Group Decis Negot (2011) 20:563–592
DOI 10.1007/s10726-011-9258-2

A Patterns System to Coordinate Mobile Collaborative Applications

Andrés Neyem · Sergio F. Ochoa · José A. Pino

Published online: 29 May 2011

Abstract Advances in wireless communication technologies and mobile computing devices open new possibilities to carry out computer-supported mobile collaborative work. However this opportunity brings also a number of challenges to designers, since collaborative applications supporting mobile activities involve requirements which are not present in stationary collaboration scenarios. For example, mobile collaborative applications should not use centralized components because it jeopardizes the autonomy required by mobile workers. In order to help designers to deal with these new requirements, this article presents a patterns system focused on the coordination support required for mobile collaborative work. Such patterns represent reusable designs that help reduce design risks, cost and time. The article also presents three mobile collaborative applications in which proposed patterns were included in their respective designs.

Keywords Coordination patterns · Patterns system · Mobile collaborative applications · Mobile collaboration

A. Neyem (✉)
Department of Computer Science, Pontificia Universidad Católica de Chile,
Av. Vicuña Mackenna, 4860, Santiago, Chile
e-mail: aneyem@ing.puc.cl

S. F. Ochoa · J. A. Pino
Department of Computer Science, Universidad de Chile, Av. Blanco Encalada, 2120, Santiago, Chile
e-mail: sochoa@dcc.uchile.cl

J. A. Pino
e-mail: jpino@dcc.uchile.cl

1 Introduction

Collaborative systems provide support for groups of persons while they communicate and coordinate their activities to reach a common goal (Ellis et al. 1991). Both communication and coordination are required to support collaboration. Communication refers to the information exchange among cooperating group members, and coordination relates to bring group activities into proper relation.

Building these collaborative systems has always been a complex undertaking because it involves issues which are irrelevant while developing single-user systems, such as human-to-human communication and awareness, group dynamics, users' social roles, group memory, and other organizational and social factors (Schümmer and Lukosch 2007). Trying to deal with these issues, the CSCW community has proposed several communication and coordination solutions for stationary collaboration scenarios (Guerrero and Fuller 2001; Schümmer and Lukosch 2007; Avgeriou and Tandler 2006). These solutions support effective face-to-face (Moran 2000; Tan et al. 2000) and virtual team work (Nunamaker et al. 2009).

Advances in wireless communication and mobile computing are opening up several opportunities to carry out computer-supported mobile collaborative work (Schaffers et al. 2006; Neyem et al. 2008; Dutta and Mia 2009). However, this new collaboration scenario has also brought various challenges to software designers. The challenges include a number of requirements which are usually present in mobile collaboration, but are not in stationary work scenarios. For example mobile collaborative applications must be as autonomous as possible because their capability to access remote shared resources is uncertain and it depends on the user's location and work context (Neyem et al. 2007). Moreover, collaborators must use small devices when the work involves high mobility (Tarasewich 2003); therefore the groupware services must be lightweight because small devices have limited computing power and hardware resources.

The main causes of these new requirements are two: (1) the communication media supporting the collaboration process is not always present and its availability and communication capability are usually unpredictable (Neyem et al. 2007); and (2) mobile work is essentially loosely-coupled, and it involves sporadic on-demand collaboration processes (Pinelle and Gutwin 2005). These particularities make unsuitable most of the general solutions designed for stationary work scenarios. Therefore, mobile groupware designers must conceive new communication and coordination strategies to support mobile collaborative work.

This article proposes a design patterns system (Buschmann et al. 2007) to support typical coordination mechanisms, but considering the particularities of the mobile collaborative work. These patterns are general reusable solutions to a commonly occurring problem in software design, which is present in a certain application context (Gamma et al. 1995). The patterns system helps designers to model the coordination services (e.g. users' and session management) required to support mobile collaboration, because they provide reusable and tested solutions. The reuse of these designs relieves developers on thinking the design of these services, and it allows them to focus on the functionality the application must expose to mobile users.

Next section describes a list of general requirements which must be considered by any solution supporting mobile collaborative work. Section 3 presents the related work.

Section 4 describes the proposed patterns system. Section 5 presents and discusses evaluation process of the patterns system. Finally, Sect. 6 presents the conclusions and future work.

2 Mobile Collaboration: General Requirements

Mobile collaboration has increasingly become an important issue in CSCW. Some application areas of mobile collaboration are the following ones: healthcare (Tentori and Favela 2008), education (Milrad and Spikol 2007), productive activities (Ochoa et al. 2008), mobile commerce (Tarasewich 2003) and emergency support (Monares et al. 2009).

However, efforts to understand the implications that mobile work and mobile collaboration have on collaborative applications design are still a research subject (Herskovic et al. 2009; Hislop 2008; Milrad and Spikol 2007). Mobile groups are highly varied in the ways they organize work, in the physical dispersion of mobile workers, and in the chosen styles of collaboration among workers (Andriessen and Vartiainen 2006; Luff and Heath 1998; Wiberg and Ljungberg 2001). While trying to make sense of this diversity, there exist efforts to describe and classify these variants by focusing on specific types of mobility (Kristoffersen and Ljungberg 2000), types of physical distributions occurring in mobile groups (Luff and Heath 1998), and levels of coupling among mobile collaborators (Churchill and Wakeford 2001; Pinelle and Gutwin 2005).

These research studies show mobile workers are those who have to work out of office, move around locally or remotely. Their activities are typically performed on an uncertain timeframe and in diverse locations. Also, workers carry with them a "portable office" with constrained resources. Provided the actors' mobility, the interaction scenario for a particular mobile worker is uncertain and it also can change in a short time period. Moreover, it is not possible to ensure availability of communication media when mobile workers decide to collaborate. These particularities of the mobile collaborative work impose several requirements on mobile collaborative applications. The authors have summarized these requirements in the following ones:

Autonomy. Collaborative mobile applications should work as autonomous solutions in terms of access to shared resources (e.g. communication and coordination services, and shared data) (Pinelle and Gutwin 2005). This is because the mobile worker cannot be sure s/he will be able to get communication support in his/her workplace at the instant s/he decides to collaborate. In addition, if s/he gets wireless communication support, the high disconnection rate and the short communication threshold jeopardize the access to remote shared resources (Neyem et al. 2007). Therefore, mobile collaborative solutions must be as autonomous as possible.

Interoperability. Since mobile work could include unknown persons trying to do casual or opportunistic collaboration, their mobile collaborative applications should offer interoperability of data and services (Neyem et al. 2008). If two or more mobile workers meet and decide to collaborate, then the heterogeneity of data, services and devices should not represent a barrier to carry out the collaboration process.

This heterogeneity is usual in mobile work; therefore mobile collaborative solutions must be as interoperable as possible.

Context-awareness. Since users are on the move to carry out their activities, their work context can frequently change (Alarcon et al. 2006). Some variables, such as communication/users availability, network topology and access to Internet/remote servers will change from one location to the next one. A mobile worker is not able to detect these changes in a simple way; therefore the collaborative application has to provide this information to the user in real-time. There is also other contextual information the mobile worker needs to know because it could affect his/her activities; for example remaining battery life or peripheral activation status. The strategy to deliver this information is a matter of design, and it will depend on each case. However, a simple and fast access to this information could affect the mobile workers' collaboration capability and productivity.

Awareness of users' reachability. This requirement can be seen as part of the context-awareness requirement; however it needs a particular consideration due to its relevance for mobile collaborative work. Since mobile workers collaborate on-demand (Pinelle and Gutwin 2005), they need to know when a potential collaborator is reachable. This reachability could refer to a collaborator's physical or virtual presence. Physical reachability involves locating a user into a physical environment and it requires location mechanisms (Castro and Favela 2008). Virtual reachability implies mainly communication capability with the potential collaborator through a digital network. Herskovic et al. report awareness of users' physical location increases the collaboration opportunities in hospital work (Herskovic et al. 2009). Farshchian reports users' virtual reachability promotes informal collaboration in several scenarios (Farshchian 2003). A psychological study carried out by Rettie indicates mobile users select, in real-time, the collaboration variant (i.e. physical or virtual) according to the following priorities: first, the complexity of the activity to be performed, and second, the collaborators' type of availability (Rettie 2005). Summarizing: awareness of users' reachability promotes (physical or virtual) collaboration during mobile work.

Use of hardware resources. Collaborative mobile applications should be able to operate in heterogeneous devices. Handheld devices with constrained hardware resources are the typical equipment to be used; examples are smartphones, mobile internet devices (MIDs) or personal digital assistants (PDAs). The ideal situation is to count on lightweight communication and coordination services to support the collaborative work, but this is not always possible (Alarcon et al. 2006). However it is feasible to activate on-demand services as a way to reduce the hardware resources overuse. The solution proposed by the collaborative applications designers to deal with this requirement will directly affect the interoperability, and consequently, the mobile workers' collaboration capability.

Low coordination effort. Mobile collaborative work involves tasks, which often are strongly partitioned among workers (Andriessen and Vartiainen 2006). This partitioning minimizes coordination demands and it allows people to work autonomously and in parallel. Although the coordination process among mobile workers is usually carried out through on-demand collaboration activities (Pinelle and Gutwin 2005), ideally it should tend towards an unattended process as much as possible. Unattended (i.e. low cost) coordination activities reduce mobile workers' cognitive load and improve

the shared data availability. Transparent mechanisms help improve mobile workers' productivity and promote collaboration among them. Otherwise, if the coordination process requires effort from mobile workers, they will collaborate just when it is absolutely required and not when they have a chance. Therefore the coordination effort for mobile workers' activities must be as low as possible with the goal of promoting collaboration.

