Designing a Service Platform for Sharing Internet Resources in MANETs

Gabriel Guerrero-Contreras, José Luis Garrido,
Carlos Rodríguez-Domínguez, Manuel Noguera, and Kawtar Benghazi

Software Engineering Department
University of Granada
Avenida del Hospicio S/N
18071 Granada, España
zahara@correo.ugr.es,
{benghazi,jgarrido,carlosrodriguez,mnoguera}@ugr.es

Abstract. Nowadays, there is great interest to develop future Internet applications supporting resource sharing in mobile networks. This usually entails maintaining the consistency of those shared resources, that is, between different replicas of the resources. Moreover, mobile networks are characterized by varying capacity, in part, caused by their mobility, which also derives in frequent networking disconnections and network partitions. Therefore, to ensure the consistency of replicated resources being shared in a mobile network is more complicated that in networks with infrastructure support, as it requires to process events associated with the use of the resources themselves as well as those related with the state of the network. In response, in this paper an event-driven platform consisting of two services (monitoring and synchronization) and an underlying middleware has been designed in order to address the consistency of shared Internet resources in mobile networks in a simple way. The synchronization service only needs to be specialized to adapt the resource, depending on its type, to the required use in a particular system. The proposal is illustrated through the example of a collaborative document editor.

Keywords: Service Oriented Architecture, Event-driven Architecture, MANET, Future Internet, Resource Consistency.

1 Introduction

Nowadays, there is great interest to develop future Internet applications supporting human collaboration in mobile networks. These networks are characterized by features such as *localized scalability* and *uneven conditioning* [1]. Localized scalability is defined as "*a good system design has to achieve scalability by severely reducing interactions between distant entities*" [1]. It avoids network congestion. Uneven conditioning is the difference between devices capacities or the difference between different environments (e.g. GPS is not available indoors) [1]. These features have a

C. Canal and M. Villari (Eds.): ESOCC 2013, CCIS 393, pp. 331–345, 2013.

direct relation with the quality properties associated to the network, like performance. More recently, in highly dynamic environments Mobile ad-hoc Networks (MANET) [2] are being imposed. One key aspect of these networks is that they self-configure their infrastructure, according to the nearby, available devices at a given time. That is, MANETs do not have a static communication scheme and, therefore, the management of shared data becomes more complicated. Furthermore, more and more services are offered in the "cloud" [3]. Since the term was coined by George Gilder [4] the cloud has expanded enormously, in particular software as a service model (e.g. Google or Amazon) [5].

In this networks type, replication is a useful design technique to achieve some objectives, such as localized scalability. Data replication is recommendable in order to obtain high-availability and good performance in a system. As the number of replicas increases, the probability of being able to reach at least a replica of a resource increases. However, to obtain these benefits it is necessary maintain the correctness of the data. In replicated data, one key aspect of correctness is mutual consistency, that is, that *"all copies of the same logical data item must agree on exactly one current value for the data item*" [6].

Maintaining the correctness of the shared data is not a simple task, particularly in environments where network disconnections are frequent. To overcome this issue, there are different solutions which can be classified in two dimensions: *pessimistic strategies*, which consist in preventing inconsistencies; and *optimistic strategies*, which do not prevent inconsistencies, but allow them to happen, and when they are detected, they are resolved.

Nowadays, some research initiatives proposed solutions based on a peer-to-peer (p2p) architecture [7, 8, 9]. Nevertheless, these initiatives are focused in resource location and utilization and don't pay attention to the consistent management of resources when accessed concurrently under frequent disconnections. In this paper an event-driven, SOA-based (*Service Oriented Architecture*) [10] platform consisting of two services (monitoring and synchronization), and an underlying middleware, is presented. This platform has been designed to facilitate the consistent management of shared resources in mobile systems. The developer will only need to specialize the synchronization service to be adapted to each resource, depending on its type and the required use in a particular system. The aim is to be able to integrate advanced applications and heterogeneous systems using standard protocols in the business field (business processes) and notably in the Web (web services) [11, 12].

The rest of this paper is organized as follows. Section 2 presents related works for the management of shared data in mobile networks; Section 3 describes the proposed service platform intended to support resource sharing in future Internet applications taking into consideration the use of mobile networks; Section 4 presents the real example of a collaborative document editor to illustrate the proposal; Section 5 discusses the proposal; and finally, conclusions and future work are summarized.

