A Middleware Service for Mobile Ad Hoc Data Sharing, Enhancing Data Availability

Malika Boulkenafed, Valérie Issarny

A Middleware Service for Mobile Ad Hoc Data Sharing, Enhancing Data Availability

Malika Boulkenafed and Valérie Issarny

Inria-Rocquencourt
Domaine de Voluceau, Rocquencourt, BP 105
78153 Le Chesnay Cédex, France
{Malika.Boulkenafed,Valerie.Issarny}@inria.fr
http://www-rocq.inria.fr/arles/

Abstract. It is now commonplace for a person to use lightweight wireless computing devices, and to make his/her data available to other people's devices using todays various networking capabilities (infrastructure-based WLAN, ad hoc WLAN, GSM, etc.). Middleware platforms initially developed for stationary distributed systems cannot be directly applied in such a mobile environment. They must adapt their functionalities so as to best cope with possible resource constraints (energy, storage) of mobile terminals as well as with the various types of wireless networks that are now available. In this paper, we present a middleware service that allows collaborative data sharing among ad hoc groups that are dynamically formed according to the connectivity achieved by the ad hoc WLAN. Our service enhances, in particular, data availability within mobile ad hoc collaborative groups, and integrates a new adaptive data replication protocol for mobile terminals, combining both optimistic and conservative schemes. Our service has been designed so as to minimize energy consumption and optimize data availability and storage consumption.

Keywords. Middleware services, data availability, collaborative work, ad hoc networks, mobile computing, wireless networks.

1 Introduction

Given the current advances in the field of mobile computing concerning both lightweight devices and wireless networking capabilities, middleware infrastructures have to adapt their services to better support this mobile wireless environment and the new classes of applications that result from it. Lightweight computing devices (e.g., PDA[1], third generation mobile phones, hand-held computers, etc.) are becoming increasingly available for a large range of users. In addition, low-power and short-range wireless connectivity among these devices (e.g., Bluetooth, IEEE 802.11, MANET[2], HiperLAN, OWS[3], and HomeRF in the

[1] Personal Digital Assistants.
[2] Mobile Ad hoc NETwork
[3] Optical Wireless Solution

M. Endler and D. Schmidt (Eds.): Middleware 2003, LNCS 2672, pp. 493–511, 2003.

context of PAN[4]) are increasingly flexible, especially with the advent of protocols supporting ad hoc communication (e.g., Bluetooth, IEEE 802.11, MANET) and enabling a collection of mobile hosts with wireless network interfaces to form a temporary network without the need for any established infrastructure or centralized administration [11], [13]. Middleware infrastructures initially developed for stationary distributed systems cannot be directly applied in such a mobile environment. Therefore, widely accepted middleware platforms have specified their *wireless* or *mobile* variants (e.g., wireless CORBA). Indeed, mobile devices are resource-scarce (low amount of persistent storage, limited battery power, etc.). It is therefore necessary to adapt all system functionalities and in particular functionalities related to data management (adequate coherency and replication management protocols), which is the focus of our paper. In this context, a number of services and/or protocols have been proposed regarding specific problems raised by mobility, such as disconnected operations (e.g., [9], [12], [15], [17], [20], [23], [25], [26], [27] and discovery of services [3].

Since it became commonplace to find users equipped with lightweight devices and short-range wireless interfaces, new needs for adapted functionalities that account with the various connectivities enabled by today's networking capabilities have emerged. There are scenarios in which no wired or wireless networking infrastructure is available, either because it may be economically impractical or physically impossible to provide the necessary infrastructure. Therefore, ad hoc networks are particularly suitable. Ad hoc networks represent the latest trend in distributed computing; work in this area has mainly dealt with routing protocols [1], [19], [24], [18] without taking into account data management within these networks. However, ad hoc networks are particularly well suited to form small groups of devices that are in the communication range of each other, in order to share and manipulate common data when the fixed networking infrastructure is not available. This requires devising adequate data management that allows the user to work in a collaborative manner and to benefit from the flexibility of ad hoc networks, thus supporting the notion of *mobile collaborative ad hoc groups*.

An example of mobile collaborative ad hoc group is a working meeting regarding some international cooperation (e.g., a European research project), which takes place in a hotel conference room, somewhere in a convenient destination easily reachable by all partners. An other example is a network gaming session, which may happen at any place (e.g., a cafeteria, train, lounge, etc.). Consider the first example. Representatives from each project partner meet periodically to work in a collaborative manner on the project. In order to facilitate travel arrangement, project partners choose a European meeting location that is easily reachable by all of them. Usually, they book a hotel, which unfortunately does not always provide any facility to support this kind of meetings, except may be a projector and an insufficient number of sockets. Attendees bring their project data on their wireless devices, which range from laptops to PDAs. It means that these devices have heterogeneous capacities in terms of persistent storage, and

[4] Personal Area Network.

battery. Furthermore, attendees do not bring all project data. Thus, data are already distributed between the attendees' devices according to both the attendees' contribution and the capacity of their devices. Every attendee must be able to easily access any project data brought by others, even if not locally cached on his device. This can be done either using floppy disk, or memory card, but it does not allow any coherency management of such replicated data. Fortunately, attendees' devices are equipped with wireless interface cards. Thus, they would like to be able to share their data using the wireless network even if there is no existing networking infrastructure in the meeting hotel, provided that some security guarantees are given regarding the wireless communication. The solution to this requirement is to use ad hoc networking (e.g. IEEE 802.11b WLAN in ad hoc mode) with adequate data management functionalities, which support secure mobile ad hoc collaborative groups. These functionalities subdivide into two core categories:

1. Secure group management that ensures: (i) transparent group creation using a discovery service to detect trusted partners (their devices) that *belongs* to the group; (ii) adaptation to the group dynamics, so that members can leave or join the group at any time; (iii) secure group communication through an encryption protocol that must account with devices' resources and network connectivity.
2. Data management that ensures: (i) that members access the most recent data version available within the ad hoc group, (ii) data coherency within the group through a conservative coherency protocol to effectively support collaborative work, since members must have the same version of shared data; (iii) adaptive data replication to avoid data loss if the devices storing them suddenly disconnect.