The software industry has seen the design reuse as a way to deal with recurrent requirements characterizing a problem. It has been shown to be a good idea since the first design patterns were defined by Gamma et al. (1995). As a consequence, this article proposes a patterns system to deal with the recurrent requirements presented in this section. IDC has predicted the number of worldwide mobile workers will reach one billion by 2011 (BNet 2008). This number roughly corresponds to 30% of the worldwide task force; therefore, appropriately dealing with these requirements could have an important impact on mobile work.

3 Related Work

Jørstad et al. have proposed a set of generic coordination services for distributed (but stable) work scenarios (Jørstad et al. 2005). These services include locking, presentation control, user presence management and communication control. There are several other researchers who have also proposed similar solutions to support coordination on wired networks (Arvola 2006; Avgeriou and Tandler 2006; Guerrero and Fuller 2001; Schümmer and Lukosch 2007). However, the contextual variables influencing the collaboration scenario (e.g. communication instability and low feasibility to use servers) and the mobile work (e.g. use of context-aware services and support for ad-hoc coordination processes) make such solutions unsuitable to support mobile collaboration.

There are also proposals to deal with coordination services in wireless networks; however such proposals assume signal stability and use centralized components (Essmann and Hampel 2005; Licea 2006). These proposals were designed to support micro-mobility or work in environments completely covered by a wireless network (i.e., using access points). Like the previous case, these solutions do not satisfy the requirements stated in Sect. 2.

Research literature reports several experiences describing the use of collaborative mobile applications (Molina et al. 2008; Monares et al. 2009; Tarasewich 2003; Tentori and Favela 2008; Zurita et al. 2008). Although some of these applications are fully-distributed and seem to match the stated requirements, they do not describe or evaluate the strategies used to support coordination in mobile collaborative scenarios. Thus, the potential design solutions cannot be evaluated when they are formalized through design patterns or reused in future applications. Schümmer and Lukosch argue that collaborative systems reuse should focus on design reuse rather than code reuse (Schümmer and Lukosch 2007).

There are also a number of collaboration patterns which can be studied in order to try to understand the coordination mechanisms encapsulated behind those collaboration models (de Vreede and Briggs 2001; Herrmann et al. 2003; Pinelle and Gutwin 2006; Zurita et al. 2008). The analysis of these patterns could provide some insight

on how to coordinate mobile workers' activities; however such research effort is still pending. Next section presents the design patterns system we propose to model the coordination services that support mobile collaboration.

4 Patterns System

Roberts and Johnson proposed a methodology to identify patterns and provide reusable designs implementation for a family of applications in specific domains (Roberts and Johnson 1996). Such methodology can also be used to evaluate potential design patterns. The process involves developing several applications in different scenarios for a particular domain to capture the commonalities shared by the various solutions to a single problem (Fig. 1). The first step is to distinguish between recurring solution ideas and those which are singular. The singular ones may be tightly bound to their application and thus they cannot be considered as general patterns.

On the other hand, we must consider that a pattern generally has relationships to other patterns and we need to identify these interdependencies in order to define a patterns network (Buschmann et al. 2007). Then, this patterns network forms a *system* (or patterns system), which provides design knowledge into a specific application domain. We use the term *patterns system* instead of *patterns language* because the proposed patterns list is not necessarily complete (Buschmann et al. 2007). Next section presents the architectural pattern representing the framework of this proposal. Then, Sect. 4.2 describes the coordination patterns composing the system. The structure used to represent each pattern is the following one: name, context, problem, solution and related mobile collaboration requirements. Section 5 presents and discusses three case studies where these patterns were applied.

4.1 CrossLayer

Context. Collaborative applications typically separate functionality in three basic concerns: communication, coordination and collaboration. Each layer provides services (modeled based on the proposed patterns) and related meta-data. These

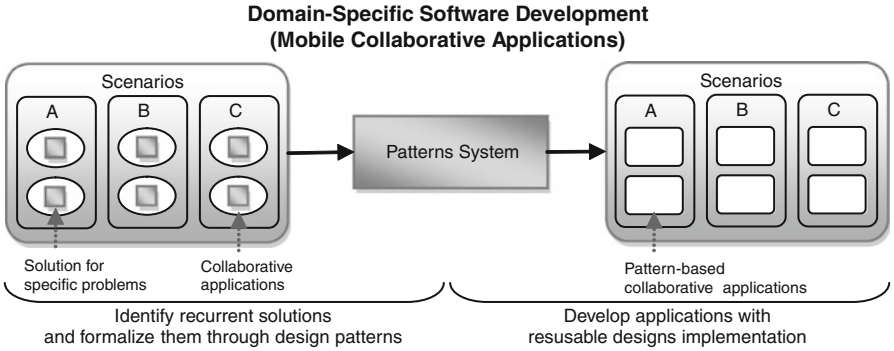

Fig. 1 Building mobile collaborative applications through a patterns

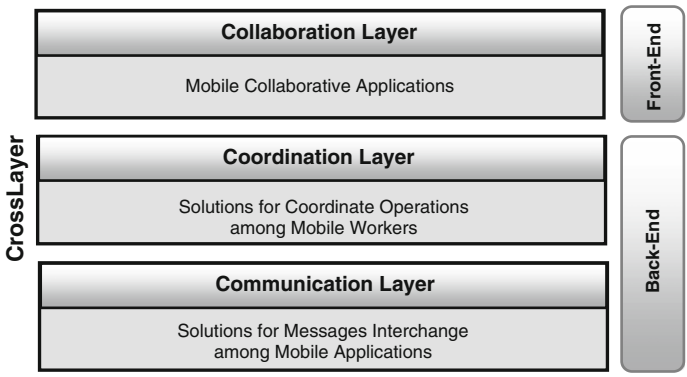

Fig. 2 Layered architecture to support mobile collaboration

services could be different in terms of concerns and granularity. The interaction between services related to different concerns is hierarchical: communication $\langle - \rangle$ coordination and coordination $\langle - \rangle$ collaboration. Interoperability among these services is required to support mobile collaboration, because frequently the service provider and the consumer run on two different computing devices.

Problem. Services provided by the collaborative system must be well structured. Otherwise, the system will be limited in terms of scalability, maintainability and adaptability. These challenges have been addressed by stationary collaborative systems through the use of centralized components such as a server. However, those solutions are inappropriate to support collaboration in mobile scenarios.

Solution. The architecture of a mobile collaborative application should be fully replicated to cope with the mobile users' autonomy. Thus, it is possible to see the collaboration scenario as a dynamic mesh without centralized components. This architecture must be layered because there is a clear hierarchy among the groupware services belonging to different concerns such as communication, coordination and collaboration (Fig. 2). The advantages of the layered architecture have already been recognized by the software engineering community (Avgeriou and Zdun 2005; Clements et al. 2003).

Services and public data structures of each layer should be accessible through an API (Application Programming Interface) in order to keep the services independence and the access control. The interaction protocol between services is part of each layer, and it can be dynamically selected based on contextual information. For example, mobile devices with little hardware resources require lightweight mechanisms for data sharing or peers discovery. If an application running on a laptop must interact with a service running on a PDA, then contextual information about the PDA's hardware resources (stored in a particular layer) will be used to dynamically adapt the interaction protocol between them. This strategy allows designers to separate the application's concerns and increase the system scalability, maintainability and adaptability. It also eases the implementation of a solution to deal with data and services interoperability.

Related Mobile Collaboration Requirements. Autonomy, interoperability and use of hardware resources.

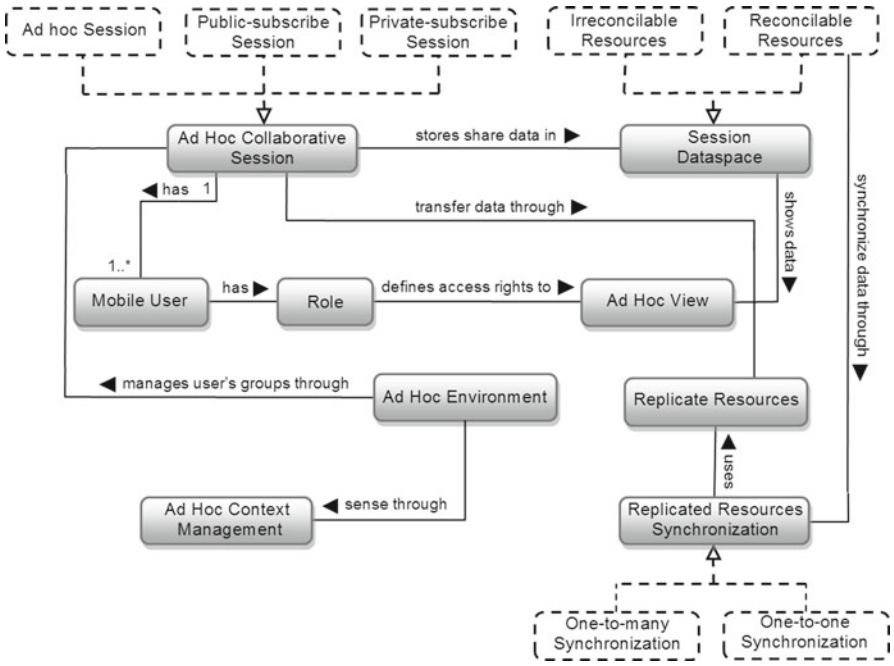

Fig. 3 Coordination patterns system for mobile collaborative applications

4.2 Coordination Patterns

The coordination patterns refer to the provision of services required by mobile workers' applications to coordinate the operations on the shared resources (e.g. files, sessions and services). This coordination is made individually (per mobile unit) and it generates a consistent view of the group activities. Figure 3 shows the proposed patterns system. Then each particular pattern is described in the next sub-sections.

4.2.1 Ad Hoc Environment

Context. Mobile collaborative applications usually require managing several work sessions. Each session groups users and shared data and services. Mobile users need to know which sessions are currently available in order to try to access those relevant ones for them, or otherwise to create a new one to collaborate with teammates.

