2012 ASE International Conference on Social Informatics (SocialInformatics 2012) / 2012 ASE International Conference on Cyber Security (CyberSecurity 2012) / 2012 ASE International Conference on BioMedical Computing

Supporting Ubiquitous Interaction in Dynamic Shared Spaces through Automatic Group Formation Based on Social Context

Juwel Rana
Pervasive and Mobile Computing
Luleå University of Technology
Luleå-971 87, Sweden
Email: juwel.rana@ltu.se

Johan Kristiansson
Ericsson Research
Luleå-971 28, Sweden
Email: johan.j.kristiansson@ericsson.com

Kåre Synnes
Pervasive and Mobile Computing
Luleå University of Technology
Luleå-971 87, Sweden
Email: kare.synnes@ltu.se

***Abstract*—This paper investigates how the management of groups that communicate electronically, such as group formation, can be simplified based on users' context and social relations. This work builds on a framework for Aggregated Social Graphs, where each node represents the relational strength to other users. The strength of a relation is calculated by utilizing information on how we communicate using mobile phone calls, emails, and social networks in combination with additional sources of information such as from calendars. A contextual group management schema is presented where contextual parameters such as tags, locations and objects are used to prune an aggregated social graph in order to automatically form a group.**

The schema is implemented in a runtime environment based on the Distributed Shared Memory service available at Ericsson Labs. The feasibility of the proposed schema is then studied through a prototype implementation both in a web-browser and as a mobile app. The study shows that a group can be formed automatically and that a lightweight communication session then can be initiated for that group.

***Index Terms*—Social Media Intelligence, Social Graphs, Contextual Group Management, Web-based Collaboration, Mobile Applications, Shared Spaces.**

I. Introduction

The Internet is rapidly transforming our society through changing how we communicate and interact with ubiquitous communication services, where new collaborative and mobile technologies are driving a change towards spontaneous and nomadic work [1], [2]. This change has an enormous impact on groups, organizations and social networks as well as our society in general [3], [4]. However, effortless provisioning of group communication services may be a key to expand the use of effective collaboration in teams and organizations while fostering new forms of Internet behavior through social computing.

Group communication services such as Google+ Hangout, Facebook, Groupboard, and Groupon are soon ubiquitously available on anything from personal computers to smartphones and pads [5]. However, group communication services like these can be quite cumbersome to use, as significant manual efforts are required to initiate and manage groups as well as configure the communication tools [6], [7]. For example, most of the group communication services consider manual operations for participants selection in arranging meeting events. Thus, the motivation behind this work is to study how lightweight group communication can be provided based on contextual group management and (dynamic) shared spaces [8], [9].

Today's social networking services follow centralized group management which lock users in a single platform [10]. On the contrary, contextual group management involves using information from multiple platforms such as call logs, social networks, calendars, etc., to automatically filter and recommend collaborators to invite to group communication sessions. Benedikt et. al, analyse several requirements for decentralized group management in distributed systems [10]. On top of those requirements, this paper proposes using a runtime environment based on Aggregated Social Graphs (ASG), that maintains social graph information on user's local devices (contact-book, call-logs) and cloud-services (LinkedIn, Facebook, MySpace) while analyzing communication history for mining user's communication pattern [11]. This is then used to identify collaborators and inviting them to a shared space for communication and collaboration purposes.

A shared space would include tools based on the context and purpose of the group communication session, basically composing the group communication service of widgets for shared notes, shared maps, real-time chat, and so on. This paper proposes using a web-based service based on the Distributed Shared Memory (DSM)[1] at Ericsson Labs to create lightweight communication services tailored to group specific contexts [9], [8]. The runtime environment thus combines using social computing with web-based communication services to achieve as automated group management as possible.

DISCLAIMER: The work has been carried out as part of an academic research project and does not necessarily represent Ericsson views and positions.

[1]https://labs.ericsson.com/apis/distributed-shared-memory

978-0-7695-4938-5/12 $26.00 © 2012 IEEE
DOI 10.1109/SocialInformatics.2012.88

The main research problems addressed in this paper are:

- ***How can automatic group formation be implemented based on communication patterns and the context of users?***
- ***How can group collaboration be supported through automatic and dynamic composition of a lightweight communication tool?***

The rest of this paper is structured as follows: Section 2 introduces background and related work, Section 3 offers different methods for contextual group formation, and Section 4 presents a high-level architecture and describes a proof-of-concept prototype implementation. Section 5 describes evaluation of shared spaces in comparison with groupware, which is followed by discussion in Section 6 on the above mentioned research questions. Section 7 concludes the paper.

II. Background

Web technologies such as HTML5, WebRTC, JavaScript, and Ajax provide a rich platform for web-based groupware. Technologies like Google docs, Google+ Hangouts, etc., provide new opportunities to develop highly interactive web applications for real-time sharing of texts, pictures, and audio/video contents. This shows the possibility of constructing a powerful platform for group communication using web technologies.

Most of the existing collaborative applications only provide manual invitation of participants, based on access to individual contact lists, and are thus generally not considering utilizing social information. Selecting contacts from the contacts list by using drag-and-drop, such as in Google+, is not always enough for providing a suitable user experience, as it requires considerable manual operation.

DSM service provides APIs for developing web-collaboration apps (which is mentioned as coApps in the rest of the paper) where the coApps can be dynamically composed into shared spaces. This so-called (dynamic) shared space may thus include coApps for real-time sharing of multiple media, such as notes, chat and maps. Note that even if shared spaces generally makes it possible to interact, communicate and share information among group members in real-time, the main benefit of a shared space is that it can be tailored to particular communication needs. Thus the goal of the paper is to create shared spaces automatically on particular needs and enable real-time collaboration functionalities within the shared spaces.