There is the need for a middleware service guaranteeing collaborative data sharing within mobile ad hoc groups and enhancing data availability while meeting the above constraints, by making adequate tradeoffs among data replication, data coherency, and resource consumption on mobile devices. Most of the replication mechanisms that have been proposed for mobile environments do not manage replication with respect to the mobile devices' resources or more generally with respect to the execution context. They all assume that mobile devices have enough cache to contain a user's entire working set of data. This is not a reasonable assumption when the system runs on an environment containing hand held devices with reduced storage capacities. In addition, these systems rely on infrastructure-based networks (e.g., reachability of a base station), and do not offer the possibility of forming ad hoc collaborative groups. These systems, further, maintain coherency of replicas optimistically, which means that they allow updates to be performed independently on any replicas. Although this is convenient for stand-alone mobile devices, it does not match the requirement of accessing a uniform version of the shared data within collaborative ad hoc groups.

This paper presents a middleware service for data sharing within a mobile ad hoc collaborative environment that addresses the aforementioned require-

ments. It supports secure group management and provides an adequate solution to enhance data availability with respect to mobility constraints (connectivity, device resources) through an adaptive availability scheme that deals with data coherency and replication, in a way that minimizes energy and storage consumption. Our adaptive replication protocol ensures data availability despite the dynamics of the collaborative groups according to the users' specific situation (e.g., profile of the device in use, network connectivity, etc.).

This paper is organized as follows: Sections 2 and 3 are the core of this paper. Section 2 presents functionalities of the middleware service necessary to support secure mobile ad hoc collaborative groups. Section 3 presents our contribution concerning data availability management within ad hoc collaborative groups. Our contribution lies in the integration of both conservative and optimistic coherency management within an adaptive data replication protocol for enhanced availability according to group dynamics and devices' specific situations. Section 4 assesses the proposed service in terms of local storage and communication overhead, response time, and energy consumption. It is shown that using our protocol, generated meta-data overhead is negligible, response time grows linearly with data update size, while group size affects it slightly. Finally, Section 5 concludes, summarizing our contribution and discussing our current and future work.

2 Middleware Service for Data Sharing within Collaborative Mobile Ad Hoc Groups

Ad hoc networks enable users equipped with lightweight computing devices and wireless interfaces to form a temporary network without the need for any established infrastructure. It is quite interesting to exploit such flexibility to allow them to share and manipulate some data in a collaborative manner (e.g., working meeting, network gaming, etc.) provided that some guarantees regarding security are given. Therefore, a mobile middleware platform should include a service that provides a number of functionalities so as to best cope with mobile collaborative ad hoc groups.

The secure group management functionalities (see §2.1) include discovery of peer mobile terminals that are in the communication range of each other. Then, to form a collaborative group, all the peer terminals should authenticate themselves. Authentication requires a trusted third party (e.g., the home server of the European project) to issue a digital certificate, which is stored on any device that previously authenticated with the given third party. Then, peers that can trust each other build a secure ad hoc group in order to share data and collaborate. Groups are restricted to one hop ad hoc networks, because we consider that the collaborating peers are usually located in the local communication range of each other. However, every peer is free to leave the group, as well as, new trusted peers can join it at any time.

Data sharing within the group is then carried out by making sure that each peer within the ad hoc group has complete knowledge of all the data

cached/stored within the group. Access to such data from any of the mobile terminals, belonging to the group, leads to copying it locally, if not already cached. Data coherency is maintained by enforcing a conservative coherency protocol that takes into account mobile devices and wireless network constraints, and provides an effective support for collaborative work, since collaborating users must have the same version of the shared data (see §2.2). Finally, in order to enhance data availability and prevent data miss, useful data for collaborative work have to be rationally replicated on peer devices, with respect to the devices' resources (storage space , energy, etc.) as further detailed in Section 3.

2.1 Secure Group Management

Security is of crucial importance in our context, it is therefore mandatory to ensure end-to-end privacy and integrity of the user's data. However, as our platform aims to run on resource constrained terminals, it is necessary to balance strong security enforcement with resource consumption, and in particular energy. For authentication purposes, a trusted third party (e.g., home server) issues Digital Certificates (DC) [21] to users' devices after proper verification of their IDs. The use of DC avoids the need for a trusted, online, third party. A DC includes the user's public key, validity period, the domain name (defined by the home server, e.g., the name of the European project) to which the user belongs, etc. The DC is signed by the trusted third party's private key. Thus, when two nodes wish to authenticate each other upon joining/creating a group, they announce each other with their Digital Certificates. Each of them then verifies the genuineness of the certificate by using the signature on the certificate (every peer must know the third party's public key). Then, the authenticated peers participate in a Group Key Agreement (GKA) protocol to come up with a common secret. A GKA protocol is essentially a contributory protocol designed to provide all the participants with a shared secret key over an open network. This secret is a symmetric key and can be used for securing all further communication. It has to be renewed upon group composition change or at the end of some prefixed security interval. Many such protocols have been proposed in the literature [2], [6], [7], [14].We are currently examining their adaptation for mobile ah hoc environment with respect to devices' resource constraints. In particular, note that the authentication protocol can be integrated with the Group Key Agreement (GKA) protocol thus saving on communication costs.

Group management builds upon a discovery protocol that aims to dynamically detect new peers that are in the communication range of a given group[5], and those that have left it. This task is addressed by service discovery protocols, like SLP (Service Location Protocol), Jini[6], UPnP[7] (Universal Plug and Play), SDP (Bluetooth Service Discovery Protocol) and Salutation[8] [3]. SLP offers a

[5] Notice that a singleton peer is consider as a group.
[6] `http://www.sun.com/jini`
[7] `http://www.upnp.org`
[8] `http://www.salutation.org`

decentralized mode that does not require any *Directory Agent* to collect service announcements in a centralized database. Therefore, it is particularly suitable for ad hoc networks. Each mobile terminal then periodically advertises for the domains[9] it belongs to (e.g., the European project) using its Digital Certificate. When it meets other mobile terminals in its communication range (remember that we consider one hop ad hoc networks) that are advertising for the same domain (i.e., peers that it can trust), the group creation takes place after proper devices authentication.