Problem. It is not possible to use centralized components (such as the list of the currently available work sessions) in mobile collaboration scenarios due to autonomy reasons. It means the list of work sessions, with their participants, must be kept in a distributed way. Therefore the ad hoc environment not only has to manage the list of available work sessions, but it also has to do it in a distributed way and by ensuring the sessions information integrity.

Solution. This pattern proposes an ad-hoc environment to deal with this problem. It contains a fully distributed list of work sessions available to each mobile user. Each

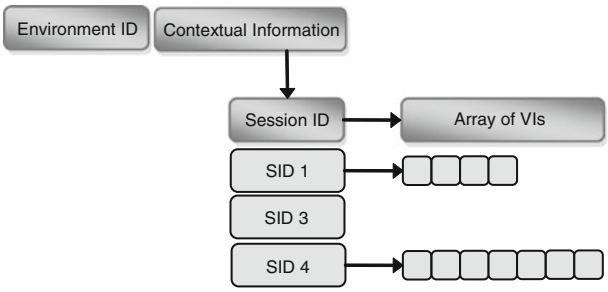

Fig. 4 General strategy for session management

mobile unit (and mobile user) has an instance of this list and a set of services to reconcile the local list with the list of teammates.

The local ad hoc environment keeps the following information: environment ID, description, creator and the list of available work sessions (Fig. 4). The descriptor also has the list of sessions where the local user is member. For those cases, each session contains the list of users' virtual identities (VI). This virtual identity is a unique ID identifying the couple user-device. This environment structure allows:

- A mobile application to support several work sessions composed of various mobile users.
- A session member's work does not interfere with the work of other session's members, even if they are working on shared objects.

The integrity of the information stored in each ad hoc environment can be kept using a reconciliation service, such as the one proposed by Messeguer et al. (2008). This information can be used by the mobile collaborative application to implement context-awareness mechanisms for data sharing and users' reachability. The contextual information related to the environment may contain data about the hardware resources of the local device. It allows managing the use of hardware resources depending on their availability. This type of environment can provide general services to support shared workspaces. Examples of the services are files transfer, message delivery, peers detection and users/sessions awareness.

Related Mobile Collaboration Requirements. Autonomy, context-awareness, awareness of users' reachability and use of hardware resources.

4.2.2 Ad Hoc Collaborative Session

Context. Mobile users trigger on-demand collaboration instances based on several goals, e.g. common work or similar interests. Typically they share data, knowledge or services as part of this collaboration process. The interaction among mobile users must be protected in order to avoid unauthorized access to resources shared among them.

Problem. Mobile collaborative applications need to implement a collaboration space in which mobile workers can interact freely without interrupts or unauthorized access to their shared resources. Similar to the ad-hoc environment, the collaborative session

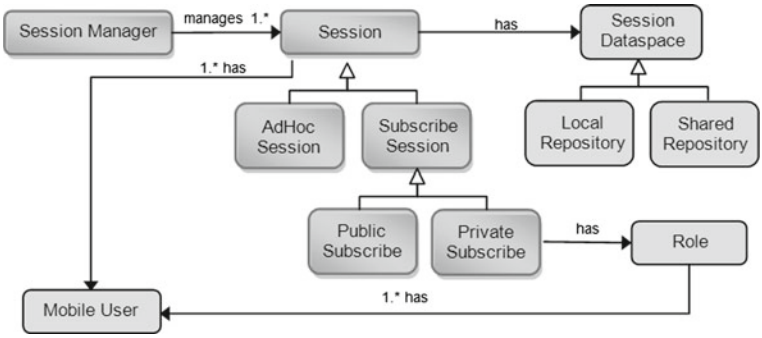

Fig. 5 General structure of the solution for ad hoc collaborative session

has to manage list of users and shared resources in a distributed way and keeping the integrity of the information shared by the session members. Each session has also to implement access control to shared resources based on each mobile user's role. Given the users' mobility, work sessions should be dynamically splitable or unifiable depending on the availability of a communication link among session members.

Solution. The solution to this problem is to use an ad-hoc collaborative session. The management of these sessions is done in a fully distributed way; therefore each mobile unit has to do it locally and keeping synchronized with the rest of the session members. Similar to traditional collaborative sessions (Guerrero and Fuller 2001), ad hoc collaborative sessions have a list of supported roles (rights to access the shared resources), members (users with roles), a session dataspace with private and public resources, and a session type considering the access control for users (ad-hoc, public or private session). The following figure illustrates the structure of the solution.

A work session is created when the first user is registered as member of it and it is deleted when the last user is unregistered. A session is potentially alive even if no users are currently connected, but having users registered in it. The work session types matching mobile collaboration are the following ones: ad-hoc, public-subscribe and private-subscribe (Fig. 5). The ad-hoc session is an open public resource that can be accessed by any user connected to the wireless network. The public-subscribe session involves a simple subscription process. Typically, users request a session subscription and automatically obtain the right to access it. Finally, private-subscribe sessions require a subscription process triggered by an invitation. Each invitation has associated a user role. If the mobile worker accepts the invitation, then s/he will play such role in that session. The strategy for session management must allow mobile users participate in more than one session and every session must have a local private and a shared repository.

Let us consider the situation depicted in Fig. 6 where users 1 and 2 are subscribed to session A. Once these users connect to the session, their public resources related to session A (those in their local shared repository) become available for any logged user. Because this process is automatic, it represents a low cost mechanism for data sharing. While the users are connected to a session they can replicate, into their local shared space, the remote shared resources that could be useful for him in the future

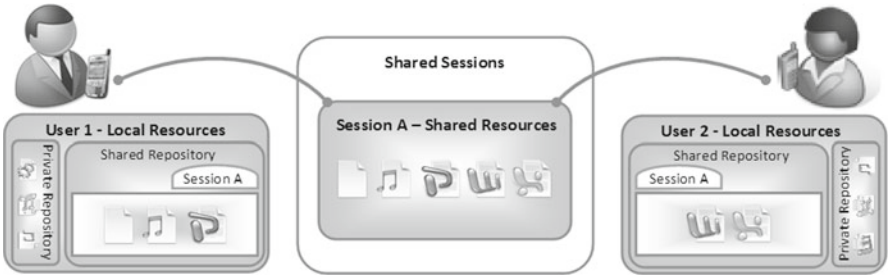

Fig. 6 Ad hoc collaborative session management

(i.e. for autonomy reasons). Of course, the users are also able to work on those shared resources. When a user leaves a session, the local private and shared resources are kept available for him/herself, by allowing the user work asynchronously.

The consistency of the shared data in an ad hoc collaborative session can be kept through two mechanisms: replication (i.e. file transfer) and reconciliation (i.e. data synchronization). Both of them represent low cost coordination mechanisms.

Typically, not all users have the same rights to access shared resources. The rights are related to the user's role for each session s/he is working on and indicates the user capability to carry out certain operations or processes on the shared resources. Mobile users usually have many work sessions with certain assigned role. Therefore, they need a mobile environment organizing and eventually coordinating multiple working sessions or user groups playing several roles. Sessions, users and roles management should be fully-distributed since the mobile environment should be autonomous.

Related Mobile Collaboration Requirements. (Data) Autonomy, awareness of users' reachability, low coordination effort.

4.2.3 Session Dataspace

Context. Team members involved in mobile collaboration produce information as a result of individual and collaborative work. These persons are frequently disconnected and perform their activities autonomously and work in parallel; therefore they need instances to share and synchronize their information.

Problem. Since mobile workers have to be autonomous, the resources required by them during an activity should be reachable all the time and they must be managed in a distributed way. It means some shared information will be replicated in the mobile units used by the work session members. This partial replication adds inconsistency to shared resources (because of the asynchronous updates), which has to be managed by session dataspace. Besides, this dataspace must be context-aware because it has to modify its content depending on the current composition of the work session.

Solution. A solution to this problem involves the use of a session dataspace in each mobile unit. This is a fully distributed component which provides a private and a public data repository for each mobile user and session. The information stored in the public repository can be accessed by any other session member; however the information in the private space is accessible just to the local user. The integration of all public

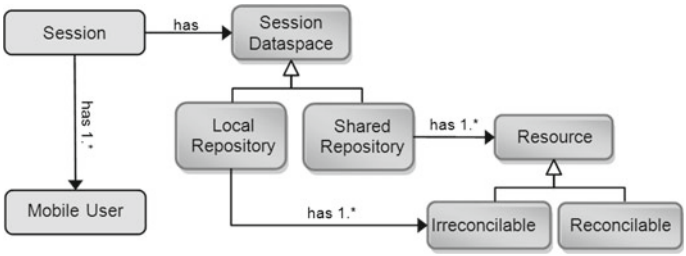

Fig. 7 General structure of the solution for shared repository

spaces belonging to the session members represents the session shared repository. This component can be used to provide awareness of the shared information to all users in a work session.

The shared repository contains two types of information resources: irreconcilable and reconcilable (Fig. 7). Irreconcilable resources are those pieces of information on whose internal structure the system has no information. The consistency among these resources can be kept just through replication (i.e. file transfer). On the other hand, a reconcilable resource is a piece of information with a well-known internal structure; therefore it can be synchronized with other instances of that resource (from other mobile users) in order to obtain a consistent representation of it. Both data sharing mechanisms mean a low coordination effort from the users. The following figure illustrates the structure of the solution.