In brief, shared spaces facilitate collaboration among the members of the shared space. Therefore, a user may have multiple shared spaces for different purposes. A shared space can be seen as an ad-hoc social network configured with selected tools for a specific tasks [12]. In other words, shared spaces are social collaboration services, where the user can exploit their social networking credentials and may have more control of their social environment for setting up group communication and collaboration.

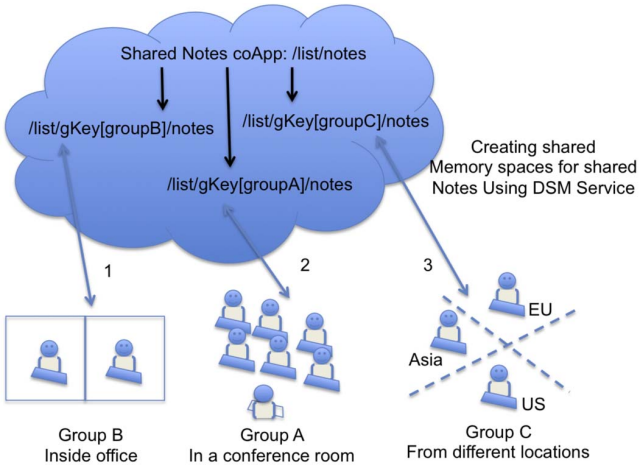

Fig. 1. Conceptual model of group based shared spaces where each of groups has an unique instance of notes coApp

Shared spaces are implemented using DSM service for concurrency control and other implementation details for the coApps. Figure 1 shows three groups *A, B* and *C* which are used for real-time collaboration by using notes coApps. The DSM service allocates an instance of the notes coApp for each of the group. All of the three groups thus has unique identifiers such as */list/groupC/notes* is for group *C*, and so on to uniquely identify memory addresses for the shared notes coApps for each of the groups. In this way, interactions are handled separately in each of the groups. Figure 2 shows the resource mapping schema proving unique identifier for each of the coApps in multiple shared spaces. Thus pressing or touching each of the shared space events such as *sharedspace/abcd* opens dedicated shared space for *sharedspace/abcd* equipped with coApps personalized for the participants of *sharedspace/abcd* event.

A. Aggregated Social Graphs

A social graph shows a user's relation to other users, normally for a single social network. An aggregated social graph merge knowledge from multiple social networks and can thus give an aggregated view of a user's relation to other users. Aggregated social graphs have been elaborately discussed in [11], proposing an Aggregates Social Graph (ASG) framework for aggregating social graphs from different social networking and communication services.

Figure 3 shows the different components and layers of the ASG framework. The Social Data Aggregator layer consists of five modules. The social data adaptor collects social data from individual's social networks such as Facebook and Twitter. The Social Data Collector aggregates the social data from multiple social networks. The unified model of interaction can be used to process the data and help rank the most valuable communication channels. Sensor and location data is collected from various devices, such as GPS devices and mobile phones, to infer context in the social strength calculation.

The top layer consists of three components. The Social Strength component that calculates the social strength between users has been previously discussed in the perspective of group communication [13], [14], [15]. The Social Graph component maintains the users' relation to other users, where each node is associated with a social strength. Lastly, the Group Formation component supports creation of groups based on the ASG framework, by pruning the aggregated social graph by contextual filtering.

This paper focuses on this last component and shows how the ASG framework can support group formation where the most appropriate contacts are invited to a shared space.

B. Related Work

Lightweight group work in everyday life by providing informal awareness, lightweight engagement, low cost meetings, artifact sharing, as well as ad-hoc membership is discussed in [8], [16]. This paper proposes contextual groups, which are formed based on users' context, tags, key words, profiles, location and so on. Contextual group means that group communication is initiated by identifying appropriate contacts based on context and then sharing activities inside the group. Some works have been done on discovering user's context based on social network data and mobile sensing [17], [18], but the task of contextual group formation has not been explored that much [15], [19].

Different group formation algorithms have recently been proposed. Michelle et. al identifies two main categories of group formation software such as over-lapping and non-overlapping groups [20]. Overlapping group formation has been studied in [6], [21], while non-overlapping group formation is considered in [20], [22]. However, only a single domain of users (such as the e-learning domain) was studied. Contextual group formation would need to consider both diverse domains and over-lapping groups for the best results. Location

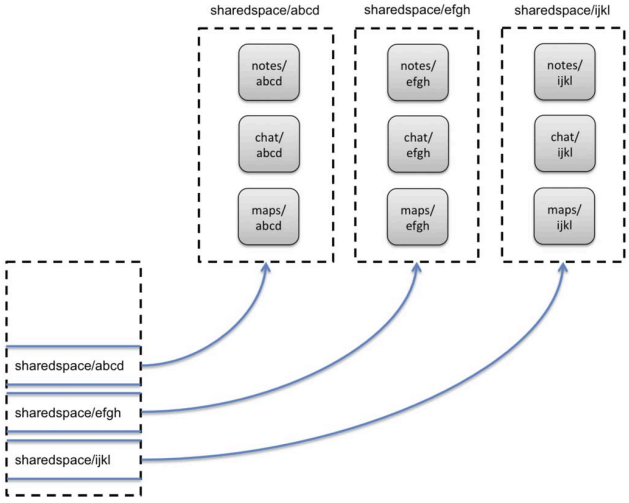

Fig. 2. Resource mapping schema: unique identifier for each of the coApps in multiple shared spaces

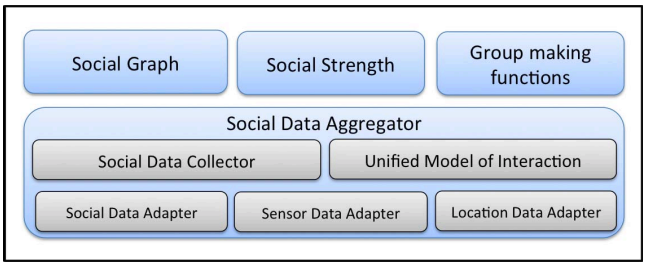

Fig. 3. Components and layers of the ASG framework

based group formation is discussed in [19]. The authors show a prototype where a group can be formed with respect to particular location. However, dynamic discovery of group participants is not covered in that work, which is considered as one of the important factors of group formation. Julian et al., shows a novel machine learning approach to discover groups in ego-centric graph [23]. The authors consider structure of social graph and profile information for group discovery, which is covered by our methods through the integration with ASG framework. Moreover, our methods consider tie-strength as a parameter of finding trust-worthy participants and multiple ego-centric graphs for organizing groups freely as well as in accordance of users' context.