2 Related Work

MANETs networks have proved its usefulness in fields such as automobile (*Vehicular Ad-Hoc Network*) [13, 14, 15]; disaster management, when the communication network is not available [16, 17, 18]; and sensors networks [19]. However, MANETs networks present a series of challenges such as [20]: *knowledge representation*, is a key issue in MANETs networks due to heterogeneous nature of these networks; *knowledge discovery*, MANETs networks are highly dynamics, thus find resources available in other hosts effectively is a fundamental task; *caching*, the use of cache data allows offline operations, but it complicates the task of maintain the shared resources consistent; *replication*, is another strategy to optimize the data management process and it allows brings closer the data to user; *query processing,* this issue is important because the applications that run in MANETs networks are context aware ;and *security,* with regard this, security is an important and complicated task due to are the device themselves the network infrastructure, and it changes constantly with a high numbers of connections and disconnections. In response, in recent years some research initiatives proposed solutions to shared resources management in MANETs, like DRIVE [7], MoGATU [8] and CHaMeLeoN [9].

DRIVE (*Dissemination of Resource Information in Vehicular Environments*) is a software platform designed to deploy in vehicles, PDAs, sensors or any device with wireless capabilities. The platform uses a p2p approach to data sharing and it is based in a layered model. The model is made up of three layers (from bottom to top): data layer, support layer and utility layer. The data layer implements the data model; the support layer defines how queries are processed and how the data is disseminated; and finally, the utility layer contains the modules to access at the different resources. Some aspects of the platform can be highlighted, such as its economic model to guarantee the dissemination of the information to the largest possible audience. With this model a node of the network will transmit the information even if it is not interested in the particular data.

MoGATU proposes a model to share data in MANETs trough p2p approach. Like DRIVE, it proposes a layered model with three layers and two intermediate sublayers. MoGATU uses a main storage system out of MANET network, however, if a device lost connection or a network partition occurs, MoGATU uses caching and replication. Besides, it defines its own transaction model, called NC-Transaction. With this model, MoGATU increases the number of successful transactions, however, this model not guarantee the global consistency of the shared data.

The main purpose of CHaMeLeoN is exchange multimedia data in a MANET network, including in streaming. CHaMeLeoN's philosophy is adapting the application to user and the user's context. Moreover, CHaMeLeoN implements motion prediction algorithms, in order to anticipate possible disconnections. This allows to place resource's replicas within the network efficiently.

The exposed software platforms are complete platforms to address resource management in MANET networks. However, these platforms are focused in resource location and utilization and don't pay attention to the consistent management of resources.

The work presented in [21] is based on a SOA approach to manage the data consistency in heterogeneous systems. In that context, there are several applications with different local data models. The models are different but they refer to the same data. The main objective of the work is that the modifications in a local model automatically disseminate to the remainder local models. The proposed architecture is based on a synchronization service and a directory service. The directory service linking local models between them and the synchronization service solves possible inconsistencies. This solution presents two main limitations: first, it does not cover the possibility of disconnections or network partitions; and second, it is an ad-hoc solution.

With reference to collaborative systems in mobile environments, these are being greatly accepted in areas such as health [22, 23] and education [24, 25]. Environments such as hospitals and schools have the advantage of a static infrastructure that can facilitate the shared data management, e.g. servers and storage systems, and therefore facilitate collaborative applications develop.

3 Service Platform Design

This section presents an event-driven, SOA-based proposal in order to address the consistent management of shared Internet resources in mobile networks in a simpler way. On the one hand, the SOA approach is used as it provides some benefits such as reuse, interoperability, scalability and ease of maintenance. On the other hand, event-driven architecture provides benefits such as broadcast communications, asynchrony and timeliness. Besides, both of these approaches reduce the coupling between the different system's components and platforms. Furthermore, this proposal is intended to be a generic solution that can be adapted to any resource type, as discussed below.

Figure 1 shows a general architecture of the service platform. The main component is the synchronization service. This service maintains the consistency of shared resources. The shared resources can be modified by several users concurrently. The synchronization service is a high-level service based on monitoring service. The latter is a basic service which stores all kind of information about changes on shared data. Both services can be replicated in order to improve the availability, e.g. when the network becomes partitioned.

