Title: An Adaptive Event-based System for Anytime, Anywhere, Awareness Services in Online Teamworks

Vladi Kolici, Polytechnic University of Tirana, Albania (vkolici@fti.edu.al)
Fatos Xhafa, Technical University of Catalonia, Spain (fatos@cs.upc.edu)
Santi Caballé, Open University of Catalonia, Spain (scaballe@uoc.edu)
Leonard Barolli, Fukuoka Institute of Technology, Japan (barolli@fit.ac.jp)

Abstract:
The fast development in mobile technologies IS drastically changing the way people work, learn, collaborate and socialize. One such important activity that has emerged and is being consolidated each time more is the online learning through virtual campuses. While most of online learning services are at present offered through web-based platforms, due to ever-increasing use of smart devices such as smartphones and tablets, researchers and developers are paying attention to exploit the advantages of mobile systems to support online learning. Specifically, the implementation of the A3 paradigm: Anytime, Anywhere, Awareness –that is, notifying users about ongoing activity in their online workspace– provides various advantages to online learners organized in online teams. In this paper we present the requirement analysis, the building blocks of the architecture for efficient event-based system and a prototype implementation of the A3 paradigm that adaptively supports the online collaborative activity.

Keywords :
Online learning ; collaborative team work ; events ; mobile computing ; awareness ;

Short Biographies

Vladi Kolici received his BS and MS in Telecommunication Engineering from the Polytechnic University of Tirana (PUT) in 1997 and 2005, respectively. He obtained his PhD from PUT in May 2009. From 1997 to 2004, he was a Research Associate, from 2005 to 2011 he has been a Lecturer and at present he is Associate Professor at the Department of Electronics and Telecommunications, Faculty of Information Technology, PUT. He is teaching several courses in the areas of wireless and mobile networking, P2P systems and quality of services. His has published several papers in international and national conference proceedings in the areas of P2P and ad hoc networks. His research interests include P2P networks, wireless and mobile wetworks and highspeed networks.

Fatos Xhafa holds a PhD in Computer Science from the Department of Computer Science of the Technical University of Catalonia (UPC), Barcelona, Spain, where he holds a permanent position of *Professor Titular (Hab. Full Professor)*. He was a Visiting Professor at Birkbeck College, University of London (UK) during academic year 2009-2010 and Research Associate at Drexel University, Philadelphia (USA) during academic term 2004/2005. Prof. Xhafa has widely published in peer reviewed international journals, conferences/workshops, book chapters and edited books and proceedings in the field. Prof. Xhafa has an extensive editorial and reviewing service and is actively participating in the organization of several international conferences. His research interests include parallel and distributed algorithms, security, optimization, networking and distributed computing. More information can be found at http://www.lsi.upc.edu/~fatos/

Santi Caballé holds holds Ph.D. (2008), Master's (2005) and Bachelor's (2003) degrees in Computing Engineering from the Open University of Catalonia (UOC). His teaching activity started in 2004 as an Assistant Professor at the department of IT, Multimedia and Telecommunication of the UOC, where he became Associate Professor in 2006 and Academic Director of the Postgraduate program of Software Engineering in 2010. He received a teaching accreditation as Tenure-track Lecturer issued by the Catalan Government (2008) and was awarded with teaching merits for the period 2005-2009 by the UOC. He is currently teaching and coordinating a variety of on-line courses of the Bachelor and

Master in Computing Engineering and other post-graduate programs in the areas of Software Engineering, Collaborative Learning and Information Systems. He is also supervising several Master and PhD thesis at the UOC. His research interests include eLearning, virtual campuses, distributed computing, Web-based computing, among others.