Operational transformation technology has been widely used for implementation of collaboration functionalities. DSM service uses operational transformation algorithms for concurrency control. It provides an easy interface for development of collaborative applications in comparison to the opencoweb Framework[2] and Apache Wookie[3]. DSM developers need to have a unique address to transient (list://) or persistent (phash://) storage. On the other hand, Apache Wave and Apache Wookie are in "incubating stage", therefore these are lack of providing simple interface for apps development. Meteor also provides support for cooperative web apps developments [24]. This focuses on latency compensation by taking simple and clean approach for data in a distributed environment. The opencoweb framework has similar JavaScript API's in comparison with DSM, however DSM service offers four different types of memory support such as transient, transient list, persistent and persistent list. Moreover, unique memory identifier of DSM service makes the apps development more straightforward comparing with other cooperative web development framework. So far, the existing solutions do not provide automatic address calculation for dynamically collaborative group management in the web environment.

C. Motivating Scenario

This sub-section presents a motivating scenario for automatic group formation.

Ebba is planning a party together with her closest friends and is meeting two of them to organize the party. They discuss some ideas and decide to create an ad-hoc social network

[2]http://opencoweb.org/
[3]http://incubator.apache.org/wookie/

for the party. She brings up her mobile creates ad-hoc social network, where her two friends are automatically added by using matching locality and tags as indication of interest, workplace, profile match. Ebba is then presented with a list of recommended persons based on her and her two friends aggregated social graph information. They decide to filter out persons in their age and that lives close to them, then limits the network size to 20 persons. They decide to remove one person on the list as she is indicated to be travelling in her Facebook wall-post and then they activate this social network.

When Ebba arrives home she creates a shared spaces inviting the persons in the party group. She places a poll component, some media components and a chat component in the shared space for the invitation, which she then configures with adding a name, a song from her current play-list and a welcome message. She then sends the invitation to the party group. Some of her friends immediately respond by answering the poll, chatting suggestions and adding a few photos of their own. New members are invited that wants to join the party after hearing about it. At the party people use the app to continue chat and share photos.

This scenario illustrates the intended use of the shared spaces. First, Ebba and her two friends are automatically added to a shared space based on their location and activity (identified by a tag, such as in calendars or in tweets). They then create a second shared space based on their social network, communication history, location, and age. This could also be done in several iterations, thus also considering friends of friends.

The number of participants is then limited to 20, using a ranking scheme for selection. The utilization of context data, in this case calendar data, indicates the availability of participants and is used to reduce the list further and not to send out invitations unnecessarily. Finally the shared space can be used to document the event and even spread the word of it further.

III. Contextual Group Formation

This section discusses different solutions for contextual group formation, based on Tags, Locations and Objects.

A. Tag-based Group Formation

A user's interests, current preferences, and background information can be considered as tags for group formation. For example, LTU can be used as a tag to form a group of users with a background from LTU. Then different filtering parameters can be used to optimize group formation function, such as using the proximity of the contacts and the social strength between the contacts. Proximity is calculated based of group owner's current location, social strength is collected from the ASG framework. The following algorithm recommend a list of contacts, where *user* is the owner of the group, *tag* contains purposes of the group, *proximity_radius* defines the area of group participants, and *social_strength_filter* permits the participants to group formation who have significant tie-strength. By this approach, the participants who are interested in tag, available within proximity and have significant tie-strength will be recommended to form group.

```
def get_recommened_contacts(user, tag,
 proximity_radius, social_strength_filter)

 recommended_contacts = Array.new
 contacts=ASG.getFirstDegreeContacts(user)

 contacts.forEachContact do |contact|

    t = contact.interest_tag == tag
    l = contact.location == user.location
                  within proximity_radius
    s = contact.social_strength >=
                 social_strength_filter

    if (t && l && s)
       recommended_contact.add(contact)
    end
  end
  send_Invitation(recommended_contacts)
end
```

Another approach of tag-based group formation is dynamically updating the group with new-contacts of interest. In that case, every-time when the ASG service updates the user's data, the group formation function checks whether new contacts satisfy the conditions to become participants of the existing group and if satisfy, it updates the group inviting the new participants. Thus, the following algorithm invites new participants dynamically to participate in the group.

```
def get_dynamic_recommened_contacts()
 new_contacts= Array.new
 prev_invited_contacts=Array.new
 current_recom_contacts=Array.new

 ASG.onUpdate(user) do

 prev_invited_contacts =
    get_invited_contacts(user, tag)

 current_recom_contacts =
    get_recom_contacts(user, tag,
    proximity_radious, s_strength_filter)

 new_contacts= current_recom_contact\
    prev_invited_contacts
 send_Invitation(new_contacts)
end
```

B. Location-based Group Formation

Location, or proximity, was considered as a filtering parameter in the previous tag-based approach. However,

location is also significant as meeting places such as a super market, an airport, a bus or train station, a cinema-hall and so on. Here, the idea is forming temporary groups of people who are available in a location on a particular moment. This kind of group can be utilized to share digital media content related to the location itself such as advertisement, announcement, in-door maps, and so on. It can also help friends who are available at that location to meet. Here groups are formed in a recurrent way such that a user is invited when proximity constraints are satisfies. Content would thus not be sent to users that do not meet the proximity constraints.

```
def temp_group_forming()
 new_contacts= Array.new
 prev_invited_contacts=Array.new
 current_recom_contacts=Array.new

ASG.onUpdate(loc) do

prev_invited_contacts=get_invited_contacts
             (location, proximity_radius)

 current_recom_contacts =
    get_recom_users(loc,proximity_radius)

 new_contacts = current_recom_contacts \
     prev_invited_contacts
 send_Invitation(new_contacts)
end
```