Moreover in order to provide a complete development platform, these services have been deployed on a communication middleware for ubiquitous systems: BlueRose [26, 27]. The middleware notifies events occurring in the system under a publish/subscribe paradigm. This feature is interesting in mobile systems, with a heterogeneous communication environment, due to it reduces coupling between interfaces. The presented services are exposed through WSDL interfaces and are accessed remotely using SOAP. However, due to event-driven approach, they also can generate or receive events. In the next subsections the platform components are described in more detail.

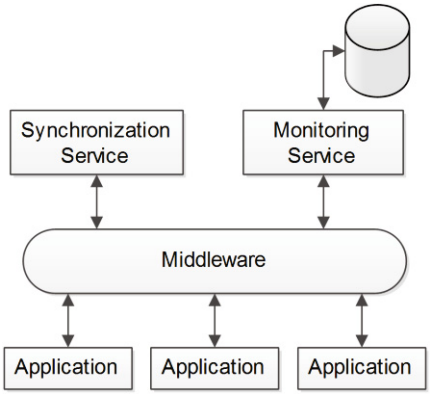

Fig. 1. General scheme of the platform to synchronizing shared resources in mobile networks.

3.1 Monitoring Service

Monitoring is a basic service which stores all kind of information about changes on shared data. This information can fulfill several purposes, e.g. version control, security control or system debug. In the synchronization service case, this information will be required when it will be needed to apply synchronization algorithms. In a device disconnection case, this information is fundamental to allow offline operations in the system.

Monitoring service supports different configurations in the system. It can be accessed from other system component to store some event, for example from the Synchronization service; or it can work as a subscriber too. With this last, the monitoring service can have different replicas which are subscribed to different events. This allows an efficient monitoring, due to the monitoring work can be divided between different replicas.

As has been mentioned, this solution expects to be a generic solution, not an ad-hoc solution. Therefore the monitoring service is designed to monitor any resource type. This is possible because in the platform all actions performed on shared resources are represented by events. The middleware BlueRose allows represent any information through resources and the monitoring service can monitor any event. Consequently the monitoring service is completely reusable and no need to modify its code.

Furthermore, the monitoring service uses a storage service as Figure 1 shows. This storage service can be replicated and/or distributed. Besides, the storage service can be in the cloud, with all the benefits that this entails. Also, if more control over data is desired monitoring service can uses a local storage service. Concerning the DBMS, a NoSQL system has been chosen in order to provide an efficient and flexible system to store events. Flexibility is needed owing to the structure of the events is unknown a priori.

3.2 Synchronization Service

The main service of the proposed platform is the synchronization service. This service is based on the monitoring service. Thus, with the information that monitoring service provides, the synchronization service can overcome the challenges of mobile networks (disconnections and network partitions) and it allows users working without connection.

In the case of a user disconnection (Figure 2), the user can continues working with cache data. While the connection is lost the generated events must be stored locally. When it recovers the connection, the first step is synchronizing the changes with the rest of the system through the synchronization service. For this, the synchronization service requires: local events of the device, disconnection interval and generated events in the system in this interval. Hence, the monitoring service is fundamental for the synchronization. Otherwise, if the synchronization service has not the generated events in the system in the interval disconnection, it cannot compare local events of the user and it cannot detect the possible conflicting changes with the remainder users. As mentioned above, cache is necessary to allow offline operations. Consequently, the client application must store the necessary data for user to continue working in disconnection case.

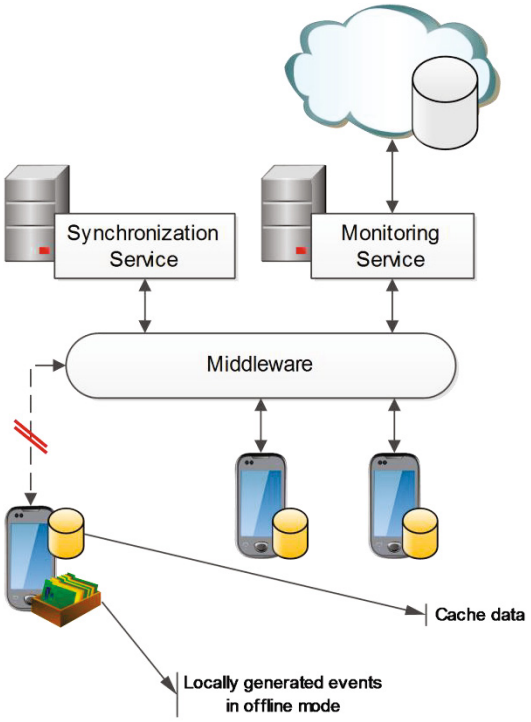

Fig. 2. A device working offline

The monitoring service is a general service, however, this is not possible for the synchronization service as the synchronization algorithms are dependent of the resource type. That is, synchronize images is not the same as synchronize text, because how to solve possible inconsistencies is distinct for each case. For this reason and with the goal of providing a reusable service, the synchronization service is a general service that must be specialized according to resource to synchronize. In this way, the common part related to manage the resource synchronization is identified and solved in the service, and the issues related with particular requirements of resources are addressed in service specialization. In this way, the proposal is intended to be applicable to any type of shared resource.