For a given period of time, a peer is chosen to lead the group. The leader then checks the value of the *group variable*, which gives the group composition, with the list of nodes' IDs returned by the discovery service. If these two values differ, the group must integrate the (potential) newcomers, and/or discard the (potential) nodes that have left. In addition, the group leader makes sure that peers belonging to the group have a global view of data cached within the group and other peers situation (available resources for collaboration). The leader election ensures that the same peer will never be a leader for two successive periods, in order to distribute the charge of group management over all the peers involved (see [4] for more details).

2.2 Data Management

Data management ensures that the user accesses the most recent data version available within the ad hoc group, and guarantees data coherency within the group. It enforces, within the ad hoc group, a conservative coherency protocol that accounts with mobility constraints (devices' resources, network connectivity) (see [5] for more details). This stems from our concern for effectively supporting collaborative work within ad hoc groups. Therefore, peers belonging to an ad hoc group must access the same version of shared information.

The coherency protocol is based on two elements: an *exclusive writer protocol* adapted to mobile environments, and a log, named *Coherency Control List* (CCL). The *exclusive writer protocol*, used within ad hoc groups, builds upon a distributed token management protocol: a unique *token*, is associated with each shared data and created upon first access to the data within an ad hoc group. Each peer must request the token ownership to be able to modify the corresponding data. Therefore, only the peer that owns the token is allowed to modify the data. However, local data can be manipulated (read/write) independently within disjoint groups, provided that the data are synchronized when a group is joined. Update propagation for a given shared data occurs only when any member of the group tries to access the data, because if a group member updates its local copy of particular data, propagation of this update is not necessary if none of the other group members wants to read or update the same data (either cached locally or not). Furthermore, updates are propagated only to the peer which is to access the data. This follows from our aim to save energy by reducing communication among peers.

[9] As defined by its home server, e.g., the European project.

The Coherency Control List is used for incremental lazy update propagation, and serves two purposes: (i) achieving coherency among replicas within an ad hoc group by the lazy update propagation from the current token owner to the peer which is to access its copy; (ii) handling divergences due to non-synchronized concurrent updates in different ad hoc groups which may ultimately require user intervention since fully automated conflict resolution can not be achieved [8], [23].

3 Adaptive Data Replication for Enhanced Availability

To enhance data availability, an adaptive data replication protocol with respect to mobile devices constraints is used. In the context of collaborative work within mobile ad hoc groups, data may become unreachable if the peers storing them suddenly disconnect. On the other hand, excessive or systematic data replication in order to address unforeseen disconnection leads to unnecessarily overloading the group's peers and in particular to greatly increasing their energy consumption and storage space. Thus, useful data for collaborative work within a group have to be rationally replicated on peer devices, with respect to the devices' resources. Following our assumptions, mobile users are able to form ad hoc groups and to share their data in a collaborative manner. As previously stated, existing mobile distributed systems do not manage replication with respect to devices' resources, nor support transparent collaborative data sharing within ad hoc groups, although some of them support collaborative database applications within infrastructure-based WLAN [8]. We aim to achieve data availability within mobile collaborative ad hoc groups through an adaptive peer-to-peer data replication protocol with respect to devices' specific profiles in order to minimize resource consumption (in particular energy).

Each peer caches the set of data necessary for its local autonomy and/or collaborative work. When a peer meets other trusted peers in its local communication range, they form a collaborative ad hoc group. Then, any peer within the group knows about the other peers' situations. Peers profiles serve to identify whether peers can be involved in increasing data availability according to their specific situation. A profile thus indicates: (i) available local storage space for sharing, (ii) whether the peer is able to store replica files other than those that are accessed on the peer depending on available energy, and (iii) expected time within the group, following user's indications (e.g., a diary). Notice further that by construction of ad hoc group management each peer knows both the data that are stored on the various peers of the group, and the situation of each of these peers. Our replication scheme then distinguishes two replica types for a data: (i) Work Replicas are stored on the peer according to data access, either before joining or when part of the group, and (ii) Preventive Replicas are generated by replicating/updating data among the Work Replicas, in order to maintain an up-to-date copy of the data within the collaborative group depending on the peers' profiles. The choice of which peers should store Preventive Replicas is determined by combining the peers' profile with semantic relationships between

already stored data on the considered peer and the generated Preventive Replica (see §3.2).

3.1 Peers Profile

Within a collaborative ad hoc group, data are distributed among peers belonging to the group. The main reason for data miss is peer disconnection. We can distinguish two types of disconnections: (i) voluntary disconnection, e.g., when the user does not want to stay within the group any longer; and (ii) involuntary disconnection, essentially due to a low battery for power-unplugged devices. Therefore, we define the peer's profile, in order to anticipate decreasing availability generated by these disconnections. The peer's profile provides the following information:

- **Available energy:** this can be approximated using the monotonously decreasing linear function[10]:

$$Energy(t) = init - \frac{init}{battery} \times t \tag{1}$$

  where *init* is the initial battery value at the time the peer joined the group, and *battery* is the estimated battery life defined as a function of the peer's charge and is periodically recomputed. $Energy(t)$ serves to determine whether the peer is able to replica files other than those that are accessed locally. Indeed, receiving a new replica involves wireless communication and thus energy consumption [10].
- **Expected time within the group:** this can be estimated using,e.g., user's indications or his/her diary. It can be approximated by the following linear function of time:

$$Time(t) = cst - t \tag{2}$$

  where *cst* is the expected time spent within the group.
- **Available local storage space for sharing:** this is given by the function $Space(t)$, whose value depends on the peer's replication rate and has to be recomputed periodically.

