Data Lockers: Mobile-Agent Based Middleware for the Security and Availability of Roaming Users Data*

Yolanda Villate**[1], Arantza Illarramendi[1], and Evaggelia Pitoura[2]

[1] Languages and Systems Department. University of the Basque Country. Apdo. 649, 20080, Spain. {jibvipey, jipileca}@si.ehu.es

[2] Computer Science Department. University of Ioannina. GR-45110 Ioannina, Greece. pitoura@cs.uoi.gr

Abstract. Accessing remote data anywhere and at anytime constitutes an important advantage in many business environments. However, when working with mobile devices, users face many problems related to: (1) device restrictions: mobile devices are resource-constrained, more vulnerable and fragile than stationary devices (*devise exposure problems*) and, (2) the communication media: wireless communications are often unstable, asymmetric and expensive (*media problems*). To alleviate these problems, we present a service, the *Lockers Rent Service*, for keeping the data of mobile users in a secure and safe space in a proxy at the fixed network, thus providing a solution to the device exposure problems. Data stored in data lockers are available even when the mobile device is disconnected. Furthermore, specific tasks are carried out at the fixed network on behalf of the mobile user, in this way relieving the media problems. The architecture of the *Locker Rent Service* is based on mobile agents.

1 Introduction

The sheer rapidity of the spread of wireless and Internet technologies is favoring a new telecommunication revolution. The possibility of accessing remote data anywhere and at anytime constitutes an important advantage in many business environments. However, when working with mobile devices, users face many obstacles related to the nature of the devices and the communication media used.

In [14], we introduced the preliminary version of ANTARCTICA[1] system. The main goal of this system is to provide a set of data services that enhance the

* This work is supported by *CICYT: Comisión Interministerial de Ciencia y Tecnología*, Spain. [TIC97-0962]; and MOVISTAR, a Spanish Cellular Phone Company.

** The work of this author was supported by a grant of the University of the Basque Country.

[1] ANTARCTICA: Autonomous ageNT bAsed aRChitecture for cusTomized mobIle Computing Assistance.

capabilities of Mobile Units (MU) and offer new possibilities to mobile users. The architecture of ANTARCTICA is based on the use of the *Client/Intercept/Server* model [11, 10] that incorporates modules and agents both at the mobile device and at an intermediary element, or proxy, situated at the fixed network called the *Gateway Support Node*[2](GSN). In this paper, we consider a scenario in which a cellular phone company offers services that mobile users, clients of the company, can contract. In such a scenario, the GSNs belong to the cellular phone company. The number of GSNs in the network depends on the number of users.

The goal of this paper is to describe the design and implementation of a service central to ANTARCTICA: the *Rent of Lockers Service*, introduced in [15]. This service incorporates mechanisms that allow mobile users to rent storage space at the GSN. The service offers a variety of types of lockers so users can choose the one that better fits their needs and preferences. Independently of the specific type of locker, user or mobile computing device, the *Locker Rent Service* offers several advantages to the users of mobile units:

1. **Storage space:** The locker becomes an extension of the MU's disk at the fixed network.
2. **Data protection:** The mobile client is the owner of the data stored in the locker. The data stored in the locker are protected against unauthorized accesses and modifications as well as any unexpected failures.
3. **Battery power saving:** The mobile client can stay disconnected for longer periods of time, while the data obtained for it are sent to its locker and stored there until it reconnects or explicitly asks for them.
4. **Wireless communications optimizations:** The space in the locker can be used to store data until it is possible or desirable to send them to the MU. Before sending data to the MU, the data can be preprocessed, filtered or adapted to the needs of the mobile unit and its user. The locker can also be used to store data needed to perform tasks that can be developed and executed at the GSN, thus avoiding their transmission to the MU.
5. **Flexibility and adaptability:** The implementation of the locker service using mobile agents provides for the dynamic adjustment of the space allocated to each locker. It also provides for the easy adaptation of the locker, by facilitating the dynamic creation of lockers tailored to the users needs. Furthermore, it physically supports mobility of the locker to follow the movements of the MUs.