Figure 8 presents a possible interaction scenario explaining the way the ad-hoc shared data repository works. All users have a private and a shared repository. Users

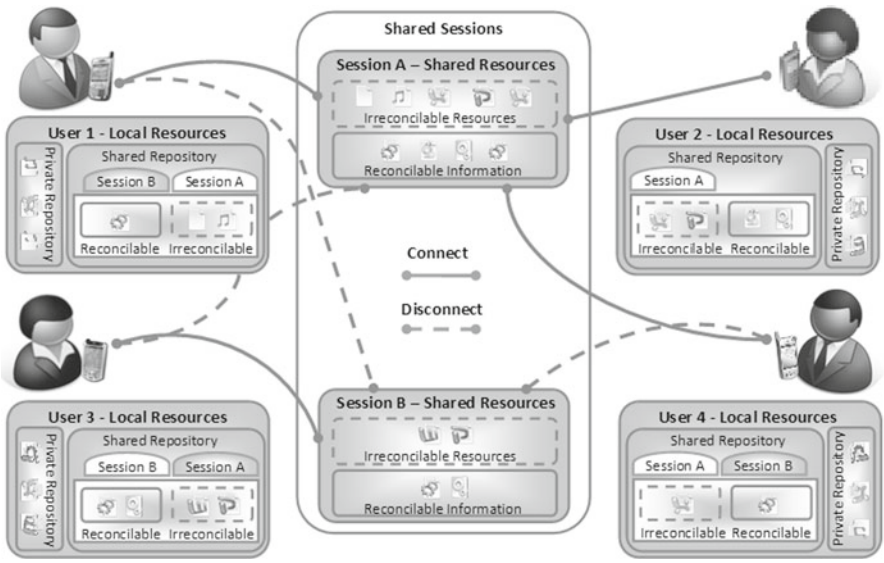

Fig. 8 Management of the ad hoc shared repository

1, 3 and 4 are subscribed to sessions A and B, whereas user 2 is subscribed just to session A. Let us assume users 1, 2 and 4 are logged to session A (indicated with white background in Fig. 6), and user 3 is logged to session B. The shared space of each session shows the set of reconcilable and irreconcilable resources which are available for the members.

Session A has three different versions of the same shared object. The reconciliation of such object instances can be done automatically or on-demand. The software designers have to determine which strategy fits with the type of activity the users are carrying out. In case of irreconcilable resources, the user is in charge of deciding (following some personal or organizational criteria) which is the last version of a replica. The user knows the file metadata information in order to make that decision, e.g. creation date, last update, owner, and version number.

In Fig. 8, session A does not show the shared resources of user 3, because that user is subscribed but not logged into that session. Something similar occurs with the shared resources kept by users 1 and 4, which are linked to session B. When a logged user leaves a session, all his/her shared resources are no longer available for the teammates, unless one of the connected users has local replicas of those resources.

The use of XML is recommended to specify shared resources. This description increases the data interoperability (because it is a standard) and eases the reconciliation process.

Related Mobile Collaboration Requirements. Autonomy, interoperability, context-awareness, low coordination effort.

4.2.4 Replicated Resources Synchronization

Context. Mobile users work autonomously most of the time and they carry out sporadic on-demand collaboration processes to keep updated and synchronized their local dataspace. Even if the collaboration process is tightly coupled, the users' mobility may cause disconnections and inconsistencies on the shared information.

Problem. Data consistency in fully distributed scenarios usually requires synchronization processes. These processes define how to synchronize the data replicas. When the synchronization process has to be done using a Mobile Ad hoc Network (with dynamic topology) including heterogeneous devices, the synchronization processes will be affected by several factors. Examples of such factors are: bandwidth between mobile devices, computing power of the involved devices, network topology and latency of changes. Moreover, this synchronization process must be done in a short time period, because frequently reconciliations are done as unattended (background) processes triggered while the user is on the move. Since the period of contact among mobile collaborators cannot be ensured, the reconciliation process should be as fast as possible.

Solution. The proposed reconciliation strategy is simple and it is based on the XMiddle reconciliation strategy (Mascolo et al. 2002). Such process minimizes the number of file transfers and the size of the transferred files, as a way to reduce the synchronization process duration and the hardware resources utilization. The algorithm transmits just the differences between data structures and, at the same time, is able to

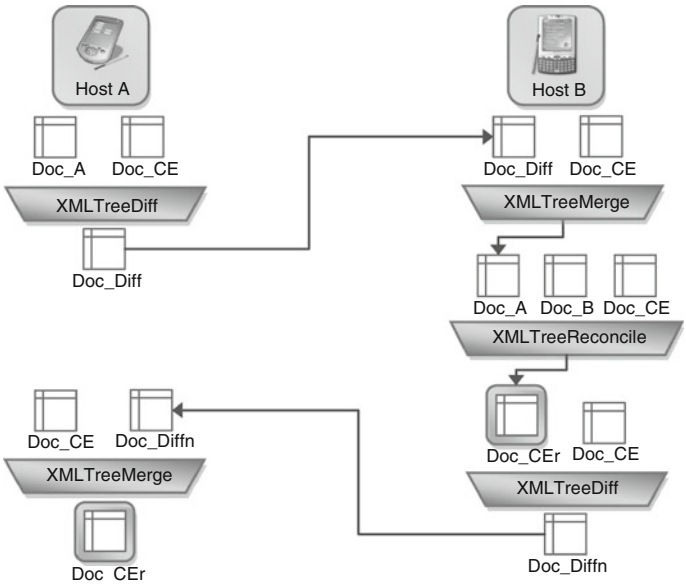

Fig. 9 The proposed reconciliation algorithm

reconstruct diverging replicas from a common previous edition on the same host in order to reconcile them locally. Then, the result of the reconciliation is propagated to the other hosts, communicating only the changes done on the common latest edition.

Figure 9 shows a synchronization example involving two replicas of the same document. The reconciliation process starts when Host A sends a reconciliation request to Host B. It receives the request and starts a local reconciliation using the information sent by A. We refer to the copy of the document stored on Host A as Doc_A and that maintained on Host B as Doc_B. Let us also assume that, after the execution of the first part of the protocol, the document Doc_CE has been chosen as Latest Common Edition, i.e., Doc_CE is the base document of the replicas to be synchronized. Host A computes the XMLTreeDiff operation (Mascolo et al. 2002) with Doc_CE and Doc_A as arguments (Doc_CE is the base document, whereas Doc_A is the modified replica). The output of this operation is the "diff" document Doc_diff, which will be sent to Host B. After receiving Doc_diff, B executes XMLTreeMerge with Doc_CE and Doc_diff as arguments in order to reconstruct Doc_A locally. Therefore, Host B now has a local copy of Doc_A and, naturally, Doc_B. Thus, the reconciliation between these two documents is performed on Host B without exchanging information with Host A. This process is carried out using the XMLTreeReconcile operation (Mascolo et al. 2002) with the following arguments: the local copy of the document Doc_B, the remote copy Doc_A and the latest common edition Doc_CE. The output is a "reconciled document" called Doc_CEn.

The final step is the generation of the reconciled document on Host B. This action is executing the XMLTreeDiff again with Doc_CE and Doc_CEn as arguments, in order to compute a new "diff" document. Afterwards, the document Doc_diffn is sent to Host

A, and XMLTreeMerge executes with Doc_{CE} and Doc_{diffn} as arguments. Now, Hosts A and B store the reconciled copies of the shared document, which will become the new latest common edition.

This algorithm can be used to support one-to-one and one-to-many synchronizations. In the second case, the reconciliation process should be divided in a set of ordered sequence of one-to-one synchronizations.

Related Mobile Collaboration Requirements. Autonomy, use of hardware resources and low coordination effort.

4.2.5 Replicate Resources

Context. Users produce data as a result of the mobile collaboration process. This data is stored in local files which mobile users usually share to support collaboration. The data sharing process will need to replicate the resources regardless of the data type: reconcilable (i.e. with a well known internal structure) or irreconcilable (i.e. in any other case).

Problem. Users' mobility causes high disconnection rates when replicating a file between mobile units. This disconnection rate requires robust mechanisms to replicate resources. Moreover, the replication process should be fast and simple enough to run on small computing devices.

Solution. The solution to this problem is to provide a file transfer mechanism allowing users interact in a work session, to replicate resources in a transparent way (Fig. 10). The file transfer process is based on the distribution of a set of small information pieces which can be sent in any order from the sender to the receiver. When a user decides to download certain remote resource, the component creates a download request (i.e. a *FileTransferTicket*). Then, the file transfer manager uses the contextual information (i.e. hardware features of the interacting mobile computing devices, and the distance between them) to determine the appropriate block size in which the resource will be broken down before being transmitted. The block size is relevant to be considered, because it directly influences the performance of the file transfer process. This information is stored in the *FileTransferWorkItem* element.

In case only part of the resource is needed on the remote user, partial file transfers are allowed in either block or striped mode. Increasing the file transfer performance is also possible with the use of multiple data channels for parallel transfer operations.

Related Mobile Collaboration Requirements. Autonomy, use of hardware resources and low coordination effort.

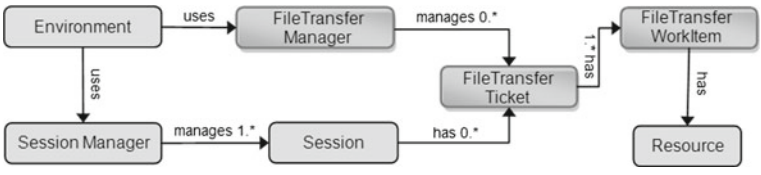

Fig. 10 General structure of the solution for replicate resources

4.2.6 Mobile User

Context. Users participating in a mobile collaborative process need to be uniquely identified regardless of the computing device they are using. Provided this is an on-demand process, mobile users need to know the identities of the potential collaborators who are currently available.

Problem. The user ID must be unique and it should identify a particular user regardless of the mobile device s/he is using. A similar identification mechanism is required for the potential collaborators (other mobile users in the same area). Moreover, the information about users' and neighbors' IDs should be managed in a fully distributed way, due to the aforementioned constraints.