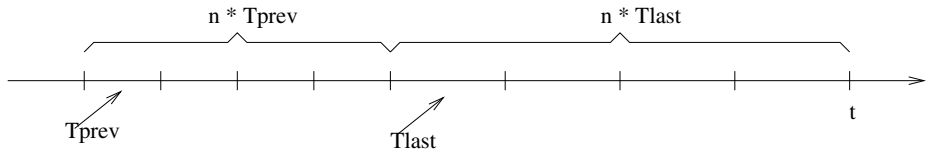

Fig. 1. Group Profile Update Periodicity

[10] Notice that we consider power-unplugged devices

The profile of any peer is always locally available. The ad-hoc group profile is then periodically updated by gathering fresh profiles from the peers belonging to the group. This period is dynamically adapted according to the group's evolution, considering that peers' mobility has a greater impact than changes in individual profiles. Indeed, peers' motion leads to modifying the number of replicas within the group. So, by adapting the update of the group profile to the group's dynamics, we control the replication rate while minimizing communication. The period is initially set to a given value T. Using *statistical modal class* the duration of the period T is adapted every n periods as follows. Let t be the current time and $\gamma = \frac{C_{prev}}{C_{last}}$, with C_{last} being the number of changes[11] over the last n periods (i.e. over $[t-(n \times T), t]$) and C_{prev} being the number of changes over $[t-(2 \times n \times T), t-(n \times T)]$ (see Figure 1). Then, if γ is smaller than one, T should be decreased for the next n periods because the group has been changing less frequently over the last n periods than it changed over the n previous periods. The new value of T then becomes equal to:

$$T' \times (1 + \frac{C_{prev} - C_{last}}{n}) \tag{3}$$

with T' being the previous value of T. On the other hand, T should be increased if γ is larger than one. We get the following new value for T:

$$T' \times (1 - \frac{1}{n \times C_{last}} \times \sum_{i=1}^{max_c} (i \times n_{ci})) \tag{4}$$

where max_c denotes the maximum number of changes over a period for the last n periods of duration T' and n_{ci} gives the number of periods during which there were i changes over the last n periods.

Over a given period T', we can associate one of the following attributes with each peer's profile functions $Energy(t)$, $Time(t)$, and $Space(t)$. Considering $Energy(t)$, we have:

- *Optimal*, the function is maximal over T' and superior to the given threshold e_{th}. This thresholds is tuned by the user or the application. Let G be a group of n peers, $Energy_i(t)$ is the function for available energy estimation for $peer_i$'s profile. $Energy_i(t)$ is *optimal* if and only if

$$\begin{gathered} \forall t \in [0, T'], \forall j = 1, ..., n \wedge j \neq i / peer_j \in G \\ Energy_i(t) \geq e_{th} \wedge Energy_i(t) \geq Energy_j(t) \end{gathered} \tag{5}$$

- *Acceptable*, the function is *acceptable* over T' if it is not *optimal*, but still greater that the threshold. $Energy_i(t)$ is *acceptable* if and only if

$$\forall t \in [0, T'], Energy_i(t) \geq e_{th} \tag{6}$$

- *Weak*, the function is considered to be *weak* over T' if it goes below the given threshold. $Energy_i(t)$ is *weak* if and only if

$$\exists t \in [0, T'], Energy_i(t) \leq e_{th} \tag{7}$$

[11] By changes we mean peers' motion that leads to joining or leaving the group

Table 1. Peer's profile classification

$Energy(t)$	$Time(t)$	$Space(t)$	**Peer's Profile**
Optimal	Optimal	Optimal	Optimal
Optimal	Optimal	Acceptable	Optimal
Optimal	Optimal	Weak	Acceptable
Optimal	Acceptable	$\forall$	Acceptable
Optimal	Weak	$\forall$	Weak
Acceptable	Optimal	$\forall$	Acceptable
Acceptable	Acceptable	$\forall$	Acceptable
Acceptable	Weak	$\forall$	Weak
Weak	$\forall$	$\forall$	Weak

The definition of attributes is applied similarly to functions $Time(t)$, and $Space(t)$, using their respective thresholds t_{th}, and s_{th}.

Then, combining the functions' attributes, we associate the previous attributes to the peer's profile as indicated in Table 1. An *optimal* profile designates a peer able to actively collaborate and share its resources within the ad hoc group. In contrast, a *weak* profile designates a peer with few resources, and hence, unable to share them within the group.

The attributes of $Energy(t)$, $Time(t)$, and $Space(t)$ do not have the same weight for determining the profile attributes. Indeed, $Energy(t)$, and $Time(t)$ indicate, respectively, the available energy and the expected time within the group. Moreover, they represent the evaluation of parameters that can generate a peer's disconnection. Thus, if one of them has a *weak* attribute, the peer's profile is considered *weak*. On the other hand, $Space(t)$ does not have the same influence on a peer's profile. Therefore, even if $Space(t)$'s attribute is *weak*, a peer's profile is still *acceptable* because replacement policies can be used to release some storage space. A peer's profile attributes serve to adapt replication to the peer's available resources.

3.2 Work and Preventive Replicas

When a peer joins a group, its locally cached data are considered as Work Replicas (WR). Furthermore, Work Replicas are generated upon access demands to non locally cached files. Those replicas are stored locally on the requesting peer regardless of its available resources. A replacement policy (LRU) is used in order to release storage space and hence delete old or seldom accessed Work Replicas provided that at least an other Work Replica is cached within the group.

Within an ad hoc group, updates among Work Replicas are lazily propagated. Therefore, coherency checks, based on CCLs, are performed upon actual access to Work Replicas. Their coherency is managed following the *log-based exclusive writer* as described in Section 2.2.