C. Object-based Group Formation

In object-based group formation, group participants are not only human users, but it also considers user's smart devices such as mobile phones, tablet devices, laptop computers, televisions, refrigerators, digital projectors, cars, and/or any kind of connected devices. Thus, an object can be associated with several human or devices as participants to form a group. For example, a family can be seen as an object, which has a physical location (i.e., the home) as identity of the object and where family members (living at that home) as well as the smart devices inside that home could form a contextual group. Therefore, to form a group considering family as the object, all these entities will be invited to the group. Having such a group, different kind of scenarios on intelligent home [25] could be accomplished. For example, Kinect device as being a participant of family group, could update all participants in the group about the ongoing activities inside home.

```
def object_group_forming()
 current_rec_entities=array.new
 prev_invited_entities=array.new
 new_entities=array.new

 ASG.onUpdate(Object) do

 prev_invited_entities=get_invited_entities
    (object_loc, proximity_radius, object)

  current_rec_entities=get_rec_entities
    (home_loc, proximity_radius, object)

  new_entities=current_rec_entities \
          prev_invited_entities
  send_Invitation(new_entities)
 end
```

D. Factors of Different Group Formation Methods

There are four different methods for contextual group formation presented above, needs to be compared to identify different circumstances for the suitability of these methods. The comparison is performed considering following factors:

Contextual information: Contextual information is naturally a vital factor for contextual group formation. Tags, locations, relationship and so on are example of contextual information used for group formation.

Group candidates: It is not always enough to only consider human users as group candidates, as it also is possible to improve group communication by including smart devices as the participants of a group. For example, a car can be member of a group as it may interact with the owner to start the heater an hour before the owner leave the office in winter season. At same time, the owner may notify his/her family members to get ready for a dinner, so that when he/she arrives at the home, get things ready to drive towards a restaurant. For sure the calendar element of the restaurant is added as your group member, which enable the car owner to book a table in time in the restaurant.

Form of groups: Two types of groups are considered. Long-term, where the participants may evolve recurrently and communicate in groups that may remain over time such as for a football team. Short-term, where a group may be formed for temporary purposes such as just to establish a relationship with the current shopping-mall that a user is visiting to enable the user to receive all the discount offers and the location of friends in the mall.

Purposes of groups: Generally, there are public and private purposes for forming a group, while social and productivity purposes are also important purposes for forming groups. Social purposes are for instance a user's intent to increase interaction among his friends and family. Productivity purposes include adding devices to a group, to increase productivity for a certain task.

The Table 1 contains comparative analysis of different contextual group formation methods. The table shows that it is hard to define which methods are better comparing with each other. The comparison indicates different methods of group formation could be useful for creating contextual group on different purposes.

E. Group Invitation Approaches

Different methods for contextual group formation were discussed above. However, those methods do not provide de-

TABLE I
COMPARATIVE ANALYSIS OF CONTEXTUAL GROUP FORMATION FUNCTIONS

Group formation methods	Contextual information	Group candidates	Form of groups	Purposes
Tag based group formation	Social strength, location	Human users	Long-term	Private
Tag based concurrent group formation	Social strength, location	Human users	Long-term	Public
Location Based group formation	Real-world artifacts	Human users	Temporary	Social
Object Based group formation	Relationship	Devices and Human users	Long-term, Temporary	Productivity

tails about sending invitation notification to the recommended contacts. For this purpose, different invitation mechanisms such as Apple's push notification [26], Android notification [27], W3C Web notification [28], web event streaming, HTTP pooling, emails or SMS can be used as solutions for invitation notification. A brief description of each of these methods is given below:

Apple push notification: The Apple push notification service can be used to notify recommended contacts to participate in shared spaces. Generally, Apple's push notification server receives notification from the providers, in this case the invitation service forwards the notification to the appropriate devices.

Android notification: In Android-based devices, the Notification Manager registers providers and on receiving notification from the registered providers, it passes the notification combing with the proper intent of the notification. Therefore, Android notification is another possibility of sending invitation notification to group participants.

W3C Web notification: Using W3C Web notification, HTML5 compatible web-browsers are able to receive web notifications from certain web apps. This approach could also be useful for sending invitations. However, to apply this approach certain requirements must be accomplished such as e-id of the user and so on.

HTTP polling: Through HTTP polling, client applications send request messages to the server for new invitation events, and wait until it gets a response from the server. This method is not very efficient due to its synchronous nature. Comparing with Android notification, HTTP polling out-performs as it is synchronous by nature and does not push notifications.

E-mail or SMS: Invitation notification can also be sent to users via e-mail or SMS services, which is the traditional approach of invitation (and the most reliable ones).

In the next section, we discuss the architecture of shared spaces, which utilizes the contextual group formation functionalities.

IV. IMPLEMENTATION

The runtime environment for dynamic shared spaces is divided into three layers, as depicted in Figure 4, where the first layer is an application layer targeting foremost mobile web applications. The applications can be deployed as mobile apps for smart mobile devices or run in a web browser.