Solution. The solution to this problem is to have a data structure, which we have called *mobile user*, containing the local user information required to support the mobile collaboration and to implement user presence. This structure is a reconcilable resource which is locally stored in each mobile device. This resource is shared among users in order to keep a common view from the users participating in a work session.

The mobile user data structure contains the mobile unit ID, the virtual (user) ID, the user's role, the user's visibility attribute and the list of neighbors (Fig. 11). The user's virtual identity (VI) is a unique ID which is linked to the IP address of the user's device. This VI is linked to the real identity (RI) which is the permanent user's ID. User sessions can be implemented as dynamic arrays of virtual identities (Fig. 12). On the other hand, the user visibility attribute allows implementing privacy policies, and awareness of user roles and user availability. The list of neighbors includes the set of potential collaborators available during a particular period.

Fig. 11 Mobile user data structure

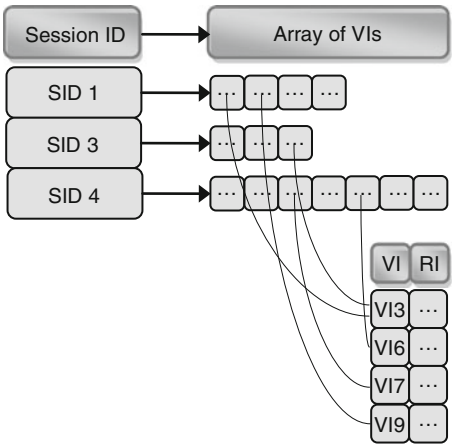

Fig. 12 Matching VIs and RIs

This neighbors list can be updated by two mechanisms: (1) peers discovery and (2) list synchronization. Peers discovery involves sending a message to a peer destination. If the destination is reached, a message is returned to the sender indicating the list of interim visited nodes. Such data is used to update the local list of reachable mobile units and neighbors. Then, a change-propagation mechanism can be triggered to the rest of the session members. In that case, the list update is done using a typical synchronization process.

Related Mobile Collaboration Requirements. Autonomy, awareness of users' reachability and low coordination effort.

4.2.7 Role

Context. Mobile collaborative applications usually require support for mobile users with different rights to access the shared information. Users having the same access rights should be treated in the same way by the collaborative system. Fully distributed access control management to shared resources is needed because mobile collaboration processes require autonomy.

Problem. Roles support needs to keep the semantics given by client-server collaborative systems, however the management must be fully distributed. Moreover, the user's role has to be kept consistent even if the user changes his/her mobile computing device.

Solution. The solution involves assigning a *role* to each mobile user for each of his/her sessions. Every private-subscribe session requires creating a role schema which has session roles defining access rights over the shared resources. Once these session roles have been defined, each user's role is linked to the mobile user through a mobile user role (Fig. 13).

Taking into account the reusability of this solution, it is possible to consider the role as a class maintaining information related to its name, the session to which it belongs and list of access rights to the shared resources (data and services). The role solution has to implement methods to store an instance, erase an instance, check if a role exists, check if a mobile user has enough rights to access a shared resource, and request a list of roles available in certain sessions.

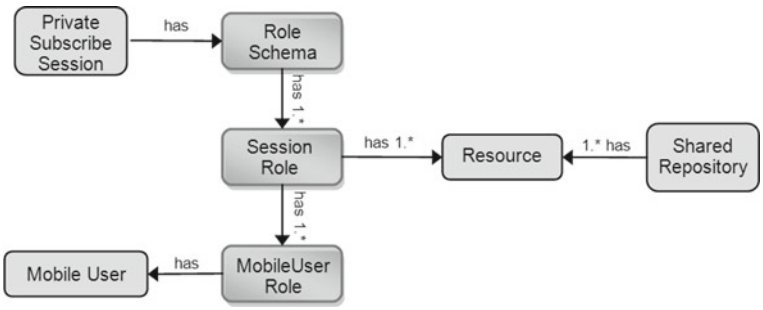

Fig. 13 General structure of the role pattern

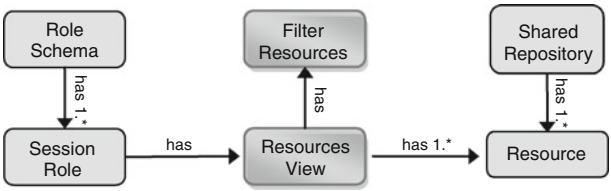

Fig. 14 General structure of the solution for ad hoc view

Related Mobile Collaboration Requirements. Context-awareness, awareness of users' reachability and low coordination effort.

4.2.8 Ad Hoc View

Context. The user's role sets the user's access rights on the shared resources (i.e. data and services); thus, users with the same role should have access to the same resource list.

Problem. Since the shared resources in an ad hoc session are distributed but no fully replicated, frequently users with the same role have access to different lists of shared resources. Mobile collaboration requires keeping the coherence of the access to these resources as much as possible, in order to avoid data islands (generating unnecessary parallel work) inside a work session.

Solution. The solution to this problem is to use an *ad hoc view* of the shared resources. This view contains a list of resources with their access grants, which are available for all users having a specific role. There is a view per role. Users with the same role should have access to the same list. These lists are reconcilable as a way to keep the coherence of each view. The only difference being allowed between the lists of two users having the same role is the resources availability. The following figure illustrates the structure of the solution (Fig. 14).

Although all shared resources are visible, some of them are reachable (if they are locally stored or they are replicable from a neighbor's dataspace) and other ones are unreachable (if neither the current mobile unit nor its neighbors have the resource). In order to increase the availability of the shared resources, a user can ask for a particular view which tries to replicate (in the local shared dataspace) the remote resources which are currently visible but unavailable for him/her.

The ad-hoc view can also be considered as a class interacting with the role class presented in the previous section. This class should provide methods to store and delete an instance, to check if a view exists, and to refresh and reconcile a view.

Related Mobile Collaboration Requirements. Context-awareness and low coordination effort.

4.2.9 Ad Hoc Context Management

Context. By context we mean the variables which can influence the behavior of mobile applications; it includes computing devices internal resources (e.g. memory, CPU speed or screen size) and external resources (e.g. bandwidth, quality of the network

Fig. 15 General structure of the solution for ad hoc context management

connection, and mobile hosts' location and proximity). Both types of variables are relevant to support coordination processes. However, the external variables are more dynamic in mobile scenarios than the internal ones; therefore, it is usually very challenging to sense, store and appropriately use the information they contain. Mobile applications need to be aware of the context in which they are being used to be able to adapt to heterogeneity of hosts and networks as well as variations in the user's environment. The management of this context information can help to optimize application behavior, compensating the resource scarcity.

Problem. Contextual information is changing all the time while doing mobile collaborative work. Mobile collaborative applications have to sense it, store it and appropriately use it to dynamically adapt its behavior. Therefore, such information has to be available all the time and it has to be as complete as possible. Usually there are computing devices participating in the collaboration process which are not able to sense some context variables; however, they are able to use this information if another device provides it to them. The challenge here is to determine how to combine the context sensing services embedded in the mobile devices, in order to provide a shared knowledge about the current work context of each user. Thus, devices with sensing capability will be also able to adapt the collaborative system behavior, depending on the changes in the local work context.

Solution. The solution to this problem involves the creation of an *ad hoc context manager*. This component has to be fully distributed and it must store, share, update and monitor current status of the context. The context status is represented through a shareable and reconcilable data structure (i.e. Mobile Node Context) (Fig. 15).

Mobile collaborative applications will adapt their functionality based on that information to cope with the changes in the work scenario (e.g., a mobile worker gets isolated or networking support is not available anymore). For instance, if the software designer wants to:

- Provide a service which is dependent on the place where the user is located, then the context manager needs to implement a model of each place as a full-fledged object, and assign a set of command objects with corresponding services to that object.
- Adapt the application behavior according to different time intervals; then the context manager must use condition/action rules to support the behavioral adaptations.
- Extend existing software to add context-aware behaviors; then the context manager must have a functionality which wraps the corresponding class with an object, which delegates the request to the component implementing the adaptation (e.g. a rule object or rule manager).

- Ease the interoperability among mobile units involving heterogeneous computing devices. This context manager can activate/deactivate groupware services on demand depending on the availability of hardware resources into the involved mobile units.

It must be noted the context manager has to be carefully engineered in order to reduce the use of limited resources, such as battery, CPU, memory or network bandwidth. A service-oriented approach can be useful to design and implement this component, because it deals with the heterogeneity of computing devices and resources shortage. Moreover, this approach involves a standard format for services, which helps increase the interoperability of the mobile collaborative solution.

Related Mobile Collaboration Requirements. Autonomy, interoperability, context-awareness and use of hardware resources.

4.3 Patterns vs. Mobile Collaboration Requirements

Figure 16 presents a correspondence matrix relating the proposed patterns and the requirements for mobile collaboration presented in Sect. 2. This matrix allows developers to select one or more design patterns in order to deal with a particular requirement.

It is important to highlight the proposed fully distributed architecture provides autonomy to mobile collaborative applications and it helps reduce the use of hardware resources by accessing local resources. Moreover, the separation of design concerns in several layers (i.e. cross layer pattern) provides flexibility and scalability to the solutions. Next section shows how these proposed patterns were used in particular mobile collaborative applications.