Leonard Barolli holds a Ph.D. degree from Yamagata University, Japan in 1997. He is presently a Full Professor at Department of Information and Communication Engineering, Faculty of Information Engineering, Fukuoka Institute of Technology (FIT). Dr. Barolli was an Editor of Information Processing Society of Japan (IPSJ) Journal and has served as a Guest Editor of many International Journals. He is engaged as a Program Committee (PC) Member in many International Conferences. Dr. Barolli has widely published in Referred Journals, International Conference Preceedings and National Workshops. His research interests include high-speed networks, mobile communication systems, ad-hoc networking, sensor networks, P2P systems, Quality of Service (QoS), traffic control mechanisms, intelligent algorithms, network protocols, agent-based systems, grid and Internet computing. Dr. Barolli is a member of IPSJ, Japan Society for Fuzzy Theory and Systems (SOFT), IEEE Society and IEEE.

1. Introduction and motivation

A new, widespread family of applications has emerged as a combination of best advantages offered by mobile computing and Cloud computing (also referred to as "mobile cloud computing" [1]). Indeed, on the one hand, in the recent years we have seen how the concept of mobile device has considerably evolved towards a variety of applications, including more applications for work purposes. The frontier between computers and mobile devices is each time more blurred, and people start to use massively new smart devices that can carry with them everywhere not only as means of communication but also to accomplish tasks online. Staying online, connected to online resources, and receiving information, from different channels, in an intuitive and easy way enables to shorten considerably the response time; it is also an effective way to request help and support from others. For instance, in online learning, scaffolding is very important to learners, either in a virtual classroom or a virtual workspace of an online team. Traditionally, scaffolding through web-based online applications might not have the desired effect given that considerable time could pass from the moment when a help or support request is issued to the time it is done effective.

By combining, Web-based and Cloud based technologies with the use of mobile devices, the online learning and collaborative activity can be much more timely, efficient and adaptive [2]. In particular, scaffolding can be a much more effective way of accomplishing learning activities because learners can request support at any time and other learners can scaffold their team mates at any time, anywhere due to the awareness services. Currently, we also see how social networks have been widely used for such purpose, by adapting to this idea, that is, using social networking properties in support to learners. For instance, one such feature explored is the emotional awareness as the emotional support is an important aspect of social networking. Several studies from mobile learning domain (e.g. [3], [4]), have pointed out that using mobile devices on online learning can contribute to increasing the perception of social connectedness of learners yielding to emotional well-being among learners and reducing the frustration often produced by the isolation in online virtual classrooms.

The implementation of the A3 paradigm: *anytime, anywhere, awareness* would make online systems more adaptive, because the online learners can adapt their activity and needs to those of others "as they go". Being anytime, anywhere and aware

about what happens in online setting is therefore a constant demand in online collaborative work. In this paper, we discuss about this paradigm focused on online teams, describing in a practical way, how the team communication can significantly improve by using mobile devices. Starting from a bottom-up approach we describe how the A3 paradigm can be applied to different mobile team working scenarios [5]. Our approach is based on the definition and use of events linked to state changes of the objects in the common learning workspace. The events are then processed and sent to mobile clients for consumption following the A3 paradigm.

In the current state of the art of the event-based system, we can find many interesting methodological proposals for designing event-based mobile applications. Thus, the authors in [6] proposed an event-based coordination approach to execute process oriented composite applications, with components on server and at the client site. Another interesting work in this context, but rather focused on task oriented team collaboration, deals with notification and awareness synchronizing task-oriented collaborative activity [7]. The authors in [8] proposed the use of XML messaging for mobile devices for effectice communication. Different from these approaches, in our approach, the objective is to make the system as simple as possible, so even it's possible to adapt the approach in [6], we want to use existing communication channels between clients, widely used by users in their social networking life, without the need of use new protocols. Additionally, we consider that in the teamwork, people prefer working together using exiting tools and our research aims to study how mobile devices can be integrated into existing online applications.