Another important issue is how the synchronization service can obtain the correct sequence of events. This is an important issue because the service is working in a distributed system where the clock values may differ. To resolve it, first is necessary adding a timestamp to event definition and use a time server to provide the same clock value to all devices of the network. Owing to different networks can interoperate (thought service replicas) Lamport's algorithm [28] to determine the order of events in a distributed computer system has been implemented.

It is especially necessary to take into account that offline operations are permitted and network partitions can occur. For these cases, when the time server is inaccessible, it will acts as follow. When the device lost connection and it starts to generate local events, it uses its clock value to generate a relative timestamp for local events. When this device recovers connection, before to connect with the synchronization service, it gets clock value from time server. With this value, it calculates the difference between its clock and the clock of time server and it recalculates the timestamp of local events. In this way, the correct sequence of events to the synchronization service is guaranteed and therefore it can apply synchronization algorithms correctly.

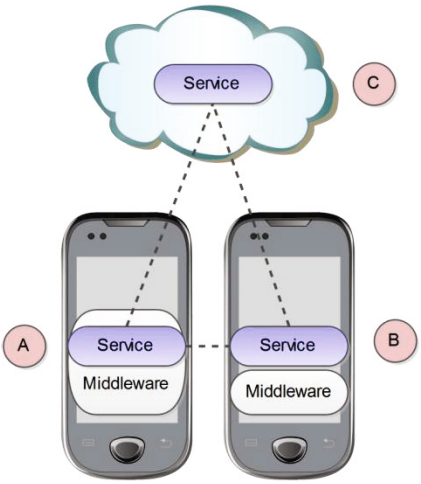

Fig. 3. Possible services deployments within platform architecture

3.3 Deployment

Regarding the deployment of services, they support different configurations, thus the platform can be adapted to requirement of a specific application. The possible configurations are:

- *Services within the communication middleware* (Figure 3, point A). They are disposed inside of middleware in order to obtain efficiency, regardless middleware overload is increased with each service added to it.
- *Services disposed over the middleware* (Figure 3, point B). This configuration matches when exists a high computational performance device in the network working as a server. Other devices benefits of decreased computational load offered by the dedicated server device.
- *Services on cloud infrastructure* [3] (Figure 3, point C). Cloud benefits are accessibility and scalability. The information or service are available wherever an Internet connection exists and resources could be increased or decreased as needed.

A possible system deployment is depicted in Figure 4. Figure 4 shows three MANET networks, these networks are not directly connected, however, they interoperate through services proposed above. To make this possible some replication protocol [29, 30] must be used to keep the service replicas updated and consistent. Replicating not only improves service availability, moreover, it helps to achieve localized scalability due to it reduces network traffic. This is possible owing to replicating allows devices use the nearest replica, as Figure 4 shows.

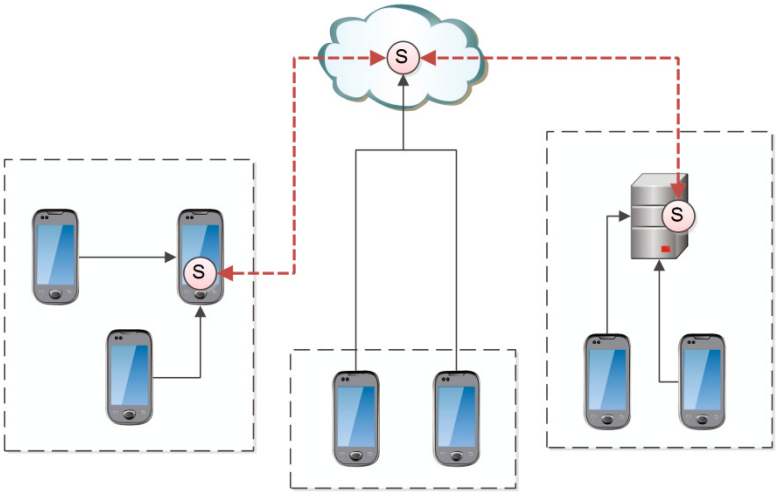

Fig. 4. A possible configuration of services in several MANET networks

4 Example: Collaborative Document Editor

As mentioned above, the synchronization service is a generic service that must be specialized with concrete synchronization algorithms of the shared resource to

manage. To test the validity of the service platform proposed, the synchronization service has been specialized into a document repository service. In this case, shared resources are text documents.