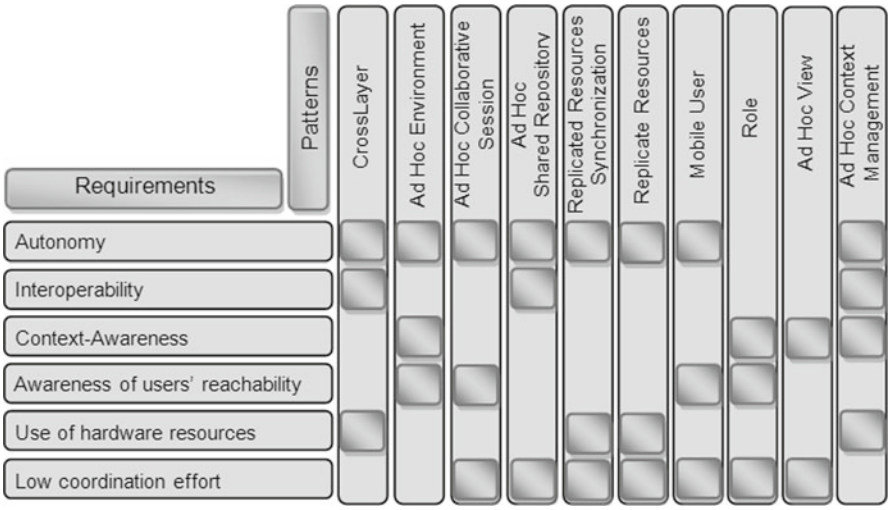

Fig. 16 Correspondence matrix

5 Patterns Evaluation Process

Several researchers highlight the difficulty of evaluating software patterns (Buschmann et al. 2007; Roberts and Johnson 1996; Schümmer and Lukosch 2007), and it seems to be a consensus that a proposed pattern needs to be used in many applications in order to become a valid pattern. That process typically involves many years; therefore any evaluation we can provide represents a preliminary one.

The evaluation strategy presented in this section adheres to the process proposed by Roberts and Johnson (1996). It involves developing and evaluating three simple applications, for different contexts, which use the proposed patterns. The results of such analysis will indicate if the proposal can be considered a potential pattern.

Next three sections present the mobile collaborative applications used in the current evaluation process. These sections also show how these patterns were embedded in the application design, and the obtained results. These applications were developed by graduate students as part of their MSc. theses. These students did not participate in the patterns definition process; they voluntarily used the patterns system as support for their applications design.

5.1 COIN (COnstruction INspector)

Contractors periodically deploy inspectors at a construction site to get an updated state of the work. Depending on the project, the number of inspectors working simultaneously, as part of a same team, can vary considerably. Figure 17 shows the main user interface of COIN (COnstruction INspector), a mobile collaborative application which supports the work of inspectors in construction projects (Ochoa et al. 2008).

The inspection process typically involves three phases: *registration, validation* and *reporting*. Inspectors review various parts of the physical infrastructure and record the project advances through annotations on digital blueprints they have available in their tablet PCs. The inspectors meet to synchronize annotations and resolve contradictory

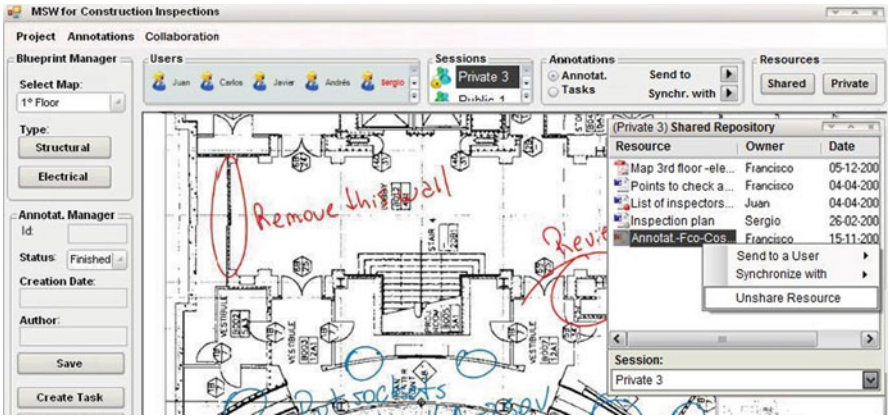

Fig. 17 COIN main user interface

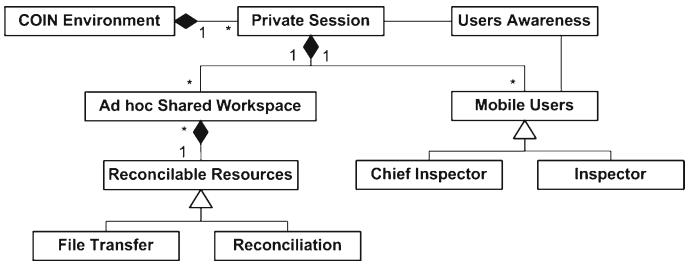

Fig. 18 COIN functionality to support collaboration

annotations after the reviewing process. Finally, the chief inspector reports the results to the contractor.

5.1.1 COIN Design

Figure 18 shows part of the COIN design, which implements the coordination mechanisms supporting the inspectors work. This application has a layered architecture that adheres to the cross-layer pattern. The *COIN Environment* (that adheres to the ad hoc environment pattern) provides a space to manage multiple work sessions (e.g. multiple inspections). Just private work sessions (i.e. a type of ad hoc session) are supported because the information used and recorded during an inspection is private and it cannot be shared with persons external to the team.

Each session has its own shared workspace (that adheres to the session dataspace pattern), list of mobile users (that adheres to mobile user pattern) and awareness mechanisms. Two users' roles were defined (i.e. inspector and chief inspector) and also two data replication mechanisms (i.e. file transfer and reconciliation through a synchronization process). All these components adhere to the patterns defined in the patterns system.

The engineer in charge of designing this application found the patterns are intuitive and easy to use. Although he did not have experience designing coordination services for collaborative systems, he feels the use of these patterns helped him to find a sound design option. After this experience, the designer thinks he is able to apply these patterns to various application scenarios.

5.1.2 COIN Design Evaluation

The coordination support embedded in COIN was evaluated and reported in Ochoa et al. (2008). The experimentation scenario considered two inspectors recording contingency issues in a simulated construction project. The obtained results showed the inspectors were able to perform the three steps involved in this process in a comfortable way.

Unfortunately, this experiment just compared the inspection process using paper-based blueprints with a process using COIN with digital blueprints. The obtained results show the *registration* activity was a little bit favorable to the COIN usage;

however in the *validation* and *reporting* stages the difference was several orders of magnitude faster when inspectors used the application. In addition, inspectors preferred to use the tool instead of the paper-based blueprints because of the simplicity to handle these resources. These persons found that COIN is appropriate to support this mobile collaborative activity. Although the results are still preliminary, they indicate the proposed patterns could be appropriate to support coordination in construction inspection scenarios.

5.2 MobileMap

Firefighters attending common emergencies (e.g. a fire or car accident) must make decisions when traveling to the emergency place and also during the emergency response process. Making such decisions requires knowing information about the emergency place, the contingency situation to address and the status of the response process. MobileMap, a mobile collaborative application, allows firefighters to share such information not only among them in the field, but also with the command center (Monares et al. 2009). This application is routinely used on laptops, PDAs and smartphones.

Typically each emergency has an incident commander in the field, who is in charge of designing, executing and monitoring the emergency response process. Furthermore, there are several other roles involved in the response, each one with particular responsibilities. For example, communication officers that provide communication support in the field, the rescuers in charge of looking for and rescuing possible victims, or response personnel who is in charge of the emergency mitigation process. All of them require making on-demand decisions in a distributed way based on the available shared information (Fig. 19a); however such decisions must be coordinated in order to keep control of the emergency response process.

The dynamics of the response process is unpredictable, because the decisions and actions are made depending on the evolution of the emergency situation and the possible damages to human life and civil infrastructure. Typically, coordinated improvisation is the common denominator in these mitigation activities.

Figure 19b–d show the shared information firemen get with MobileMap. Figure 19b presents a city map where it is possible to identify the emergency place (identified with a cross), the fire trucks location, and the location of interest points, such as hospitals or

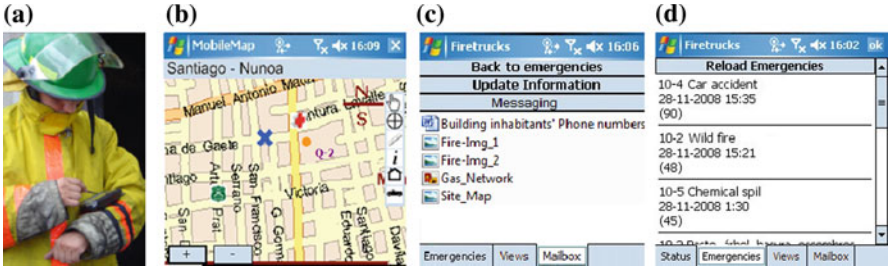

Fig. 19 MobileMap application

police departments near the emergency place. Figure 19c shows a set of files (e.g. pictures of the current emergency or maps of the affected area) that are shared among firemen in the field and also with firefighters that are going to the emergency place. Finally, Fig. 19d shows the list of the last emergencies and detailed information about them.

5.2.1 MobileMap Design

Figure 20 shows the architecture of MobileMap, which is layered and adheres to the cross layer pattern. The components in grey adhere to the proposed coordination patterns and components in white are just part of the application functionality, which does not involve coordination mechanisms.

This application uses just a shared repository and a list of users who access the public resources depending on their roles. The environment pattern is not implemented because the work involves just a public work session. MobileMap does not consider that a mobile user can be working in two parallel emergencies.

The mailbox component, which implements the session dataspace pattern, includes only the shared repository (specified in Fig. 20 as "shared file"). These resources are shared through on-demand file transfer operations (i.e. it uses the replicate resources pattern), because all this shared information is considered as an irreconcilable resource. Given the high mobility of firefighters, this consideration increases the robustness of the resource sharing process.

The component information manager is in charge of providing access to the shared resources, depending on the grants (i.e. role) of the user requesting such information. This component implements the ad hoc view pattern. Users manager implements the session pattern; therefore it keeps the record of mobile users (with roles) connected to the session. This is managed as a public-subscribe session. The public type of the session eases access by firemen who may belong to various fire companies.