As per the use of protocols, again the idea is not to use any specific protocol to deliver messages, but instead, to use existing ones building a gateway between our platform and the delivery platform. In all, our aim is to build a new layer integrated onto existing teamwork systems, which can be also possible to be used in another kind of systems, which need to spread messages and notifications to groups or particular users. Taking into account the current convergence between computers and mobile devices, the system should be used not only to add mobile capabilities to existing systems, but also to add a complete delivery system to those that do not have such delivery services, namely, A3 services.

Due to the computational limitation of mobile devices, the extension of web-based systems to support also mobile clients require specific design and implementation to make the system usable, efficient and scalable. In this work we use an event-based approach to extend web-based applications to support Anytime, Anywhere, Awareness in online teams. It should be noted that the A3 paradigm is meant not only as informing channel but also for social support and scaffolding. The key feature here is the definition of events linked to state changes of the objects in the common workspace. The events are then processed and sent to mobile clients for consumption.

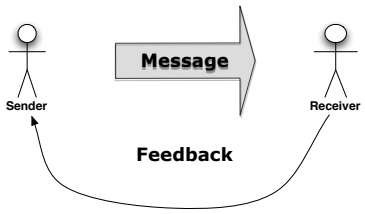
Goals: The main design goals of an adaptive A3 paradigm for onlione collaborative setting can be listed as follows:

- *Simplicity*: reusing existing communication channels and technologies.
- *High availability*: building a system that relies on technologies that can support high availability by default.
- *Scalability*: taking into account in the design that, without a big effort, the system can scale efficiently to more users and larger amounts of online teamwork.
- *Modularity*: choosing a software architecture that guarantees that the different parts can be improved, changed or new ones added without affecting the other parts of the system.
- *Independency from devices*: again the modular software architecture will help to add new devices without big efforts.
- *Security*: in the event-based system we need to ensure that the system will take into account the needed security features.
- *Openness*: building an open system is a key requirement nowadays to facilitate integration and inter-connection with other systems that use or need event-based notifications.

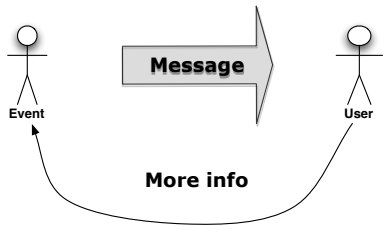
The rest of the paper is structured as follows. In Section 2, we describe the main concepts and requirements upon which our proposal relies. Then, we outline the design of the event management in Section 3. In Section 4, we introduce the implementation issues and prototyping in order to demonstrate how the event management could be done. In section 5, we present how the system can be fit into an eLearning scenario. Finally, we present some conclusions and an outlook to future research opportunities in Section 6.

2. Main concepts and requirements

We follow the transactional model of communication [9] where the source sends a message, through a channel, to the receiver and a feedback comes back to the sender:



Then, we transform it into the model where an event originates a message that is sent, through a channel (SMS, Twitter, email, etc.) to a user.



Once the message is received, the user can take some further actions in order to get more information relevant to an object, a user, etc. Let us see now more in detail these steps.

Event is the main concept in our system. We define an *event* as the change of one of the *attributes* of an *object*. *Objects* are entities that have properties that can be *observable*. These attributes can change overtime, but while they remain the unchanged, they define one possible *state* of the object. In our system each *object* has associated a process, called *observer*, which will detect the changes in the objects' state. In order to group all the *objects* that are related according to some coherent criteria, we have defined the notion of *context*. Also each *object* has associated one or more possible actions, namely *retroactions,* which are actions that users can further take with the system when a message notification is received from the system.

In the system, we have the targeted users –the ones that will receive the messages. In many applications, however, we may have a collection of users, joined together in a group. In the former case, notifications are sent to a particular user while in the later, the messages are sent to all users belonging to the same group. We can talk in this case about group notification. This introduces the concept of *subscription*. A *subscription* is the link between particular users, or groups of users, and the events that produce the objects due to changes in their state. In case of a group subscription, sending a message to the group means sending a message to each member of the group. *Unsubscription* will happen when a user, or a group of users, cancel a *subscription*.