When the profile of a peer p caching the latest version of a Work Replica (p is the last writer on this replica) becomes *weak*, p propagates an update (if not already done) to another Work Replica stored on a peer with an *optimal* profile

Table 2. Replicas Management within an Ad Hoc Group

Within an Ad Hoc Group		Events	Actions
Their are many WRs	only one replica is up-to-date	the peer caching the up-to-date replica becomes *weak*	update a WR, that becomes also a PR
	many WRs are up-to-date	a peer caching a WR becomes *weak*	Do nothing
Their is one WR	-	the peer caching the WR becomes *weak*	create a PR on an appropriate peer

if one exists, or with an *acceptable* profile otherwise. This updated Work Replica also becomes a Preventive Replica (PR). Notice that the lazy update propagation is no longer respected in this case, updates are systematically propagated to the Preventive Replica. If peer p is the only peer caching this Work Replica (determined using the group variable), then a new replica (Preventive Replica) has to be created, as summarized in Table 2. The choice of which peer should store the new Preventive Replica is determined by combining its profile with semantic relationships between data as further described below.

Preventive Replicas serve to maintain an up-to-date copy within the group. This is actually our minimal replication rate. Some peers can already cache Preventive Replicas created within other ad hoc groups. These replicas are considered to be Work Replicas when these peers join a new group and become part of the peers' regular local cache. Furthermore, a Preventive Replica within an ad hoc group becomes a Work Replica when: (i) an access demand is made on it; (ii) a new Work Replica is created within the group on a peer with at least *acceptable* profile (see Table 1). This can happen either when a new peer joins the group, or a peer belonging to the group requires an access to the given replica.

The choice of which peer of an ad hoc group, should store a Preventive Replica is first guided by its profile. Therefore, peers with *weak* profiles are never going to be selected to store a Preventive Replica. Then, peers already caching a corresponding Work Replica are chosen. In this case, the chosen Work Replica is updated regardless of lazy update propagation. Hence, it also becomes a Preventive Replica. If none of the peers with an *optimal* or an *acceptable* profile are caching a corresponding Work Replica, then we rely on semantic relationships between data cached on peers and the Preventive Replica to be created.

Let p_1 be a peer caching the latest version of a Work Replica f_1W. The profile of p_1 becomes *weak* so p_1 has to create a Preventive Replica. Assuming, that none of the peers caching f_1W has a convenient profile, then p_1 has to chose a peer caching the most semantically related set of data to f_1W in order to create its Preventive Replica. The choice of semantically related data is specific to data type. For instance, when considering files, the peer p_1 chooses semantically related set of files to f_1W as follows:

- as a general rule, files cached on p_1 or in the same directory as f_1W are the most closely related. File naming also provides important clues to semantic relationships, e.g., files with the same name but different extensions are closely related.

Table 3. Selecting a Peer for Caching a Preventive Replica

Profile	Is Caching a Work Replica	Is Caching Semantically Related Data	Peer's Selection
Optimal	true	∀	Optimal Selection
Optimal	false	true	Acceptable Selection
Optimal	false	false	Weak Selection
Acceptable	true	∀	Acceptable Selection
Acceptable	false	true	Acceptable Selection
Acceptable	false	false	Weak Selection
Weak	∀	∀	Not Selected

- p_1 computes semantic distances between f_1W and locally cached files, as described in [16], in order to create clusters of semantically related files. Notice that f_1W can belong to different clusters. Peers locally caching n files belonging to f_1W's clusters are considered to be semantically related to f_1W. The parameter n is tuned by the user or the application.
- After the clustering operation, p_1 orders the peers according to their eligibility to store the Preventive Replica. Table 3 gives all the cases involved in a peer's selection for storing a Preventive Replica. *Optimal* selection peers are the best candidates to store the Preventive Replica. Otherwise, a peer among *acceptable* selection peers is chosen. If there are no *optimal* or *acceptable* selection peers within the group, then a *weak* selection peer is chosen. A *weak* selection peer does not cache Work Replicas or semantically related data, but its profile allows it to become involved in increasing data availability. However, if the group contains only not selected peers, data availability cannot be maintained without compromising these peers' resources, which are already *weak*.

4 Assessment

In order to evaluate our collaborative ad hoc group and data availability management protocol in terms of performance and data overhead, we have implemented our middleware service within a file system. The resulting distributed file system for mobile ad hoc data sharing [22], [4] is implemented in Objective Caml 3.04[12]. Performance measurements concern in particular the group management, coherency management, and replication protocol. They have been done on a platform of ten laptops (Compaq Armada M700, M300, and Toshiba Satellite 1800-911) with a 500 MHz Pentium III CPU, 256 KB of cache, 200 MB of RAM and a 10 GB hard disk running under Linux Mandrake release 7.1 (2.2.15-4mdk) operating system. The wireless LAN is IEEE 802.11b in ad hoc mode using the Lucent 11Mb WaveLAN "'SILVER"' PC Card wireless interface.

Figure 2 gives the time taken for creating a group, leaving and joining a group, which are linear with the group size. The main cost of group creation lies in detecting trusted peers and sending the meta-data (group composition,

[12] `http://caml.inria.fr/index-eng.html`

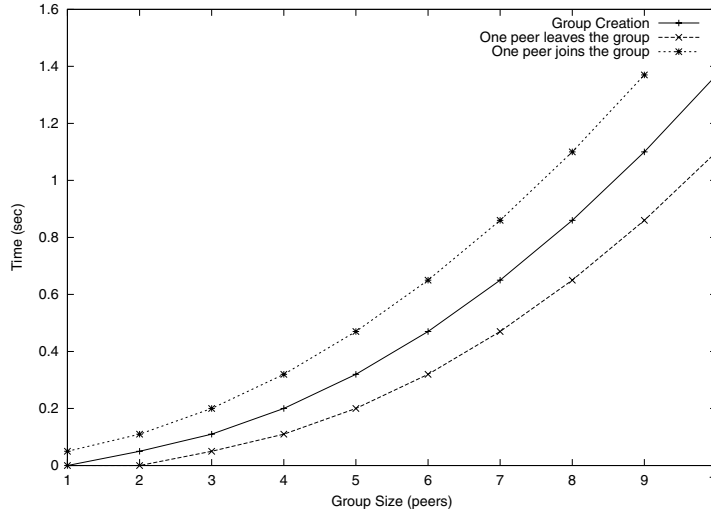

Fig. 2. Group Management Overhead

Table 4. Cost of Getting the Token

Group size (*peers*)	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Time (*sec*)	0.00949	0.01281	0.01982	0.02129	0.02645	0.02977	0.03493	0.03825	0.04341
Energy (*mW.sec*)	1.58	2.91	5.4	6.67	9.08	11.87	15	18. 51	22.38

CCLs, profiles, etc.) by the group leader so as to allow peers belonging to the group to have a global view of shared data and other peers' profiles.