The system displays the following behavior (see Figure 5). The repository service keeps the full set of documents in the system. While users are online they are working with the documents of repository service. Users make changes to documents, these changes can be modified created or deleted a document. Moreover, these changes can be done concurrently and users who are working with same document can see these changes in real time.

The repository service, it is a specialization of the synchronization service, is responsible to integrate these changes consistently in documents set. Besides, the client application must have a cache copy of documents with which user is working, since it is not possible to foresee a disconnection. The number of files in cache is determined by several factors, such as disconnection frequency. To solve this, there are different approaches to caching in mobile environments [31].

Meanwhile, the monitoring service stores all generated events in the system. These events contain information about changes over documents and they are notified under a publish/subscribe paradigm.

When a client device disconnection occurs (User 1 in Figure 5), the client application starts working with cache documents copy. Besides, the client application starts storing generated events by user locally (Figure 5, step 1). These local events will be necessary to synchronize when connection is re-established. The client application can continue working offline with the unique limitation about the memory capacity of the device to store all the events that can be generated.

When connection is re-established, the client application first re-adjusts the events' timestamps generated offline (see section 3.2). Then it requests be synchronized with the system through the synchronization service (Figure 5, step 2). With this purpose, the application provides to synchronization service the disconnection interval and the generated events locally. This is the client application notifies to the synchronization service the changes that user has made on shared documents while the device was offline.

The repository service receives this request and it requests to monitoring service the modifications made by system users on shared documents during disconnection interval. The monitoring service queries the storage service for events set generated within the disconnection interval (Figure 5, step 3). When the monitoring service gets response, it sends the events set to the synchronization service (Figure 5, step 4). When the synchronization service has the generated events in the system and the generated events of the disconnected user, it can apply the synchronization algorithms and it can apply the resulting changes on documents repository (Figure 5, step 5). In this case (shared document), when the synchronization algorithm is appliqued the following cases can occur:

- *The document that has been modified by offline user has not been modified by another user in the system*. In this case, the changes can be applied directly.

- *The document that has been modified by offline user also has been modified by other user. But there are not conflicting changes.* In this case, the repository service automatically integrates both changes in the document. We have deemed that a conflictive change is when two or more users modify the same position of the document (the same *byte*). There are other options, such as consider conflictive modify the same word or phrase. Due to if two users modify the same phrase unknowingly the resulting phrase would be probably inconsistent.
- *The document that has been modified by offline user also has been modified by other user. There are conflicting changes.* In this situation, the repository service creates a parallel version of the document with the changes of the disconnected user. Thus, two or more versions of the documents are maintaining. The users themselves are responsible to resolve the conflict. The repository service only has the responsibility to ensure that do not miss a single user's change.

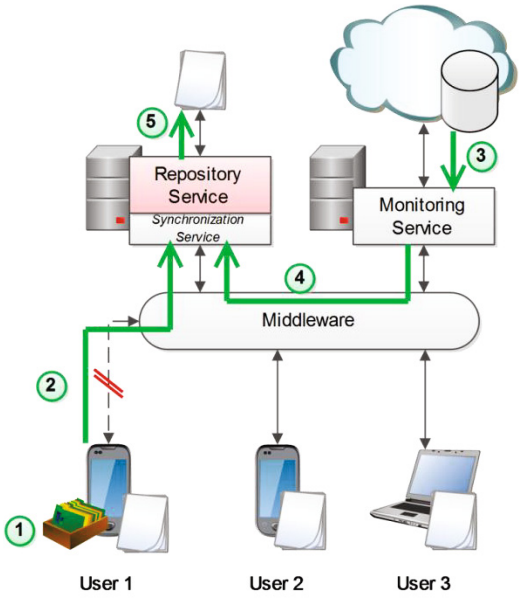

Fig. 5. Behavior of repository service when a reconnection occurs

This way, the consistency can be guaranteed, due to if there are conflicting modifications, these are saved in different versions of the document and none modification is lost. There are other strategies to maintain the consistency of shared documents when there are multiple writers [32]. For example, lock a document when there is a writer. This approach is simpler to implement, however, a tool that implements this approach really is not a collaborative tool, because users cannot write collaboratively in real time. Other derived approaches are interesting. For example, only some parts of a document can be locked. A weakness of this approach is that a disconnected user cannot lock the document, but even so it could uses this approach combined with others.