The action of sending a message is called *notification*. Messages can vary depending on the communication channel we use, so we can talk about SMS, MMS, email, tweet, etc., messages. Messages can be sent in two modes: push and pull. In a *push* mode, the user will receive the message as soon as the message becomes available. In *pull* mode, the user will only receive the message through queries for new messages. The former is a synchronous transmission mode while the later is an asynchronous transmission mode.

We sum up the key concepts of the event-based notification system in Table I.

Table I: Concepts and terminology.

Event	The change in one of the attributes of the object.
Attribute	Property of the object that can be observed.
Object	Things that has properties that can be observable.

Observer	Process that detects events.
Context	Group of objects/properties that are coherently related.
Retroactions	Actions that users can take in order to interact with the objects / the system.
User	Person that use the system.
Group	User association.
Message	Information that a user receives.
Notification	Action of sending a message.
State	Values of the objects attributes in a certain period of time.
Subscription	Link between users, o groups of users, and the events that produce the changes in objects' states.
Unsubscription	Cancelation of the subscription.
Push mode	Mode of sending a message when it becomes available, once an event happens.
Pull mode	The message is sent due users' queries about new messages.

The analysis led to the entity diagram shown in Figure 1.

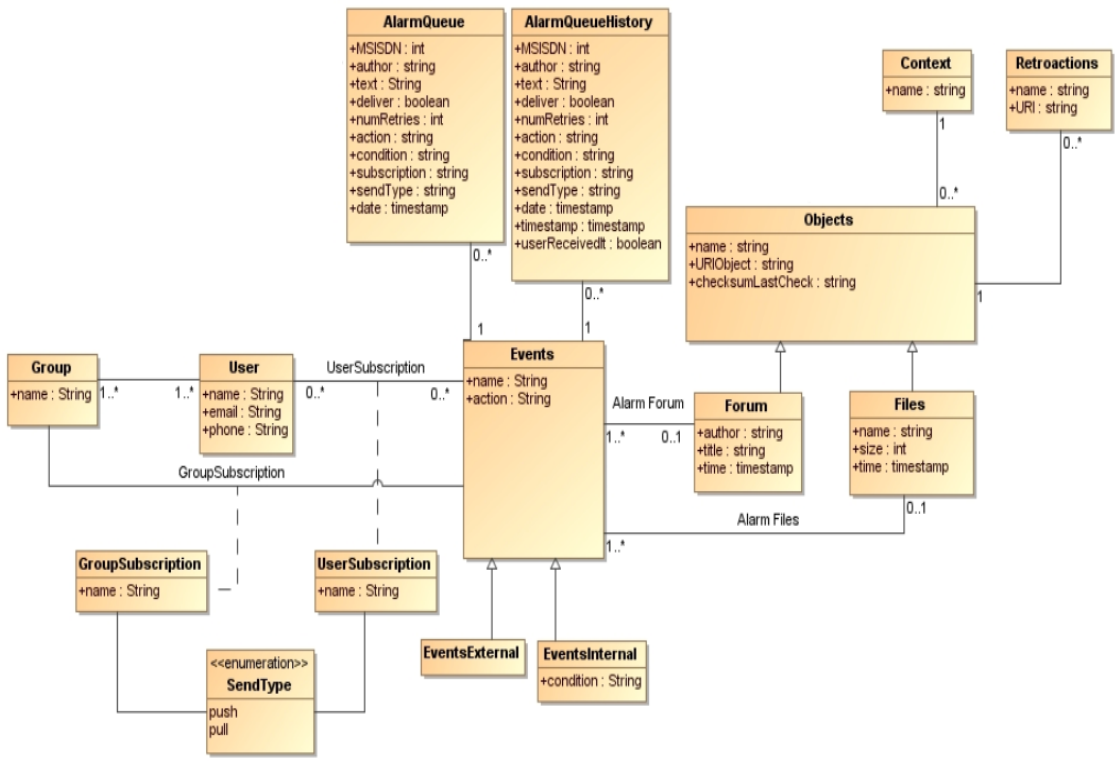


Figure 1: Entity diagram of the event-based system

3. Design of the event management

The event management model includes management of users, subscriptions and events (see Figure 2 for a graphical representation).