Table 4 shows that the cost, in terms of response time and energy consumption, of getting the token (without the updates) to modify a data, is constant within a particular group. Note that the cost of an additional peer in terms of response time is only about 0.005 sec. However, the cost of one additional peer in terms of energy depends on the group size. Indeed, being in ad hoc mode means that even non destination peers consume energy when messages are being sent. That is why, adding one peer to a group of three peers does not have the same energy impact as adding one peer to a group of ten peers. However, the cost in terms of energy consumption still very low. Note that being idle in an ad hoc configuration costs 843 mW.sec.

The cost, in terms of response time, of getting updates is proportional to the update size, as depicted in Figure 3. Using our coherency protocol, getting access to coherent data locally depends on the update size, while group size affects only slightly the performance of our protocol. However, the energy consumption depends on both the group size and the update size. It is given (based on equations from [10]) by:

$$\varepsilon_{msg} = (0.1 + 0.25 * n) * size + 221 + 163 * n \tag{8}$$

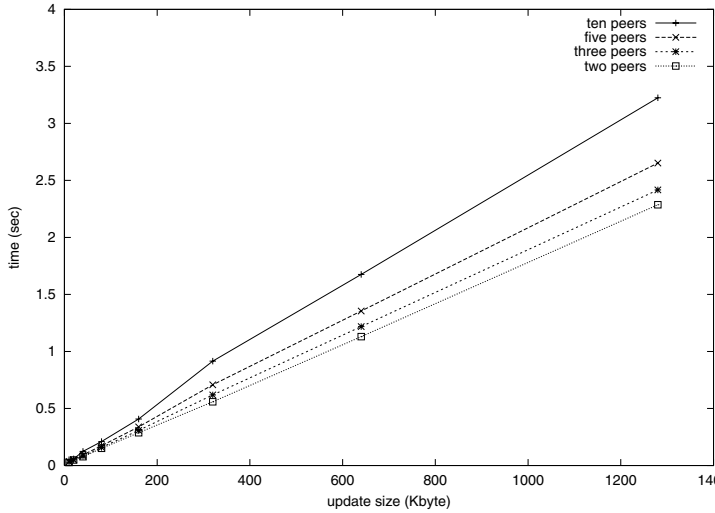

Fig. 3. Cost of Getting Updates

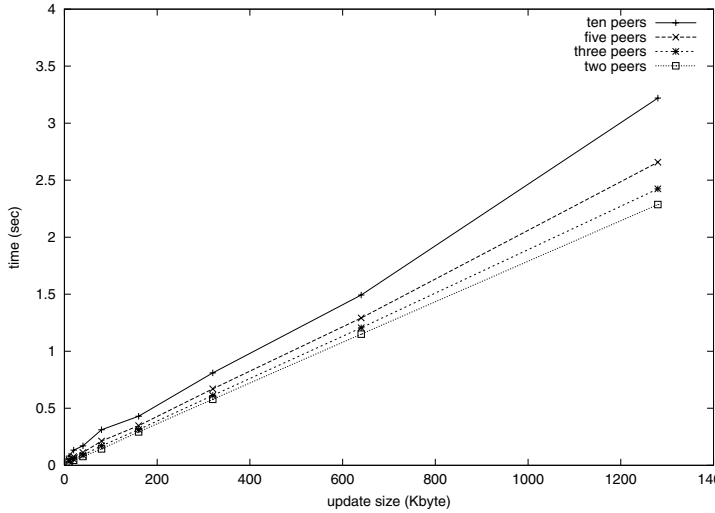

Fig. 4. Lazy Update Propagation

where n is the ad hoc group size and *size* the update size.

Figure 4 and 5 show the benefits of our coherency management protocol (lazy update propagation) in terms of response time. Indeed, the response time, when lazily propagating updates (in the case of write operations) ranges from 0.0269 sec to 3.219 sec according to the group size and the update size, as depicted by Figure 4. However, when the updates are propagated whenever they occur, which corresponds to an optimistic coherency management (note that