With regard to the repository service, we must note that the implemented synchronization algorithms are independent of client editor application. Consequently, the repository service can be used in any text editor conforming to the adopted event model. For last, underscore that it has not been necessary modify the monitoring service (see section 3.1), this service is sufficiently flexible to adapt to any shared resource. On the contrary, the synchronization service has been extended, through inheritance, to adapt to the shared resources of the case, text documents.

5 Discussion

With the objective of obtaining a higher acceptance in the use of advanced applications in MANET networks, the service platform presented in this paper intends to address the following requirements:

- Interoperability. As shows Figure 3, though the networks are not directly connected however, they can interoperate through services proposed due to service replicas.
- Adaptability. The designed services are not an ad-hoc solutions, the proposal is a generic solution that can be adapted to any shared resource type. On one hand, for the monitoring service this is possible due to event-driven approach (see section 3.1). On the other hand, for the synchronization service this is not possible. In this case, the synchronization service is designed as a generic service that can be extended through inheritance to adapt it to any shared resource type (see section 3.2).
- Platform-independent. Owing to use a SOA-based approach and to standards of publications, communications and access, the services implemented are platform-independent.
- Offline operations. As it has been explained above, this proposal is specially oriented to allow offline operations. This is particularly interesting in environments with problems maintaining connections, where disconnection of devices and network partitions are frequent. This feature is achieved owing to replication techniques, caching and the designed protocols to obtain ordered events.

However, some problems could be found such as network congestion, for instance, in collaborative document editing (see section 4). In this case, due to the synchronization is at low level (character/byte), each time that a character is introduced or deleted an event is generated and published. In a case with many users and many modifications, would be possible generate and transmit too many events. To solve this problem, would be possible synchronize at word level or study other possibilities such as partly block the documents when a writer is present in a particular document area. This will reduce the events generated in the system.

Moreover, use a SOA-based approach entails some issues to consider [33], particularly in large systems. Amongst other issues, it should be taken into consideration that when a service is integrated and working in a large system, before to modify it an impact study is necessary. This may limit system evolution, besides, as the system grows could be more complicated conforming to all standards of SOA approach.

6 Conclusions and Future Work

The service platform presented in this paper is a combination of a distributed computation techniques and event-driven and service-based architectures. The platform is composed of an underlying event-driven communication middleware which provides a publish/subscribe service to propagate the events. Over this middleware two services are disposed: the monitoring service, a basic service, which monitors and stores any event of the system and the synchronization service which process these events (provided them by monitoring service) in order to synchronize the replicas of shared resources. Besides, these services have been developed with the aim of being generic services. The monitoring service is reusable and it can be used with any resource type and no need to modify its code. However, the synchronization service needs to be adapted to the resource, depending on its type and the required use in a particular system. This is due to the synchronization algorithms are dependent of the resource type. Therefore, the common part related to manage the resource synchronization is identified and solved in the service, and the issues related with particular requirements of resources are addressed in service specialization.

The proposal presented especially addresses the case of systems that exhibit discontinuous operation. For this reason, the platform has been designed to make sure that synchronization service always gets the right events sequence. Even with the possibility of some devices are working offline or with network partitions. This is achieved, in particular, using a time server, adding timestamps to events and using Lamport's algorithm [28] to determine the order of events in the distributed computer system.

Besides, replication and caching techniques have been used in order to achieve localized scalability and high availability of the proposed services. Additionally, to get a better fulfillment of the applications requirements, the services have been designed for they can be deployed within the middleware, in order to obtain efficiency, regardless middleware overload is increased; on the middleware, when exists device in the network working as a server; or in a cloud infrastructure, in order to provide benefits such as accessibility and scalability. This way, flexibility of the platform is increased and it can provide greater adaptation of the system to the requirements of a particular application.