The second layer provides the main functionalities of the runtime environment and is defined as a Web API for application developers. The Web APIs are REST and SOAP messaging compatible, therefore it is easier for the application

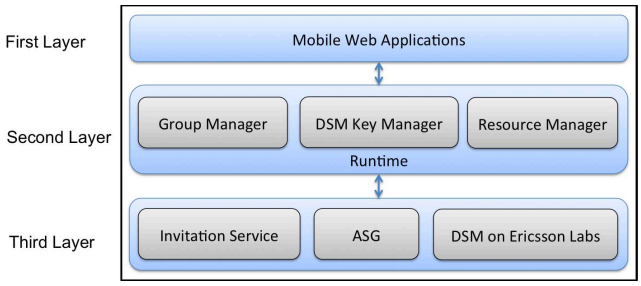

Fig. 4. Different Layers of the Shared Spaces Runtime

developers to utilize these APIs in their application development processes. The Group Manager performs the tasks related to group formation and discovery. The DSM Key Manager generates unique keys and the Resource Manager assigns dedicated coApps for new groups. DSM service requires unique keys for memory management and concurrency control of the particular coApps.

The third layer contains an Invitation Service that invites participants to a group while considering multiple ways of invitation. The DSM Service provides underlying technologies for managing concurrency for all coApps. The ASG service is a central part of this layer, as it supports the Group Manager component to form groups and the Invitation Service to invite participants.

A. Address Key Generation

As mentioned above, the DSM service requires a unique address for each of the contextual group. The group address key is therefore needed to represent a group with a shared space together with the coApps. In our approach, the group key is an UUID identifier version 3[4], which is used to address all resources in the shared space for a particular group or shared space. More specifically, it will be used for managing real-time syncing of shared elements such as widgets for chatting and media sharing through the DSM service. The group key is also used to generate a unique Shared Space Access Point (SSAP) for proving access to the shared space.

B. Dynamic Shared Space Initiation

The sequence to form a group and create a dynamic shared space is depicted in Figure 5. Group formation is done by discovering collaborators (members of the group) by using context, for instance described as a tag, and then generating the group key to create the group. Creating a group leads to

[4]http://en.wikipedia.org/wiki/Universally_unique_identifier

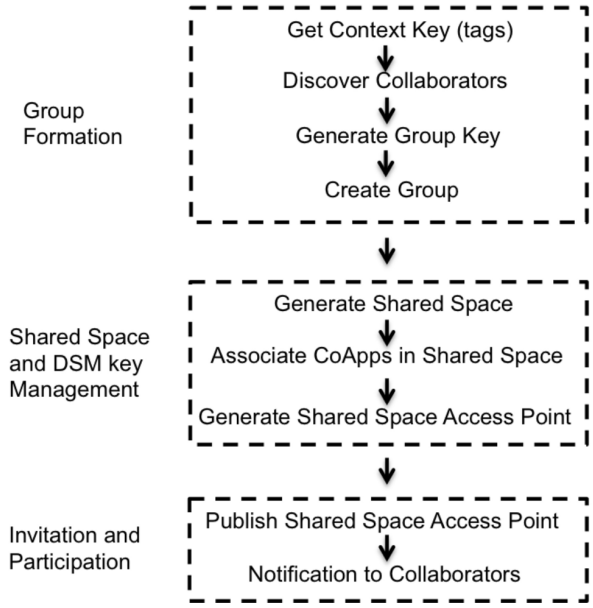

Fig. 5. Group formation and dynamic shared space creation

generation of a shared space, which then is associated with coApps and addressed by SSAP identifier. Lastly, the SSAP is used to invite the collaborators.

The pseudo code below creates a new-shared space with coApps as described above. The code uses templates of the coApps for replicating the resources to form a shared space for a new group. All these templates contain a function called DSMonReadyF, which is dynamically updated in each of the coApps using the group address key. Finally, it associates all coApps to the shared space for compiling coApps to a single shared space interface. The interface is shared through SSAP among the participants for group communication and collaboration.

```
Shared spaces and coApps generation

OUTPUT: sharedSpacegroupKey
Require: coApps template, groupKey

 groupKey = UUIDgenerator();
 for id = 0 ! coApps:length do
    Generate Resources for coApps[id]
    Associate groupKey to coApps[id]
    Update DSMonReadyF to coApps[id]
    Associate coApps[id] to
                 sharedSpacegroupKey
 end for
Publish sharedSpacegroupKey
```

The above pseudo code shows DSM address key management scheme and resource allocation approaches for a shared space for performing group collaboration. The purpose of providing individual address keys for each shared space is to be able to dynamically join to that shared space. For this purpose, the pseudo code generates a group address key and then it allocates all the resources with the same key in the DSMonReadyF functions. Finally it provides a SSAP to the shared space for inviting collaborators for performing collaboration using the shared space.

C. Service Integration

Figure 6 illustrates interactions among different services for setting the runtime environment of shared spaces. From the user's device interface, the user provides requests of forming groups using keywords to the Group Manager (1). The Group Manager accesses cloud services, such as the Group Discovery Service, and DSM Address Key Manager, and processes the user's requests. The Group Discovery Service provides recommended contact lists (2) and also receives the group access key from DSM Key Manager (3) for the requested group.

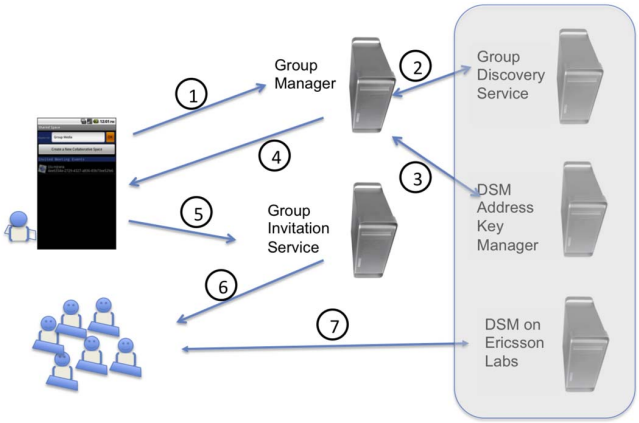

Fig. 6. Service Interaction

The Group Manager then accesses call logs, location and contact lists for refining and prioritizing the participants list. After that the Group Manager sends back the recommended list of participants to the user interface (4). The user can then review and modify the list before sending the final list of participants to the Invitation Service (5). If the user does not like reviewing the list of recommended contacts, automatic invitation will allow to sent the list directly to Invitation Service. The Invitation Service distributes the invitation to the participants using different communication tools such as e-mail, SMS or Tweets (6). The invitation message contains the shared space access point of the resources for this newly formed collaborative environment. Finally, the participants are able to access the shared space, through the DSM service.