Lockers can be rented by individual users (*private locker*) or by groups of users (*shared locker*). A **private locker** is related to a single user; the data stored belongs to that particular user and can be accessed or modified only by authorized agents representing the user. A **shared locker** is a locker rented by a group of users. A shared locker distinguishes between data to be used by all users in the group and data for each particular user, by managing sub-lockers for each of them. Moreover, the space in the locker that is shared by all users

[2] The Gateway Support Node name is borrowed from the General Packet Radio Service (GPRS). We take GSM and GPRS as the cellular network model for our work.

of the group constitutes an encounter place for these users who are able to communicate, share data, interchange messages, write messages for the rest of the group in a blackboard service provided and collaborate together.

There are two lines of research related to our work. The first line includes research that relies on the idea of renting resources: in [2] the main focus is on accounting CPU cycles and communications consumption made by the hosted processes; Xdrive [16] is a commercial product that offers disk space located at safe hosts (situated at the X:drive company), accessible through the Internet using a Web browser, or a desktop application to make this space appear as a new disk partition in the user's computer. On the contrary, the main idea of the *Locker Rent Service* is to provide mechanisms that allow mobile users to use storage space that is external to their mobile computers but remains always as close as possible to their actual physical location. The space available in the ANTARCTICA lockers is intended to be used on behalf of the user by the agents working for him. Moreover, the locker is an autonomous and auto-managed extension of the MU, dynamically customizable to the user needs and network status, that gives the agents more autonomy to work for long periods of time even when the MU is disconnected. The second line of related research exploits the use of *proxies* and software agents [6, 4, 5, 3, 7, 9]. To our knowledge, our work is the first one to combine both aspects, by employing proxies and agents to provide users with storage external to their mobile computers. However, our work can benefit from other works, such as the CODA project [6], by implementing the ideas and techniques they propose for the maintenance of data consistency among copies stored in the MU cache, the GSN and the servers.

The rest of this paper is structured as follows. In Sect. 2, we present the main features of the agents that participate in the service. In Sect. 3, we introduce some advanced features related to the management of the space available and the data storage, the security and the mobility of the lockers. In Sect. 4, we present performance results. Finally, Sect. 5 offers conclusions.

2 Agents that Participate in the Service

The architecture of the *Locker Rent Service* is based on mobile agents. Agents are executed in execution environments called *places* [1]. Mobile agents can travel from one place to another. The proposed service incorporates one place on the user computer called the MU Place, and two places situated in the GSN, the *Inventory Place* and the *Locker Place*.

Mobile agents provide the means to improve performance by: using local resources and services offered at the proxy; supporting asynchronous communications; facilitating an efficient management of data transfer interruptions; and, allowing to reduce the traffic of data in the wireless network. Moreover, our autonomous agents are devoted to serve particular users and thus are customized according to the task they must develop, the preferences of the user they represent, the actual state and characteristics of the computer machine and the network used.

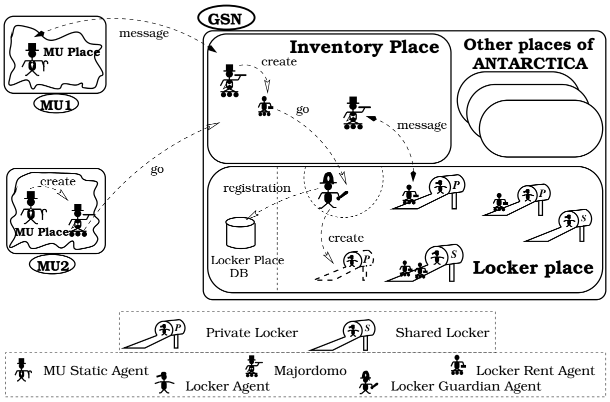

Fig. 1. Elements involved in the Locker Rent service

As shown in Fig. 1, there are five different kinds of agents involved in the *Locker Rent Service*. Each specific kind of agent has a different specialization area and a mission to perform, for which it may need to interact with other agents and places.

The MU Static Agent. At each MU there is a static agent running on a place in the MU, any time the MU is active. This static agent isolates the user's applications from network availability and the specific characteristics of the MU, and is responsible for the administration of the MU's resources.

The Majordomo Agent. The *majordomo* agent is created in the MU by the static agent and launched to a GSN at the fixed network. In each GSN, there is a place called *Inventory* where *majordomo* agents execute and can get the information they need about other GSNs, places and services in the system. Once launched, the majordomo agent remains in the GSN for the time period required, working on behalf of its MU even while the MU is disconnected.