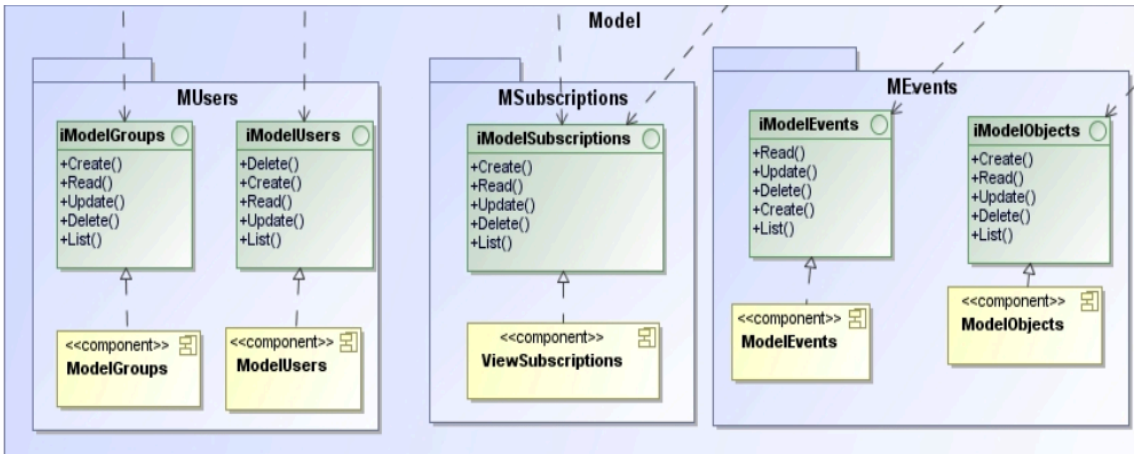


Figure 2: System model

As can be seen, the system is divided into two main building blocks or modules:

3.1 User management and interaction module: In this module we manage all the users, groups, events, objects and subscriptions / un-subscriptions. We take into account that third parties system could use our system.

3.2 Core system module: In this module, we manage all the processes that observe the objects, in order to detect events, and generate notifications to the groups or particular users. Again, we take into account that third parties can use this module of our system. In this later case, it is upon third parties systemsto take the responsibility to observe the objects and send to our system all the notifications.

In terms of system architecture, the notification system is organized into following layers.

Client layer
This layer interacts with the system. Users use existing client interfaces to interact with the system depending on the implementation of the presentation layer. The client interface can be of various forms: from a web browser (in order to interact with the presentation layer in charge of manage the accounts) to a SMS client that shows the messages sent by the system to the user.

Presentation layer
This layer is in charge of interacting with clients. We have a variety of presentation layers: web, wap, SMS…, one for each client layer. Also we have defined an API in order to be used by third parties. This API offers the possibility to users to subscribe and unsubscribe and also to do queries to the system (especially, for the pull mode). Additionally, in this layer we have designed also the packages that are used to send

messages. Depending on the preferred communication channel, we can find SMS connections, twitter account, email account…etc.

Business layer
This layer implements all the logics of the application by carefully implementing the observer objects, the subscriptions as well as the mapping of groups/users to subscribed events. We implement the observer so that the processes that are constantly observing the objects in order to detect events, and the dispatcher processes in charge to deliver the message(s) to the corresponding groups/users once an event is detected. Each time a new event is detected by the observer, a new message is queued into a messaging queue (see Figure 3). In order to simplify the notification processes, the queue manages only "generic" users messages, this means that messages to groups are converted into users messages as well. The dispatcher is then the process in charge of the delivery and applies the retry policy if a message is not delivered to its destination.