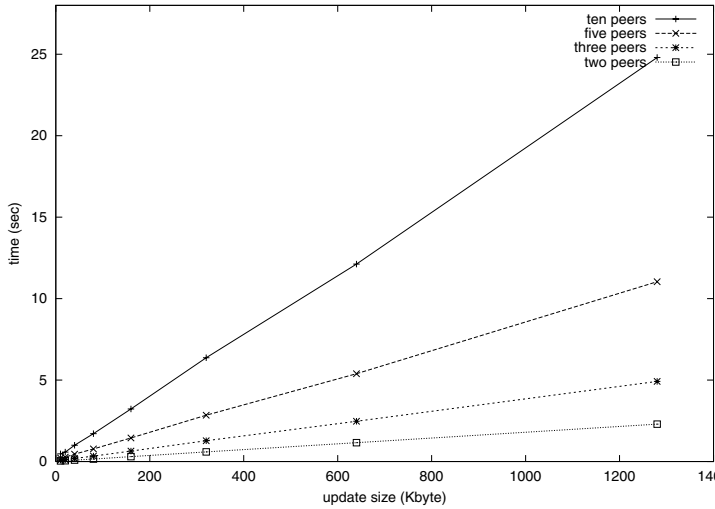

Fig. 5. Optimistic Update Propagation

peers belonging to group are in the communication range of each other), the response time (in the case of write operations) ranges from 0.0361 sec to 24.8 sec according to the group size and the update size, as depicted by Figure 5. Let compare these two coherency management protocols for a given scenario. Initially, we have an ad hoc group of four peers, all caching the data. We have made four successive write operations by different peers, and we suppose that update size is 640 Kbytes for each write operation. Following our protocol, the time taken to first write access a coherent data file is equal to 0.019 sec, for the second successive write operation it is equal to 1.20 sec, for the third it is equal to 1.22 sec, and for the last write operation to 1.27 sec (this results from our lazy update propagation). However, if the updates are propagated whenever they occurs, in the same scenario, the time taken to write access a coherent data file is equal to 3.639 sec per write operation. Thus, by lazily propagating updates, we save time and communication for the four write operations, which is equivalent to 7218 mW of energy consumption.

Concerning the data overhead introduced by our availability management protocol, we can distinguish two types of overhead: the local storage overhead, it includes the data necessary for coherency and replication management (group composition, CCLs, profiles, etc.) and that are locally stored; the message overhead, which includes the data sent within messages in addition to the actual updates. The local storage overhead is given by (in Bytes):

$$\begin{aligned} Local\_overhead &= 3 \times int + 2 \times bool + (5 \times real + 16 \times char) \times p \\ &+ \sum_{j=1}^{G} (16 \times char \times p_j + int \times Modif_j) + \sum_{i=1}^{W+R} (16 \times char + size\_ccl_i) \end{aligned} \quad (9)$$

$$size\_ccl_i = int + \sum_{j=1}^{G}(16 \times char \times p_j + int \times Modif_j)) \quad (10)$$

Where p is the number of peers caching locally the data, $Modif$ is the number of write operations, G is the number of different group configurations where updates occur, W is the number peers waiting for write, R is the number peers waiting for read, and int, $real$, $char$, and $bool$ the sizes of respectively integer, real, character, and boolean variables. The message overhead[13] is given by (in Bytes):

$$Message\_overhead = 2 \times int + file\_name \times char + \sum_{j=1}^{G}(16 \times char \times p_j + int \times Modif_j) + sum_{i=1}^{W+R}(16 \times char + size\_ccl_i) \quad (11)$$

Where $file\_name$ is the length of the file name. In these equations we considered that peer's ID is a string of 16 characters.

Using our availability management protocol, the local storage overhead as well as the meta-data sent in addition to the updates are negligible regarding the average size of data files, e.g., the message overhead represents 0.41 percent of the whole message to be sent. Further evaluations using simulation should be done, in order to measure the performances of our protocol within different scenarios and profiles including high peers mobility, this is part of our future work.

5 Conclusion

Mobile ad hoc networks are particularly suitable to form small collaborative groups in order to share and manipulate common data. Until now, work concerning ad hoc networks mainly deals with routing protocols without taking into account data management within these networks. Data management over wireless networks relies essentially on adequate replication management for enhanced availability. Manipulating data within mobile ad hoc collaborative groups requires two core functionalities:

1. Secure group management that ensures: (i) transparent group creation using a discovery service to detect trusted partners (their devices) that *belongs* to the group; (ii) adaptation to the group dynamics, so that members can leave or join the group at any time; (iii) secure group communication through an encryption protocol that must account with devices' resources and network connectivity.
2. Data management that ensures: (i) that members access the most recent data version available within the ad hoc group, (ii) data coherency within the group through a conservative coherency protocol to effectively support collaborative work, since members must have the same version of shared

[13] Notice that profile's parameters are not involved in the message overhead

data; (iii) adaptive data replication to avoid data loss if the devices storing them suddenly disconnect.

Existing distributed systems that deal with mobility do not manage replication with respect to wireless devices resources, nor support transparent collaborative data sharing within ad hoc WLAN groups.

Therefore, we have designed and presented a middleware service that allows collaborative data sharing among ad hoc groups that are dynamically formed according to the connectivity achieved by the ad hoc WLAN. Our service enhances, in particular, data availability within mobile ad hoc collaborative groups, and integrates a new adaptive data replication protocol for mobile terminals.

We believe that data availability is a crucial requirement in such environment, and it is one of the prominent QoS criteria for collaborative work. Our replication protocol is based on a combined coherency management (using both optimistic and conservative schemes) in order to account for various connectivity as enabled by today's wireless networking capabilities. It is further based on two replica types: Work Replica, and Preventive Replica. The former is generated according to access demands to non locally cached files, and benefits from the lazy update propagation following our *log-based exclusive writer* coherency protocol. The latter serve to maintain an up-to-date copy within the group, and is generated according to the profile of the peer holding the latest version of the corresponding Work Replica. The choice of which peer should store a Preventive Replica is determined by combining the peers' profile with semantic relationships between locally stored files and the generated Preventive Replica.

It is shown that using our middleware service, the meta-data overhead generated is negligible, the response time grows linearly with update size, while group size affects it slightly, and, the lazy update propagation saves time and energy consumption. We are further working on evaluating performances of our protocol, by simulating various scenarios and peers' profiles.

Acknowledgments

The authors would like to acknowledge the work carried out by Raghav Bhaskar and Daniel Augot regarding security management in mobile ad hoc networks and their proposition to adapt group key agreement protocol to such environment. We would like also to acknowledge the contribution of David Mentré, Anis Ben-Arbia, and Animesh Pathak to the design and the implementation of the ad hoc group management protocol and the coherency management protocol. This work has been partially funded by the ITEA VIVIAN Project (ITEA 99040).

References

1. Micah Adler and Christian Scheideler: Efficient communication strategies for ad-hoc wireless networks. In *ACM Symposium on Parallel Algorithms and Architectures*, pages 259–268, 1998.