Moreover, for validity purposes of the proposal, the synchronization service has been specialized into a document repository service. The repository service implements application-independent algorithms to synchronize consistently text documents in collaborative environments. Consequently, the repository service can be used in any text editor conforming to the adopted event model. Besides, there are no restrictions about maximum disconnection time, while the user device has space to store local events, the user can continue working. With the use of the resulting repository service, users can edit document collaboratively in real time when they have connection with the system, and also they can edit the documents when they are offline. These documents will be synchronized when the connection recovers.

For last, the proposal platform seeks for benefits such as reusability, scalability, availability, interoperability, low coupling with platforms and its design facilitates offline operations.

Regarding future work, a depth study of the specific synchronization requirements of certain applications will be carried out, in order to provide synchronizations algorithms and assess adaptability and performance of the proposed platform. Additionally, a performance study of system is foreseen.

Moreover, the proposed platform will applied in ambits such as context information management; and semantic annotation tools for web, under which a high level of collaborative capabilities is desirable. Finally, the platform will integrate other high levels services, such as a location service [34], in a largest platform for context-aware and ubiquitous systems.

Acknowledgments. This research work has been funded by the Ministry of Economy and Competitiveness of the Spanish Government under the Research Project TIN2012-38600 and by the Vice-Rector's for Scientific Policy and Research of the University of Granada.

References

1. Satyanarayanan, M.: Pervasive computing: Vision and challenges. IEEE Personal Communications 8(4), 10–17 (2001)
2. Bansal, M., Rajput, R., Gupta, G.: Mobile ad hoc networking (MANET): Routing protocol performance issues and evaluation considerations (1999)
3. Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R., Konwinski, A., Zaharia, M.: A view of cloud computing. Communications of the ACM 53(4), 50–58 (2010)
4. Gilder, G.: The information factories. Wired 14(10) (2006)
5. Cusumano, M.: Cloud computing and SaaS as new computing platforms. Communications of the ACM 53(4), 27–29 (2010)
6. Davidson, S.B., Garcia-Molina, H., Skeen, D.: Consistency in a partitioned network: a survey. ACM Computing Surveys (CSUR) 17(3), 341–370 (1985)
7. Xu, B., Wolfson, O.: Data management in mobile peer-to-peer networks. In: Ng, W.S., Ooi, B.-C., Ouksel, A.M., Sartori, C. (eds.) DBISP2P 2004. LNCS, vol. 3367, pp. 1–15. Springer, Heidelberg (2005)
8. Perich, F., Joshi, A., Chirkova, R.: Data Management for Mobile Ad-Hoc Networks. In: Enabling Technologies for Wireless E-Business, pp. 132–176. Springer, Heidelberg (2006)
9. Ghandeharizadeh, S., Helmy, A., Krishnamachari, B., Bar, F., Richmond, T.: Data Management Techniques for Continuous Media in Ad-Hoc Networks of Wireless Devices. In: Furht, B. (ed.) Encyclopedia of Multimedia. Springer, Heidelberg (2008)
10. MacKenzie, C.M., Laskey, K., McCabe, F., Brown, P.F., Metz, R., Hamilton, B.A.: Reference model for service oriented architecture 1.0.OASIS Standard, 12 (2006)
11. Pasley, J.: How BPEL and SOA are changing Web services development. IEEE Internet Computing 9(3), 60–67 (2005)
12. Schroth, C., Janner, T.: Web 2.0 and SOA: Converging concepts enabling the internet of services. IT Professional 9(3), 36–41 (2007)