Integration layer
This layer is in charge of integrating data layer and business layer, that is, the data layer interaction with the business layer and the IO layer.

IO Layer
In this layer we implement the data persistence chosen to store all the system information.

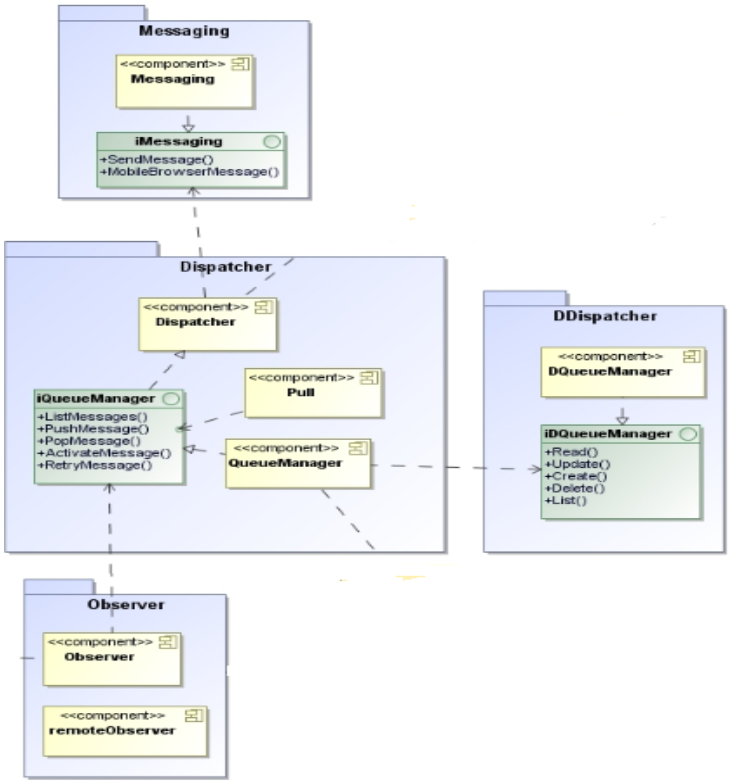


Figure 3: Message dispatcher and observer

The diagram of Figure 4 illustrates the architecture described, following a standard layered architecture.

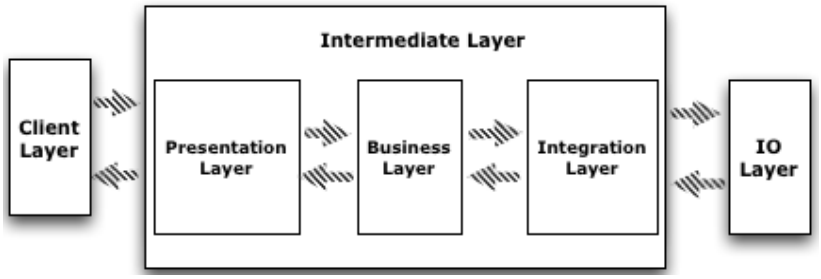


Figure 4: System architecture

4. Implementation issues, prototyping and evaluation

In order to achieve scalability and high availability, we choose first of all HTTP protocol. With a combination of load balancing and web server nowadays both goals can be reached easily. In order to ensure the same flexibility on data persistence, we choose MongoDB. On presentation layer we develop the following interfaces:

Web admin
This interface allows the management of users, events and subscriptions. In the prototyping there was a unique interface without authorization and authentication.

The prototyping divides this interface into three blocks:
- Users: in order to manage users and groups
- Warnings: in order to manage events, objects, context and retroactions
- Subscriptions: in order to manage users and groups subscriptions and un-subscriptions.

Mobile interface
This interface enables users to receive messages on their mobile devices and do some actions at their end.