D. Proof-of-concept prototype

The prototype is implemented to prove two major activities. The first activity is to discover participants and then invite them to join in a newly created shared space event. The second activity is participation in that shared space event for performing collaborative task, at present each of the shared

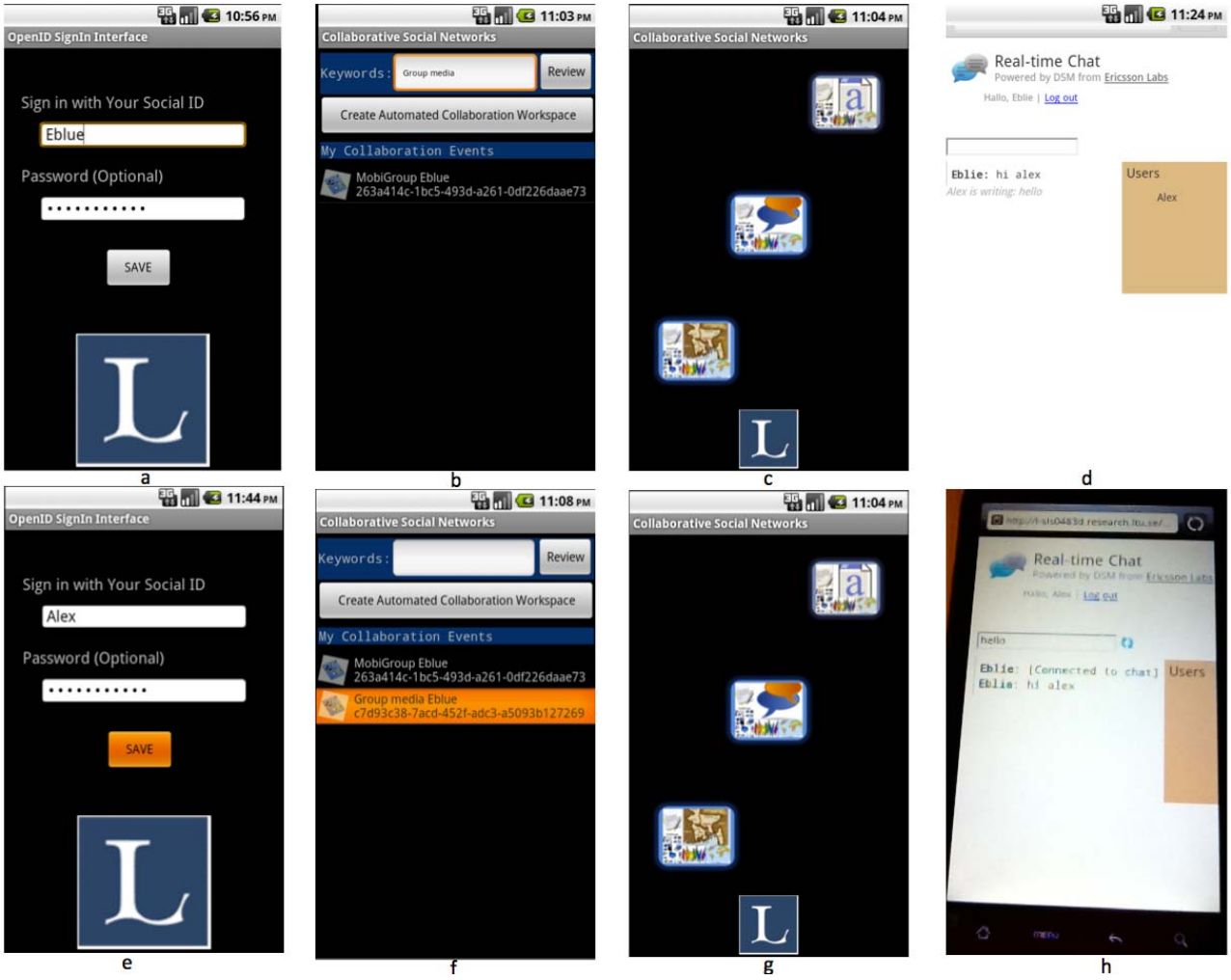

Fig. 7. Creating shared Space, Inviting Collaborators and Performing Collaboration Using Shared Spaces Prototype

space events contains a shared note coApp, a real-time chatting coApp, a map-sharing coApps.

The prototype provides a sign in feature to the users for using the apps (Figure 7a). After sign in, the user may type a keyword as the context of forming contextual group (Figure 7b). Based upon the context keyword, the recommended participants list can be reviewed by using Review button. If the user wants to invite all the participants without reviewing, it presses the ”Create Automated Collaborative Workspace” button. By pressing that button the user gets invitation message to join at the shared space to that particular group where different tools are available for performing collaborative activities (Figure 7c). For example, the user may initiate real-time chat among the invited participants by using the real-time chatting tool (Figure 7d).

When another user receives a new invitation message, it appears in the list of shared space events as a new invitation event (Figure 7f). When a user touches a particular invitation event, then the user is connected to that shared space session, and is being able to collaborate with other participants at that session (Figure 7g). In this case as shown in Figure 7, Eblie invites Alex to perform collaboration for Group media project activities. After invitation and creation group media shared space, if Alex selects real-time chat tool, then Alex will be able to communicate with Eblue.

