the accessibility of the application needs to be evaluated. The overall process for the modification of an already existing application to a Cloud Application is shown in **Figure 3**.

Here, the major question is, whether the application is available to large audience (in our context the learners) via standardized interfaces. Of course this becomes pretty easy in case of web applications, as with Brainstormer, but might be a little bit more crucial with respect to non web-based applications. Here, usually, the question is not only about the user interface with whom the learner usually interacts, but also the question of how data can be added (in terms of input channels (Bollen et al., 2012)) to the applications, might be an indicator for the availability of the application itself. Brainstormer shows also, that these questions are not mutually exclusive, as it provides a web interface as well as an API for third-party clients to contribute content.

The next would be to deploy the application to an IaaS infrastructure. Depending on the architecture of the application, this step might be more or less trivial. Our experience from the deployment of two learning applications to an IaaS infrastructure shows that this step is usually not as trivial as it might appear on the first view. This layer is however already covered when using PaaS providers as Cloudfoundry.

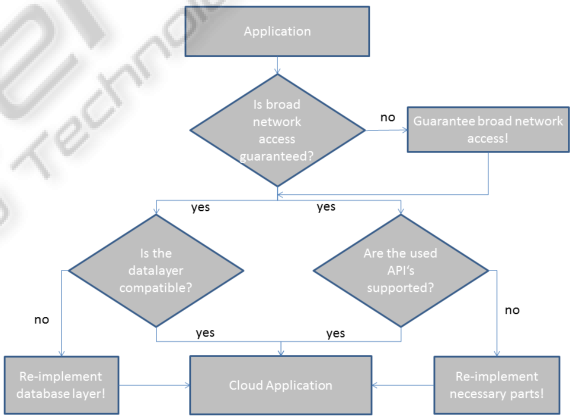

Figure 3: Necessary process for the modification of an existing application to a Cloud Application.

Basically, the problems that usually occur can be subdivided into two different categories. The first category yields towards the data layer of the application. Usually, applications used in learning scenarios depend on some sort of database that stores the developed learning objects and provides additional material to the learner. The examples described in (Giemza et al., 2012) and (Bollen et al., 2012) that we tried to deploy to an IaaS infrastructure, the type of database that the application used before was not supported. Therefore, we had to re-implement the database layer. Of course another solution to this challenge might have been to use our already existing database, but this would have resulted in the question whether we can ensure the necessary scalability, availability and performance of the database. Especially these kinds of problems should be solved by the deployment of the application to an IaaS infrastructure. The case of Brainstormer was easier, as Brainstormer uses a MongoDB database in the backend. Cloudfoundry provides instances of MongoDB, therefore the migration was limited to a re-configuration of the database settings.

The second step for moving an application to a Cloud Computing environment, takes the used frameworks and external API's into account. One of the major characteristics of modern software development is the high degree of re-use of already existing frameworks and API's. Here, it is important to ensure that the provider supports the frameworks and API's used for the development of an application. The usage of standard external services, e.g., Web Services, demands special attention. It needs to be ensured that not only the application itself is highly scalable (from a technical point of view) but also that the used external services are scalable in the same dimension.

Finally we can conclude, that Brainstormer can be considered as a Cloud Application with respect to our definition and the explained steps to migrate a Web Application to a Cloud Application.

5 CONCLUSIONS AND OUTLOOK

The Meet2Learn application is an example of integrating and managing information from heterogeneous sources to support personalized and group learning. It takes the heterogeneity of the surrounding information infrastructure as a given and aims at user-side integration. It combines general information from the campus environment with personal profiles and location information. It also includes the use of several types of cloud services. In this context, the previously developed Brainstormer application for generating and collecting ideas in a group has been "cloudified".

The system will be ready for use in the summer term (starting in April 2013). It will be evaluated with a group of beginners from Interactive Media and Applied Cognitive Science. From a systems perspective, the application will be enhanced by using ontologies and semantic processing for identifying specific user interest and needs.

REFERENCES

Hanseth, O. (2010). From Systems and Tools to Networks and Infrastructures - from Design to Cultivation: Towards a design Theory of Information Infrastructures., In Jonny Holmstrøm (ed.), *Industrial Informatics design, Use and Innovation.* IGI Global: 122-156.

Lucke, U. (2011). A pervasive game for freshmen to explore their campus: Requirements and design issues. in Proceedings of the IADIS International Conference on Mobile Learning. Spain. pp. 151-158.

Giemza, A., Verheyen, P., & Hoppe, H. U. (2012). Challenges in Scaling Mobile Learning Applications: The Example of Quizzer. 2012 IEEE Seventh International Conference on Wireless, Mobile and Ubiquitous Technology in Education. pp. 287–291

Ryan, R., & Deci, E. (2000). Intrinsic and Extrinsic Motivations: Classic Definitions and New Directions. Contemporary Educational Psychology, 25(1), pp. 54-67

Jansen, M., Uzun, I., Hoppe, H.U., Rossmanith, P. (2005). Integrating Heterogeneous Personal Devices with Public Display-Based Information Services. Proceedings of IEEE WMTE 2005: 149-153

Largillier, T. & Vassileva, J. (2012). Using Collective Trust for Group Formation. Procedings of the 18th International Conference4 on Collaboration and Tehcnology (CRIWG 2012). Springer: 137-144.

Isotani, S., Inaba, A., Ikeda, M., Mizoguchi, R. An ontology engineering ap-proach to the realization of theory-driven group formation. International Journal of Computer-Supported Collaborative Learning 4(4), 445-478 (2009)

Hoppe, H. U. The use of multiple student modeling to parameterize group learning. In Proceedings of *AIED '95*. Washinton D.C., August 1995. 234-241.

Jansen, M., Bollen, L., Baloian, N., Hoppe, H.U. (2013). Cloud Services for Learning Scenarios: Widening the Perspective. In Proceedings of the 1st International Workshop on Cloud Education Environments. Guatemala. pp. 33-37

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., & Riedl, J. (1994). GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In R. Furuta & C. M. Neuwirth (Eds.). Proceedings of the 1994 ACM conference on Computer supported cooperative work. pp. 175-186

Collide Brainstormer – http://brainstormer.cloudfoundry.com/ - Last visited: Feb. 2013

Weinbrenner, S., Giemza, A., & Hoppe, H. U. (2007). Engineering Heterogeneous Distributed Learning Environments Using Tuple Spaces as an Architectural Platform. Seventh IEEE International Conference on Advanced Learning Technologies. pp. 434-436

Brophy, J. (2004). *Motivating students to learn. British Journal of Educational Technology* (Vol. 36, p. 418).

Lawrence Erlbaum Associates.
Mell, P., Grance, T. (2009). The NIST definition of Cloud Computing. Version 15
Pettersson, O. and Vogel, B. (2012). Reusability and Interoperability in Mobile Learning: A Study of Current Practices. 2012 Workshop on Scalability and Interoperability Dimensions for Mobile Learning Applications (SID-ML 2012), In Proceedings of the Seventh IEEE WMUTE 2012.
Giemza, A., Bollen, L., Jansen, M., Hoppe, H.U. (2012). An Architecture for Supporting Heterogeneous Multi-Device Learning Environments. In: Proceedings of the 7th IEEE International Conference on Wireless, Mobile and Ubiquitous Technologies in Education
Bollen, L., Eimler, S.C., Jansen, M., Engler, J, (2012). Enabling and Evaluating Mobile Learning Scenarios with Multiple Input Channels. In: Proceedings of the 18th International Conference on Collaboration and Technology (CRIWG)