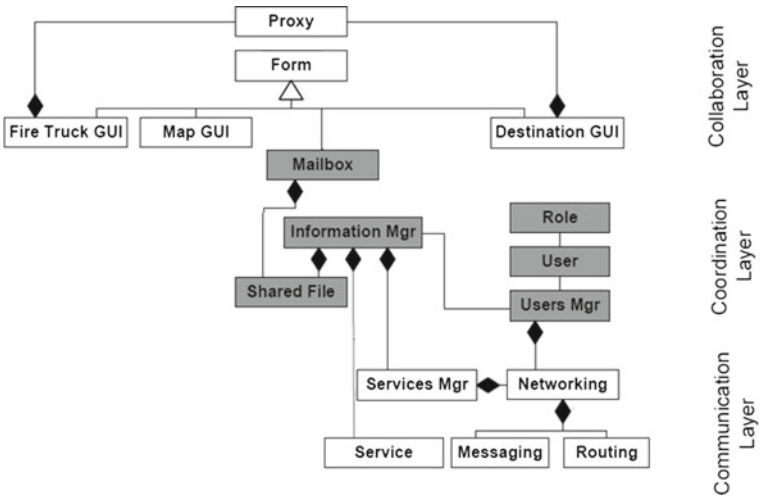

Fig. 20 MobileMap architecture

Finally, users and roles components adhere to the mobile user and role patterns respectively. The functionality implemented in these components is similar to the previous collaborative application (i.e., COIN).

The engineer in charge of designing MobileMap had experience developing mobile collaborative applications and also using design patterns for distributed systems. In the design of this application, the engineer used those design solutions he found more appropriate to support such mobile activity. It is interesting to see in Fig. 20 that the coordination components adhere to the proposed patterns. However, even more interesting is to observe that some of these components (e.g. information and users managers) implement variants to the proposed patterns. This means that (1) the designer was able to understand the whole meaning of the used patterns, and (2) these patterns can be adapted to deal with variants of the stated problem. This designer thinks the use of the proposed patterns helped him to reduce the design effort and also to find specific coordination solutions to embed in the application.

5.2.2 MobileMap Design Evaluation

This application, and therefore the design embedded in it, has been evaluated through two mechanisms (1) focus groups with firefighters who make decisions during emergencies, and (2) the empirical use of the application in real emergencies. Three focus groups have been done with firemen from several companies; each focus group involved 5–7 persons. Most of them act periodically as incident commanders.

The main functionality of MobileMap was explained in the focus groups. Thereafter they were able to use the application to make decisions on a hypothetical response process. All participants were able to enter and leave the public session and also share information. They felt comfortable using the tool and estimated that the information provided by MobileMap can help reduce up to 50% the use of radio channels during emergencies. Therefore it will contribute to deal with a limitation currently present in most fire companies around the world. Moreover, the availability of the supporting information should reduce the time required to make a decision and increase its quality.

The Nunoa command center and the 2nd Fire Company (both from Santiago, Chile) were the users of the tool. The tool has been used in five typical urban emergencies, in parallel with the focus groups. Partial results have been presented in Monares et al. (2009). These results are similar to those envisioned by firemen during focus groups: (1) they were able to use the application in the field, (2) the supporting information helps to make fast (and perhaps better) decisions, (3) the number of radio messages was reduced between 40-50% when compared with historical values. Besides, the experimentation process showed the application helps fire truck drivers to reduce mistakes concerning the selection of the route towards the emergency, and also to arrive faster to the emergency place. These preliminary results indicate the patterns embedded in MobileMap help to coordinate firemen during urban emergencies.

5.3 MOCET

MOCET (Mobile Collaborative Examining Technique) is a mobile educational collaborative application running on Tablet PCs. Its purpose is to help students carry

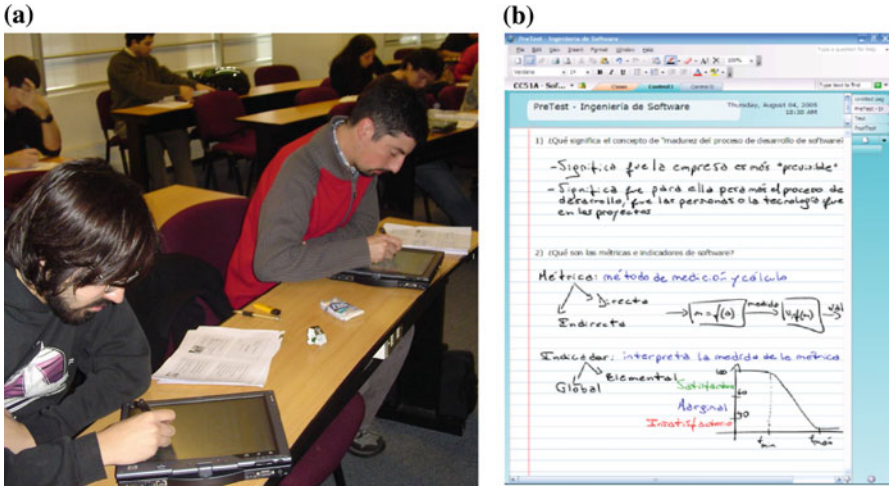

Fig. 21 MOCET application

out a particular examining process (Fig. 21a). The process has two stages: the *exam* and the *self-grading*. These stages are carried out in two consecutive sessions.

The dynamics of the exam is similar to a traditional one; however the process now involves the use of technology. Typically the students retrieve the exam statement from the instructor's computer, carry out the answering process and submit the answers (i.e. a digital document as shown in Fig. 19b) to the instructor through a file transfer operation.

The students collectively discuss each item of the exam in order to build the right answers in the next session. The instructor moderates the session. The students can have two types of interventions during the discussion: (1) to provide a proposed answer with the corresponding justification and (2) to provide a position (with justification) related to the answer proposed by another student. After reviewing each exam item, the students have to correct that item in their own exam. The correction assigns a score to the answer and justifies the assigned score. After such process, the instructor (or teaching assistant) reviews and grades the exam. Students who assigned a correct score and justification (i.e. the student's review is similar to the instructor's review), get extra points for the exam final score, since they understood which were the right answers to the exam item.

5.3.1 MOCET Design

Figure 22 shows part of the MOCET architectural design. The MOCET environment implements the ad hoc environment pattern. It was included in the tool because the instructor could have two or more different groups doing different tests in the same room. In that case, each group has its own session (i.e., examination) which adheres to the ad hoc collaborative session patterns. During the exam (first stage of the process), the application implements a private-subscribe session between each student and the

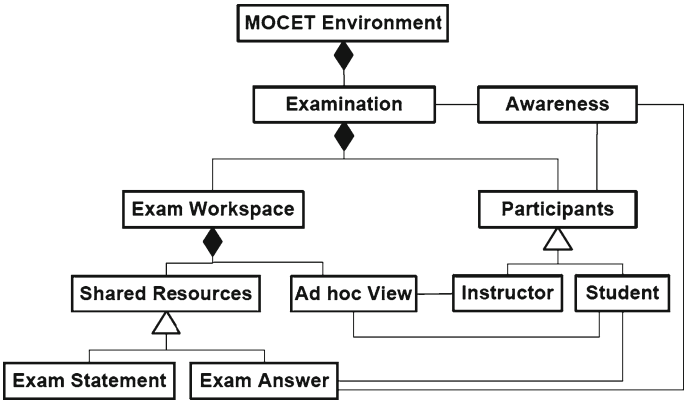

Fig. 22 MOCET architecture

instructor. Thus, the students are able to interact with the instructor (e.g. to retrieve and submit the exam), but do not with other students. During the self-grading process they also use private-subscribe session in order to avoid any attempt of illegal copy.

All sessions share a unique exam workspace that adheres to the session dataspace pattern; however the dataspace implemented in MOCET is cross-sessions. It means all users are able to access it depending on their roles, but these users are not able to see each other unless they belong to the same private session.

The shared resources component adheres to replicate resource pattern because the shared resources are irreconcilable. The participants' component implements the mobile user pattern. These participants could have one of two roles: instructor or student. The access to shared resources (i.e. exam statement and answer) depends on the users' role. The Ad hoc view component, which adheres to the pattern with the same name, is in charge of this access control process.

Similar to the COIN project, the engineer in charge of designing MOCET was not experienced in modeling mobile collaborative applications. However he was able to create an interesting design of the coordination services. This person indicates the patterns system helps him to avoid thinking a solution outside his expertise area. Provided the patterns were easy to understand, he just reused them. He thinks it helped reduce the effort, complexity and risks of the design activity. After using this application in a real scenario, the coordination patterns resulted to be also a good solution.

5.3.2 MOCET Design Evaluation

MOCET has been used to support exams in software engineering courses at the University of Chile (Ochoa et al. 2009). More than ten experiences have been performed in such scenario. Each experience consisted of students answering the exam using Tablet PCs and also students using paper and pencil for the same purpose. The obtained results show both MOCET was able to support the process and also the students preferred to answer using the tool instead of paper and pencil. It indicates the tool not only is easy to use, but also it embeds appropriate mechanisms to coordinate

the process performed by students and the instructor. Instructors participating in the process shared the students' view.

All coordination mechanisms encapsulated in the answering process worked appropriately. Probably, these results are also showing the proposed coordination patterns are appropriate to support nomadic work with micro-mobility.

6 Conclusions and Further Work

Mobile collaboration has brought the opportunity to support work activities in scenarios where workers have to be on the move to carry out a task. Several researchers have envisioned a positive impact on productivity and quality of work when users follow a mobile collaboration strategy (Andriessen and Vartiainen 2006; Hislop 2008; Schaffers et al. 2006). However, the features of these collaborative activities bring new challenges to collaborative system designers. Requirements, such as user autonomy, low coordination effort and high availability of shared resources, impose several constraints on the communication and coordination services required to support mobile collaboration. For example, no centralized components can be used because the users' mobility can make these resources inaccessible.