2. G. Ateniese, M. Steiner, and G. Tsudik: New multiparty authentication services and key agreement protocols. *IEEE Journal of Selected Areas in Communications*, 18(4):pp. 1–13, 2000.
3. C. Bettstetter and C. Renner: A comparison of service discovery protocols and implementation of the service location protocol. In *Proceedings 6th EUNICE Open European Summer School: Innovative Internet Applications*, 2000.
4. Malika Boulkenafed, Anis Ben Arbia, David Mentré, and Valérie Issarny: AdHocFS: Sharing files in wlans, 2002. http://www-rocq.inria.fr/solidor/work/AdHocFS.html.
5. Malika Boulkenafed and Valérie Issarny: Coherency management in ad-hoc group communication. In *Proc. of the joint VIVIAN-ROBOCOP workshop on Software Infrastructures for Component-Based Applications on Consumer Devices*, Lausanne, Switzerland, September 16, 2002.
6. M. Burmester and Y. Desmedt: A secure and efficient conference key distribution system. In LNCS Springer, editor, *Proc. of Eurocrypt'94*, pages pp. 275–286, 1995.
7. W. Diffie, D. Steer, L. Strawczynski, and M. Weiner: A secure audio teleconference system. In LNCS Springer, editor, *Proc. of Crypto'88*, 1988.
8. W. K. Edwards, E. D. Mynatt, K. Petersen, M. J. Spreitzer, D. B. Terry, and M. M. Theimer: Designing and implementing asynchronous applications with bayou. In *Proc. of the Symp. on User Int. Software and Tech.*, Banff Alberta (Canada), Oct. 1997.
9. Todd Ekenstam, Charles Matheny, Peter L. Reiher, and Gerald J. Popek: The bengal database replication system. *Journal of Distributed and Parallel Databases*, 9(3):187–210, 2001. citeseer.nj.nec.com/449358.html.
10. L. Feeney and M. Nilsson: Investigating the energy consumption of a wireless network interface in an ad hoc networking environment. *In proc. of the IEEE Infocom*, 5(8), 2001. www.sics.se/ feeney.
11. J. D. Gibson: *The Mobile Communication Handbook*. CRC Press, 2nd edition, 1999.
12. Richard G. Guy, Peter L. Reiher, David Ratner, Michial Gunter, Wilkie Ma, and Gerald J. Popek: Rumor: Mobile data access through optimistic peer-to-peer replication. In *ER Workshops*, pages 254–265, 1998. citeseer.nj.nec.com/guy98rumor.html.
13. T. Imielinski: *Mobile Computing*. Kluwer Academic Publishers, 1996.
14. M. Just and S. Vaudenay: Authenticated multy-party key agreement. In *Proc. of ASIACRYPT'96*, 1996.
15. John Kubiatowicz, David Bindel, Yan Chen, Patrick Eaton, Dennis Geels, Ramakrishna Gummadi, Sean Rhea, Hakim Weatherspoon, Westly Weimer, Christopher Wells, and Ben Zhao: Oceanstore: An architecture for global-scale persistent storage. In *Proceedings of ACM ASPLOS*. ACM, November 2000. citeseer.nj.nec.com/kubiatowicz00oceanstore.html.
16. Geoffrey H. Kuenning: Seer: Predictive File Hoarding for Disconnected Mobile Operation. PhD thesis, University of California, Los Angeles, May 1997.
17. P. Kumar and M. Satyanarayanan: Supporting application-specific resolution in an optimistically replicated file system. In *Proceedings of the 4th Workshop on Workstation Operating Systems*, 1993.
18. Haiyun Luo, Paul Medvedev, Jerry Cheng, and Songwu Lu: A self-coordinating approach to distributed fair queueing in ad hoc wireless networks. In *INFOCOM*, pages 1370–1379, 2001. citeseer.nj.nec.com/luo01selfcoordinating.html.

19. Sergio Marti, T. J. Giuli, Kevin Lai, and Mary Baker: Mitigating routing misbehavior in mobile ad hoc networks. In *Mobile Computing and Networking*, pages 255–265, 2000. citeseer.nj.nec.com/marti00mitigating.html.
20. Cecilia Mascolo, L. Zanolin, and Wolfgang Emmerich: Xmiddle: an xml based approach for incremental code mobility and update. *Journal of Automated Software Engineering*, 9(2), 2002.
21. Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone: *Handbook of Applied Cryptography*. CRC Press, 4th edition edition, 1996. http://www.cacr.math.uwaterloo.ca/hac/.
22. Davide Mentre, Malika Boulkenafed, and Valérie Issarny: AdHocFS: A serverless file system for mobile users. Research Report 4303, INRIA-Rocquencourt, 2001.
23. K. Petersen, M. J. Spreitzer, D. B. Terry, M. M. Theimer, and A. J. Demers: Flexible update propagation for weakly consistent replication. In *Proc. Symp. on Operating Systems Principles (SOSP-16)*, pages 288–301, Saint Malo, Oct. 1997.
24. Amir Qayyum: *"Analysis and evaluation of channel access schemes and routing protocols for wireless networks"*. PhD thesis, University of Paris Sud, Orsay, France, November 2000.
25. P. Reiher, J. Popek, M. Gunter, J. Salomone, and D. Ratner: Peer-to-peer reconciliation based replication for mobile computers. In *European Conference on Object Oriented Programming '96 Second Workshop on Mobility and Replication*, 1996. citeseer.nj.nec.com/5097.html.
26. M. Satyanarayanan, James J. Kistler, Puneet Kumar, Maria E. Okasaki, Ellen H. Siegel, and David C. Steere: Coda: A highly available file system for a distributed workstation environment. *Journal of IEEE Transactions on Computers*, 39(4):447–459, 1990. citeseer.nj.nec.com/satyanarayanan90coda.html.
27. H. Yu and A. Vahdat: Design and evaluation of a continuous consistency model for replicated services. In *Proc. of the 4rd Symposium on Operating Systems Design and Implementation*, 2000. citeseer.nj.nec.com/yu00design.html.