The prototype has the following interfaces:

- *Pull*: In order to manage pull messages users will be able to activate reception of single messages or messages between any two dates. Activating a pull mode message means that the user will receive the message through the channel that was defined on the event action.
- Also, we implemented an interface to receive an *SMS* with a link in order to check the pull that needs to be activated between two dates. This interface doesn't activate the messages but just visualize them.
- *Mobile*: as an example of mobile interfaces the prototype offers an interface to view a message still to be delivered, all the pull messages between two dates as well as historical information.
- All the messages show the retroaction that the user can do on them.

Third parties interfaces

Third parties can use the following interface in order to interact with the system. The prototype implements the interface *Subscribe* in order to subscribe third parties users. Prior to the subscriptions the events have to be defined in the system. External users can use an external reference that the system uses in order to subscribe the users to the events. In order to manage this subscription on external systems, we implement the list, read, create, update and delete actions. In this way all the actions can be done through the interface.

The prototype implements all the functionalities needed to define users and groups and the subscription to events of these users and groups. In terms of objects, and for the purposes of this work, we defined just the discussion forum and files as context where events arise due activity of online users and online teamworks. In order to detect changes on forums, the module, which detects the changes, reads the RSS associated to the forum to detect new posts (or replies to existing ones). For the files, the module developed checks the modified date of the local file. These were the actions that the observer uses to detect the events, once the observer detects an event, it find the subscribers associated to that, either being groups or particular users. In order to deliver the resulting notifications to the groups, all the users belonging to the group are identified and the message is stored with the status "to be delivered" on the warning queue for each user; for a single user, the system just queues the messages to be delivered to the user. In this way there is no difference between delivery of the messages to groups and single users, there is just one unique queue with user messages.

5. Implementing eLearning scenarios

In this section we see how the system could add benefits to the followings learning pedagogical methodologies: constructivism, problem-based learning and informal learning.

5. 1 Constructivism

In this methodology it is really important the student's interaction in order to learn. Our system would give the possibility to the students to be more in touch with the virtual classroom and the resources (discussion forums, files) without the need to visit each time the web interface, for instance, in order to find new messages. This will allow them to be more quick and dynamic in their relationship with the virtual classroom and other learners. For the case of a group of students working together, say in a document, the students could be warned when some student adds some contributions, informing both the fact that the contribution was done but and also who did it and having the possibility to do retroaction on that. In this way students will be more focused on learning not on the technology they are using.

Taking into account that the system supports mobile devices, students can have the possibility to interact even when they are not in front of a computer, but anywhere

as long as they are connected to a network and can receive notifications, thus enabling an efficient sharing of work and knowledge.

5.2 Problem-based learning

There has been identified the following core characteristics of problem-based learning: centred learning, learning occurs in small groups, teachers act as facilitators or guides, a problem forms the basis for organized focus and stimulus for learning problems stimulate the development and use of problem solving skills, new knowledge is obtained through means of self. We can apply again the same benefits as in the case of the constructivism. Because the learning takes place in groups that develop a concrete project, if the teacher gives a challenge to the group of students, the group will immediately be aware of it, and also the teacher can receive almost in real time (if the system is configured with an appropriate channel) about the progress of the group and in this way supervise the work in progress in the best way possible with the support of the awareness information.

5.3 Informal learning

In this methodology, students "*learn on the fly*". In this scenario we can think that our system is a kind of knowledge hub, where we have producers and consumers. Producers are all the users of the system that create objects linked with different areas, from mathematics to philosophy, from cooking recipes to jazz music… While, consumers, according to their motivations, subscribe to it and become part of different communities. Maybe they can work on a document, or be just informed about the last news that happens at the CERN. Again, our A3 implemented paradigm can help members of the community to stay tuned for any changes in the information and offer the interaction services with other members' community.