13. Zeadally, S., Hunt, R., Chen, Y.S., Irwin, A., Hassan, A.: Vehicular ad hoc networks (VANETs): status, results, and challenges. Telecommunication Systems 50(4), 217–241 (2012)
14. Studer, A., Shi, E., Bai, F., Perrig, A.: TACKing together efficient authentication, revocation, and privacy in VANETs. In: 6th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks, SECON 2009, pp. 1–9. IEEE (2009)
15. Godbole, V.: Intelligent Driver Mobility Model and Traffic Pattern Generation based Optimization of Reactive Protocols for Vehicular Ad-hoc Networks. International Journal of Information and Network Security (IJINS) 2(3), 207–215 (2013)
16. Reina, D.G., Toral, S.L., Barrero, F., Bessis, N., Asimakopoulou, E.: Evaluation of ad hoc networks in disaster scenarios. In: 2011 Third International Conference on Intelligent Networking and Collaborative Systems (INCoS), pp. 759–764. IEEE (November 2011)
17. Reina, D.G., MaríN, S.L., Bessis, N., Barrero, F., Asimakopoulou, E.: An evolutionary computation approach for optimizing connectivity in disaster response scenarios. Applied Soft Computing (2012)
18. Reina, D.G., Toral, S.L., Barrero, F., Bessis, N., Asimakopoulou, E.: Modelling and assessing ad hoc networks in disaster scenarios. Journal of Ambient Intelligence and Humanized Computing, 1–9 (2012)
19. Cordeiro, C.D.M., Agrawal, D.P.: Ad hoc and sensor networks: theory and applications. World Scientific (2011)
20. Islam, N., Shaikh, Z.: A survey of data management issues & frameworks for mobile ad hoc networks. In: 2011 International Conference on Information and Communication Technologies (ICICT), pp. 1–5. IEEE (July 2011)
21. Svensson, E., Vetter, C., Werner, T.: Data consistency in a heterogeneous IT landscape: a service oriented architecture approach. In: Proceedings of the Eighth IEEE International Enterprise Distributed Object Computing Conference, EDOC 2004, pp. 3–8. IEEE (September 2004)
22. Selanikio, J.D., Kemmer, T.M., Bovill, M., Geisler, K.: Mobile computing in the humanitarian assistance setting: an introduction and some first steps. Journal of Medical Systems 26(2), 113–125 (2002)
23. Rahbar, A.: An E-Ambulatory Healthcare System Using Mobile Network. In: 2010 Seventh International Conference on the Information Technology: New Generations (ITNG), pp. 1269–1273. IEEE (April 2010)
24. Sancristobal, E., Martin, S., Gil, R., Orduna, P., Tawfik, M., Pesquera, A., Castro, M.: State of art, Initiatives and New challenges for Virtual and Remote Labs. In: 2012 IEEE 12th International Conference on Advanced Learning Technologies (ICALT), pp. 714–715. IEEE (July 2012)
25. Kahiigi, E., Ekenberg, L., Hansson, M.: Exploring the e-Learning State of art. In: Conference on E-Learning, Academic Conferences Limited, pp. 349–368 (2007)
26. Rodríguez-Domínguez, C., Benghazi, K., Garrido, J.L., Valenzuela, A.: A platform supporting the development of applications in ubiquitous systems: the collaborative application example of mobile forensics. In: Proceedings of the 13th International Conference on Interacción Persona-Ordenador, p. 41. ACM (2012)
27. Rodríguez-Domínguez, C., Benghazi, K., Noguera, M., Garrido, J.L., Rodríguez, M.L., Ruiz-López, T.: A Communication Model to Integrate the Request-Response and the Publish-Subscribe Paradigms into Ubiquitous Systems. Sensors 12(6), 7648–7668 (2012)
28. Lamport, L.: Time, clocks, and the ordering of events in a distributed system. Communications of the ACM 21(7), 558–565 (1978)

29. Hara, T.: Effective replica allocation in ad hoc networks for improving data accessibility. In: Proceedings of the IEEE Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies, INFOCOM 2001, vol. 3, pp. 1568–1576. IEEE (2001)
30. Karumanchi, G., Muralidharan, S., Prakash, R.: Information dissemination in partitionable mobile ad hoc networks. In: Proceedings of the 18th IEEE Symposium on Reliable Distributed Systems, pp. 4–13. IEEE (1999)
31. Barbará, D., Imieliński, T.: Sleepers and workaholics: caching strategies in mobile environments. ACM Sigmod Record 23(2), 1–12 (1994)
32. Posner, I.R., Baecker, R.M.: How people write together [groupware]. In: Proceedings of the Twenty-Fifth Hawaii International Conference on System Sciences, vol. 4, pp. 127–138. IEEE (1992)
33. Lewis, G.A., Morris, E., Simanta, S., Wrage, L.: Common misconceptions about service-oriented architecture. In: Sixth International IEEE Conference on Commercial-off-the-Shelf (COTS)-Based Software Systems, ICCBSS 2007, pp. 123–130. IEEE (February 2007)
34. Ruiz-López, T., Rodríguez-Domínguez, C., Noguera, M., Garrido, J.L.: Towards a Reusable Design of a Positioning System for AAL Environments. In: Chessa, S., Knauth, S. (eds.) EvAAL 2011. CCIS, vol. 309, pp. 65–79. Springer, Heidelberg (2012)