This paper presented a patterns system to support the design of coordination services required by mobile collaborative applications. These patterns have been used to deal with the stated requirements in several mobile collaborative systems. Particularly, Sect. 5 showed how these patterns were used to provide coordination services for three applications: COIN, MobileMap and MOCET. These applications were developed by graduate computer science students as part of their MSc. theses. All of them were able to use the patterns to support the design of these applications. It indicates these abstract designs were specified in a way that can be reused by other people. Typically, this type of reuse helps reduce the design risks, cost and time.

The experimental use of the applications embedding the proposed coordination mechanisms is showing these patterns are at least suitable to support coordination in such work scenarios. These patterns also serve as educational and communicative media for developers, students or researchers on how to design coordination mechanisms for mobile collaborative applications. They also foster the reuse of proven solutions.

These patterns will be extended to consider additional variants of them. One extension strategy considers the inclusion of new mechanisms to support the coordination process in mobile work scenarios. The second strategy involves the patterns extension to provide lightweight coordination mechanisms which can be used by mobile devices with low computing power, e.g., cellular phones.

Acknowledgments This work was partially supported by Fondecyt (Chile), grants No.: 11090224, 11060467 and 1080352, VRAID-PUC No. 19/2009 and by LACCIR grant No. R0308LAC004.

References

Alarcon R, Guerrero LA, Ochoa SF, Pino JA (2006) Analysis and design of mobile collaborative applications using contextual elements. Comput Inf 25(6):469–496

Andriessen JHE, Vartiainen M (2006) Mobile virtual work: a new paradigm? Springer, Berlin

Arvola M (2006) Interaction design patterns for computers in sociable use. Int J Comput Appl Technol 25(2/3):28–139

Avgeriou P, Zdun U (2005) Architectural patterns revisited—a pattern language. In: 10th European Conference on Pattern Languages of Programs, 1–39. UKV Konstanz, Germany

Avgeriou P, Tandler P (2006) Architectural patterns for collaborative applications. Int J Comput Appl Technol 25(2/3):86–101

Buschmann F, Henney K, Schmidt DC (2007) Pattern-oriented software architecture. A pattern language for distributed computing, vol. 4. Wiley, London

BNet (2008) IDC predicts the number of worldwide mobile workers to reach 1 billion by 2011. URL: http://findarticles.com/p/articles/mi_m0EIN/is_2008_Jan_15/ai_n24230213. January

Castro LA, Favela J (2008) Reducing the uncertainty on location estimation of mobile users to support hospital work. IEEE Trans Syst Man Cybern C Appl Rev 38(6):861–866

Churchill EF, Wakeford N (2001) Framing mobile collaboration and mobile technologies. In: Brown B, Green N, Harper R (eds) Wireless world: social and interactional implications of wireless technology. Springer, New York, pp 154–179

Clements P, Bachmann F, Bass L, Garlan D, Ivers J, Little R, Nord R, Stafford J (2003) Documenting software architectures: views and beyond. Addison-Wesley, Reading

de Vreede GJ, Briggs RO (2001) ThinkLets: five examples of creating patterns of group interaction. In: Ackermann F, Vreede GJD (eds) Group decision and negotiation. La Rochelle, France, pp 199–208

Dutta S, Mia I (eds) (2009) The global information technology report 2008–2009: mobility in a networked world. World Economic Forum & INSEAD

Ellis CA, Gibbs S, Rein GL (1991) Groupware: some issues and experiences. Commun ACM 43(1):38–58

Essmann B, Hampel T (2005) A design pattern for mobile-distributed knowledge spaces. In: Proceedings of the 2005 symposia on metainformatics, Esbjerg, Denmark

Farshchian B (2003) Presence technologies for informal collaboration. In: Riva G, Davide F, IJsselsteijn WA (eds) Being there: concepts, effects and measurement of user presence in synthetic environments. IOS Press, Amsterdam

Gamma E, Helm R, Johnson R, Vlissides J (1995) Design patterns: elements of reusable object-oriented software. Addison-Wesley/Longman Publishing, MA, USA

Guerrero LA, Fuller D (2001) A pattern system for the development of collaborative applications. J Inf Softw Technol 43(7):457–467

Herrmann T, Hoffmann M, Jahnke I, Kiele A, Kunau G, Loser K, Menold N (2003) Concepts for usable patterns of groupware applications. In: International ACM SIGGROUP conference on supporting group work. ACM Press, Florida, USA, pp 349–358

Herskovic V, Mejia D, Favela J, Moran A, Ochoa SF, Pino JA (2009) Increasing opportunities for interaction in time-critical mobile collaborative settings. In: Carrico L, Baloian N, Fonseca B (eds) CRIWG 2009. LNCS, vol 5784, pp 41—48

Hislop D (2008) Mobility and technology in the workplace. Routledge, Oxon

Jørstad I, Dustdar S, Van Thanh D (2005) Service oriented architecture framework for collaborative services. In: Proceedings of the 14th IEEE international workshops on enabling technologies: infrastructure for collaborative enterprise. IEEE Press, New York, pp 121–125

Kristoffersen S, Ljungberg F (2000) Mobility: from stationary to mobile work. In: Braa K, Sorensen C, Dahlbom B (eds) Planet Internet. Studentlitteratur, Lund, pp 137–156

Licea G (2006) Supporting reusability in fixed and mobile groupware applications. Int J Comput Appl 28(2):99–111

Luff P, Heath C (1998) Mobility in collaboration. In: ACM conference on computer-supported cooperative work. ACM Press, New York, pp 305–314

Mascolo C, Capra L, Zachariadis S, Emmerich W (2002) XMIDDLE: a data-sharing middleware for mobile computing. J Personal Wireless Commun 21(1):77–103

Messeguer R, Ochoa SF, Pino JA, Navarro L, Neyem A (2008) Communication and coordination patterns to support mobile collaboration. In: 12th International conference on computer supported cooperative work in design. IEEE CS Press, New York, pp 565–570

Milrad M, Spikol D (2007) Anytime, anywhere learning supported by smartphones: experiences and results from the MUSIS project. Educ Technol Soc 10(4):62–70

Molina AI, Giraldo WJ, Jurado F, Redondo MA, Ortega M (2008) Model-based evolution of an E-learning environment based on desktop computer to mobile computing. In: Proceedings of the international conference on computational science and its applications. LNCS, vol 5073, pp 322–334

Monares A, Ochoa SF, Pino JA, Herskovic V, Neyem A (2009) MobileMap: a collaborative application to support emergency situations in urban areas. In: Proceedigns of the 13th international conference on computer supported cooperative work in design (CSCWD'09). IEEE Press, Los Alamitos, pp 565–570

Moran T (2000) Shared environments to support face-to-face collaboration. In: ACM CSCW 2000: workshop on shared environments to support face-to-face collaboration. Philadelphia, Pennsylvania, USA, December

Neyem A, Ochoa SF, Pino JA (2007) Designing mobile shared workspaces for loosely coupled workgroups. In: Haake JM, Ochoa SF, Cechich A (eds) CRIWG 2007. LNCS, vol 4715, pp 173—190

Neyem A, Ochoa SF, Pino JA (2008) Integrating service-oriented mobile units to support collaboration in ad-hoc scenarios. J Universal Comput Sci 14(1):88–122

Nunamaker JF, Reinig BA, Briggs RO (2009) Principles for effective virtual teamwork. Commun ACM 52(4):113–117

Ochoa SF, Pino JA, Bravo G, Dujovne N, Neyem A (2008) Mobile shared workspaces to support construction inspection activities. In: Zarate P, Belaud JP, Camilieri G, Ravat F (eds) Collaborative decision making: perspectives and challenges. IOS Press, Amsterdam pp 270–280

Ochoa SF, Collazos C Bravo G, Neyem A, Guerrero LA, Ormeño E (2009) A computational tool for supporting the evaluation as a mechanism to improve learning. In: 9th IFIP world conference on computers in education (WCCE 2009), paper 80, Brazil, July

Pinelle D, Gutwin C (2005) A groupware design framework for loosely coupled workgroups. In: 9th European conference on computer-supported cooperative work. Springer, Netherlands, pp 65–82

Pinelle D, Gutwin C (2006) Loose coupling and healthcare organizations: adoption issues for groupware deployments. Comput Support Cooper Work 15(5–6):537–572

Rettie RM (2005) Presence and embodiment in mobile phone communication. Psychol J 3(1):16–34

Roberts D, Johnson R (1996) Evolve frameworks into domain-specific languages. In: Proceedings of the 3th patterns languages of programming conference (PLoP), Illinois, USA

Schaffers H, Brodt T, Pallot M, Prinz W (2006) The future workplace—perspectives on mobile and collaborative working. Telematica Instituut, The Netherlands

Schümmer T, Lukosch S (2007) Patterns for computer-mediated interaction. Wiley, West Sussex

Tan D, Poupyrev I, Billinghurst M, Kato H, Regenbrecht H, Tetsuani N (2000) The best of two worlds: merging virtual and real for face-to-face collaboration. ACM CSCW 2000: workshop on shared environments to support face-to-face collaboration. Philadelphia, USA

Tarasewich P (2003) Designing mobile commerce applications. Commun ACM 46(12):57–60

Tentori M, Favela J (2008) Collaboration and coordination in hospital work through activity-aware computing. Int J Cooper Inf Syst 17(4):413–442

Wiberg M, Ljungberg F (2001) Exploring the vision of anytime, anywhere in the context of mobile work. In: Malhotra Y (ed) Knowledge management and virtual organizations. Idea Group Publishing, Hershey, pp 157–169

Zurita G, Antunes P, Baloian N, Carriço L, Baytelman F, de Sá M (2008) Using PDAs in meetings: patterns, architecture and components. J Univ Comput Sci 14(1):123–147