6. Conclusions and outlook

In this work we have explored the enormous possibilities brought by mobile devices to the online teamwork activities. One such important advantage is the implementation of the A3 paradigm: Anytime, Anywhere, Awareness, that is, notifying users about ongoing activity in their online workspaces. In this work, we presented an event-based system that extends web-based applications to support Anytime, Anywhere, Awareness in online workteams. The system not only can serve as informing channel but also for social support and scaffolding through fast notifications of learner's requests. In designing and prototyping the system, the efficient event detection, which is the main activity of the system, is the hardest part. For this reason the idea of a system being a notification HUB for other online learning systems seems interesting.

As nowadays the border between mobile devices and computers is each time more blurred, the system needs to be enough modular to ensure that it can deal with new communication channels and with new ways to deliver messages. Indeed, some years ago mobile devices have rather specific channels and types of messages, for

instance SMS, MMS, WAP push… and mobile browsers needed specific markup, like WML or XHTML, because the computational limitations of the devices. However, nowadays smartphones, and also feature phones, can receive the same kind of messages as a computer, for instance email, and can open web pages in the same way as a computer. This opens up many possibilities to design and implement very efficient notification systems for mobile learners.

One interesting aspect that we would like to further develop is define taxonomy of events that would fully support the A3 paradigm of anywhere, anytime, awareness in groupware systems, following work in [10,11,12].

REFERENCES

[1] Niroshinie Fernando, Seng W. Loke, Wenny Rahayu, Mobile cloud computing: A survey. Future Generation Computer Systems, Volume 29, Issue 1, January 2013, 84–106. doi:10.1016/j.future.2012.05.023

[2] Razek, M.A., Bardesi, H.J.Towards Adaptive Mobile Learning System, In Proceedings of the 11th International Conference on Hybrid Intelligent Systems (HIS), 493 - 498, 2011, IEEE CPS

[3] Richard M. Lee and Steven B. Robbins. Measuring Belongingness: The Social Connectedness and the Social Assurance Scales. *Journal of Counseling Psychology,* vol. 42, no. 2 (1995), pp. 232–241.

[4] Thomas Visser, Pavan Dadlani, Daan van Bel, and Svetlana Yarosh, Designing and Evaluating Affective Aspects of Sociable Media to Support Social Connectedness. *Proceedings of the 28th International Conference on Human Factors in Computing Systems*, Atlanta, Georgia, 2010.

[5] Xavier Rivadulla, Fatos Xhafa. Disseny i prototipatge d'un sistema d'events per a dispositius mòbils en el treball en grup. *Master's Thesis in Computer Science*. Open University of Catalonia.

[6] Tore Fjellheim, Stephen Milliner, Marlon Dumas, and Julien Vayssiere (2007). A process-based methodology for designing event-based mobile composite applications. *Data Knowl. Eng.* 61, 1 (April 2007), 6-22.

[7] Eduardo S. Barrenechea, Paulo S. C. Alencar: An Adaptive Context-Aware and Event-Based Framework Design Model. *Procedia CS* 5: 593-600 (2011). 2010

[8] Jaakko Kangasharju, Tancred Lindholm, Sasu Tarkoma (2007). XML messaging for mobile devices: From requirements to implementation. *Computer Networks* 51(16): 4634-4654

[9] D. C. Barnlund (2008). A transactional model of communication. In. C. D. Mortensen (Eds.), *Communication theory* (2nd ed., pp. 47-57). New Brunswick, New Jersey: *Transaction*.

[10] Fatos Xhafa, Alex Poulovassilis: Requirements for Distributed Event-Based Awareness in P2P Groupware Systems. AINA Workshops 2010: 220-225

[11] Fatos Xhafa, Alex Poulovassilis: Awareness in P2P Groupware Systems: A Convergence of Contextual Computing, Social Media and Semantic Web. EIDWT-2011: 14-21

[12] Alexandra Poulovassilis, Fatos Xhafa: Building Event-Based Services for Awareness in P2P Groupware Systems. 3PGCIC 2013: 200-207