V. Evaluation

This section provides comparative evaluation among different groupware with respect to shared spaces prototype. Some of the groupware services already cover large set of group communication tools such as audio/video support, shared editing and so on. Many of those tools are highly scalable in nature and perform well in practise such as Skype, Google+ hangouts. Thus, the evaluation is on upcoming challenges in group communication considering the fact of social recommendation. We identify couple of future requirements in group communication such as automatic group invitation, social context adaptation, and so on [18]. Based on

TABLE II
COMPARATIVE ANALYSIS

GroupWare	Automatic invitation	Context adaptation	Global contacts adaptation	coApps integration
Google Hangout	-	-	-	x
AdobeConnect	-	-	-	-
Skype	-	-	x	x
Shared Spaces	x	x	x	x

such metrics, the comparative evaluation is performed at the first place. Thus the metrics are as follows:

Automatic invitation: Groupware tools may invite potential participants for forming the group via automatic discovery of participants and sending invitation message to them. In this way, it avoids unnecessary manual inputs for setting up a group for collaboration.

Social context adaptation: By adapting social contexts, groupware tools may able to identify appropriate contacts based on the contextual information of the contacts from heterogeneous communication sources.

Global contacts adaptation: By adapting global contacts from different communication sources, groupware tools may able to form groups in a large perspective by exploiting a user's openID or mobile phone number. In a way, it reduces sign-in operations to get contacts from heterogeneous group communication sources.

coApps integration: Integrating coApps within the groupware, maximizes the number of interactions among the group members. For example, shared object annotating tools may simplify collaboration tasks which is widely using now a days for sharing social recommendations for movie and music selections in services such as Netflix[5], and Spotify[6].

Table 2 shows a comparative analysis of dynamic shared spaces considering Google+ Hangouts, Adobe-Connect and Skype. The invitation process in Google+ Hangout is manual and needs Google ids for setting up a group. Creating a hangout and email out the URL or post it for everybody to see and join does not solve contextual group formation problem. Moreover, it does not adapt users' emails by using context and global contacts. However, Google+ Hangouts provides coApps for media streaming and distributing. Adobe-Connect is also lack of automatic invitation. It provides a URL for every collaboration sessions and participants need to get in the particular collaboration session by doing manual operations. It could be better by adapting context and global contact or integrating with diverse coApps. Google+ Hangout supports importing coApps and has a good API for creating new coAps. Skype do not support automatic invitation and context adaptation but it considers global contact adaptation. For example, Skype platform is integrated with MySpace and Facebook social networking services to facilitate communication with facebook contacts or Myspace contacts using Skype services. It also supports coApps such as IDroo [29], which may be used for collaborative meetings. In general, shared spaces are able to invite group members both as automatically or by using a review process. It supports social context adaptation, global contact adaptation as well as it supports coApps as main collaboration tools.

[5]www.netflix.com

[6]http://www.spotify.com/

VI. DISCUSSION

This section discusses the research questions addressed in this paper. Then, it describes ubiquitous interactions in the context of group collaboration.

How can automatic group formation be implemented based on communication patterns and the context of users?

This paper presents four methods for automatic group formation by using tags, locations or objects in combination with the ASG framework. This enables group formation to consider social parameters and the context of users, though creation of a social graph with information from several social networks and communication tools (pattern identifies through analysis of call logs, etc.) where the social graph then is pruned with contextual parameters.

The prototype implementation shows that these methods are feasible, both when utilized for mobile and Web applications. The usage of context is still very simplistic, but is enough to prove that the social graph can be pruned by considering multiple types of contextual data.

How can group collaboration be supported through automatic and dynamic composition of a lightweight communication tool?

Groups can be automatically formed, as described above. The next step is then to create the group communication environment as automatic as possible, while supporting tools to be dynamically included on a need basis. This work utilizes the DSM service that supports creation of shared spaces, which are lightweight applications that can be composed into a tailored service for particular needs.

The conceptual prototype indicates that the full chain of events from forming a group to select specific apps for collaboration can be automatic. It also shows that the runtime environment is possible to run apps both in mobile devices and in web browsers.

A. Supporting Ubiquitous Interaction

This paper focuses on supporting ubiquitous interaction by facilitating group collaboration, for which three tasks are important. The first task is finding appropriate collaborators for the group, the second task is allocating resources to initiate collaboration and the third task is inviting participants using proper communication channels. All these tasks are accomplished by shared spaces prototype.

The shared space prototype shows that today's social networking services can be utilized to form lightweight shared spaces to perform specific collaboration task. In this work, we show that shared spaces are not an alternative of social networking services, rather it builds on the social information from the networks to make lightweight group communication easier and more efficient. It is found that the Ericsson Labs DSM service may play an important role for developing collaborative applications, tailored to particular needs.

VII. CONCLUSIONS

The paper presents a runtime environment for creating lightweight shared spaces for communication and collaboration needs. The proof-of-concept prototype shows that contextual group could be formed in a disruptive way i.e., outside of the boundary drawn by state of the arts social networking services without losing users' contacts and contents in those services. Precisely, the paper shows different functions for automatic group formation, and adaptation of social computing in-group specific shared spaces. The proposed runtime environment makes use of aggregated social graphs for group management, where information on how we communicate is gathered from call logs, calendars, social networks, etc.

In future work we will continue to performance analysis of contextual group formation functions, measure resource efficiency of shared space prototype based on qualitative and quantitative data collected from user study.

VIII. ACKNOWLEDGEMENTS

This work was funded by the Research Area Multimedia Technologies at Ericsson Research, where Stefan Hkansson has been vital for the direction and execution of the research. We thank to Professor Peter Parnes for the feedback in this work. The work was also supported by the Centre for Distance-spanning Technology (CDT) and by the Satin-II research project partly, funded by the European Regional Funds. The European Institute of Innovation and Technology (EIT) ICT Labs also supported the work, through the action lines ICT-mediated Human Activity, Future Media and Content Delivery.

REFERENCES

[1] A. Kaplan, "If you love something, let it go mobile: Mobile marketing and mobile social media 4x4," *Business Horizons*, 2011.

[2] D. Zhao and M. Rosson, "How and why people twitter: the role that micro-blogging plays in informal communication at work," in *Proceedings of the ACM 2009 international conference on Supporting group work*. ACM, 2009, pp. 243–252.

[3] A. Lenhart, *Social media & mobile internet use among teens and young adults*. Pew Internet & American Life Project, 2010.

[4] E. F. Churchill, "The (anti) social net," *interactions*, vol. 17, no. 5, pp. 22–25, Sep. 2010. [Online]. Available: http://doi.acm.org/10.1145/1836216.1836222

[5] S. McNulty, *The Google+ Guide Circles, Photos, Hangouts, and More*.

[6] M. Wessner and H.-R. Pfister, "Group formation in computer-supported collaborative learning," in *Proceedings of the 2001 International Conference on Supporting Group Work*, ser. GROUP '01. New York, NY, USA: ACM, 2001, pp. 24–31. [Online]. Available: http://doi.acm.org/10.1145/500286.500293

[7] D. Gurzick, B. Landry, and K. F. White, "Alternate reality games and groupwork," in *Proceedings of the 16th ACM international conference on Supporting group work*, ser. GROUP '10. New York, NY, USA: ACM, 2010, pp. 303–304. [Online]. Available: http://doi.acm.org/10.1145/1880071.1880121

[8] Y. Sun and S. Greenberg, "Places for lightweight group meetings: the design of come together," in *Proceedings of the 16th ACM international conference on Supporting group work*, ser. GROUP '10. New York, NY, USA: ACM, 2010, pp. 235–244. [Online]. Available: http://doi.acm.org/10.1145/1880071.1880111

[9] M. Hoffmann and T. Sumner, "Supporting distributed participatory design with lightweight communication tools," *Technical Report CU?CS?2001*, 2001. [Online]. Available: http://www.sociotech-lit.de/HoSu01-SDP.pdf

[10] B. Elser, G. Groh, and T. Fuhrmann, "Group management in p2p networks." in *ICCCN*. IEEE, 2010, pp. 1–8.

[11] J. Rana, J. Kristiansson, and K. Synnes, "Enriching and simplifying communication by social prioritization," in *Advances in Social Networks Analysis and Mining (ASONAM), 2010 International Conference on*. IEEE, 2010, pp. 336–340.

[12] R. Juwel, K. Johan, and S. Kare, "Dynamic media distribution in ad-hoc social networks," in *2nd International Conference on Social Computing and its Applications (SCA2012)*, 2012, pp. 546–553.

[13] J. Kristiansson, J. Hallberg, R. Juwel, K. Synnes, and S. Håkansson, "Social data ranking and processing," Dec. 28 2010, uS Patent App. 12/979,493.

[14] A. Ankolekar, G. Szabo, Y. Luon, B. A. Huberman, D. Wilkinson, and F. Wu, "Friendlee: a mobile application for your social life," in *MobileHCI '09: Proceedings of the 11th International Conference on Human-Computer Interaction with Mobile Devices and Services*. New York, NY, USA: ACM, 2009, pp. 1–4.

[15] R. Grob, M. Kuhn, R. Wattenhofer, and M. Wirz, "Cluestr: mobile social networking for enhanced group communication," in *GROUP '09: Proceedings of the ACM 2009 international conference on Supporting group work*. New York, NY, USA: ACM, 2009, pp. 81–90.

[16] S. Greenberg, K. Tee, and C. Gutwin, "Artifact awareness through screen sharing for distributed groups," 2008. [Online]. Available: http://hdl.handle.net/1880/46642

[17] N. Lane, Y. Xu, H. Lu, A. Campbell, T. Choudhury, and S. Eisenman, "Exploiting social networks for large-scale human behavior modeling," *Pervasive Computing, IEEE*, vol. 10, no. 4, pp. 45 –53, april 2011.

[18] G. Groh, *Contextual Social Networking*. Habilitation Thesis in Computer Science, 2012.

[19] R. Lübke, D. Schuster, and A. Schill, "Mobilisgroups: Location-based group formation in mobile social networks," in *PerCom Workshops*, 2011, pp. 502–507.

[20] R. I. González-Ibáñez and C. Shah, "Group's affective relevance: a proposal for studying affective relevance in collaborative information seeking," in *Proceedings of the 16th ACM international conference on Supporting group work*, ser. GROUP '10. New York, NY, USA: ACM, 2010, pp. 317–318. [Online]. Available: http://doi.acm.org/10.1145/1880071.1880128

[21] A. Inaba, T. Tamura, R. Ohkubo, M. Ikeda, and R. Mizoguchi, "Design and analysis of learners' interaction based on collaborative learning ontology," in *Proceedings of EuroCSCL01*, 2001, pp. 308–315.

[22] C. Shah and G. Marchionini, "Awareness in collaborative information seeking." *JASIST*, pp. 1970–1986, 2010.

[23] M. Julian and L. Jure, "Learning to discover social circles in ego networks," in *NIPS*, 2012.

[24] (2012, Aug.) Meteor. [Online]. Available: http://meteor.com/

[25] J. Gómez-Romero, M. A. Serrano, M. A. Patricio, J. García, and J. M. Molina, "Context-based scene recognition from visual data in smart homes: an information fusion approach," *Personal Ubiquitous Comput.*, vol. 16, no. 7, pp. 835–857, Oct. 2012. [Online]. Available: http://dx.doi.org/10.1007/s00779-011-0450-9

[26] (2012, Aug.) Apple push notification service. [Online]. Available: http://developer.apple.com/

[27] (2012, Aug.) Android notifications. [Online]. Available: http://developer.android.com/guide/topics/ui/notifiers/index.html

[28] (2012, Aug.) W3c web notifications. [Online]. Available: http://www.w3.org/TR/notifications/

[29] (2012, Aug.) Idroo. [Online]. Available: http://www.idroo.com/