Each mobile unit has its own *majordomo* agent with the aim of providing adequate services to its owner. Communications of the MU with the exterior are surveyed by both the static agent of the MU and its *majordomo* agent in the GSN. These two agents work together in order to adapt and optimize communications and use of the wireless media.

The Pair of Locker Agents. When the *majordomo* agent needs to use the *Locker Rent Service*, it creates an agent called *locker rent agent* and sends it to a *Locker Place*. The *Locker Place* is the place in the GSN that offers the *Locker Rent Service*. Although it would be possible for the majordomo agent to use directly the *Locker Rent Service*, we chose to create a special locker rent agent because it would be difficult for the majordomo agent to use efficiently all the different services offered by the ANTARCTICA system heeding at the same time the different user requirements.

Lockers are implemented using a new kind of static agents that we call *locker agents*. The *locker agents* are created by the *locker guardian agent* (we explain its features next) in the *Locker Place*, and each one of them is assigned to a specific user or group of users, i.e., to their *locker rent agents*. This pair of the *locker agent* and the *locker rent agent(s)* constitutes the locker and takes care of storing the user's data, saving email messages, processing results and communicating with the MU's *majordomo* agent. The fact that both agents have to communicate with each other and interchange data incurs some overhead. However, this interaction is local as both agents reside in the same place, the *Locker Place*. Furthermore, by having both kinds of agents, we can take advantage of specializing them.

The *locker agent* is specialized in interacting with the place and the available resources in order to store and recover data, and assure the privacy and security of its locker. So it makes the characteristics of the place and its resources transparent to the *locker rent agent* and to its user. Moreover, the *locker agent* of a shared locker interacts with a group of *locker rent agents*, acting as an intermediary and maintaining services such as a blackboard where the users can leave messages and data for other users. It also handles concurrent data accesses.

On the other hand, the *locker rent agent* is specialized on interacting with the user's *majordomo* agent. It has the knowledge about the user, as well as semantic knowledge about the data and how to utilize them. In that way, it makes transparent for the *locker agent* the characteristics of the specific user and its MU.

The Locker Guardian Agent. The *Locker Place* offers disk space for renting, so it has mechanisms to administer and monitor the available space, to assign spaces and survey their use, and to register data necessary for billing for the service. In order to get these functionalities, the *Locker Place* has associated a database and a *guardian agent*.

When the *Locker Place* is created a static agent is created within it, the *locker guardian* agent. The *locker guardian* agent is always active, its mission is to control the population of agents in the *Locker Place*, check the authorization and authentication of incoming agents, create and dispose *locker agents*, maintain a registry of the agents in the database (DB) and monitor the use of the resources.

3 Advanced Features

In this section, we introduce some advanced features of the *Locker Rent Service*.

3.1 Dynamic Management of Space

As mentioned in the previous section, lockers are implemented by a pair of agents collaborating together: the *locker rent agent* and the *locker agent*. The latter agent constitutes the locker itself. This solution allows us to manage the locker in a dynamic and flexible manner. There are no physically separated compartments to reserve, instead *locker agents* are created with space assigned in which they can store data. This means that lockers are created as they are rented by new users. Each *locker agent* occupies disk space according to its needs and limitations. When it needs more space, the agent occupies it, and when it needs less, the agent releases space. The space used by the locker is exactly the storage used by the data saved there. Although there is a limit in how much space the agent can use (by contract or specification), flexible policies can be used to optimize the use of space and improve the service provided.

3.2 Storage Management and Metadata

The data stored in a locker are actually text files that contain the raw data (files, e-mails, images, WWW pages, sql results) as well as metadata used to rebuild the object stored in the file to its original format. Metadata also provide summary information about the data stored such as their origin and content; and allow us to deal with large pieces of data, such as large files, images, sql results, by dividing them into smaller pieces that later can be reorganized and put together in order to rebuild the original. This feature can be very useful when considering for example the transmission of data from the MU to the GSN. Instead of sending the whole piece of data in a single message to the MU, the data can be broken into smaller pieces and then send each piece in an independent message. This way, if the communication is broken during the transmission or the MU has to disconnect, the transmission of the data can be resumed later, avoiding the retransmission of the pieces of the file that where already successfully sent and received at the MU.

3.3 Security

In the context of mobile agents, security is always a principal concern because generally agents can not trust the hosts they visit, and hosts can not trust the agents they receive. The protection of agents against malicious hosts is an open problem still without a completely satisfactory solution. The approach most commonly used is to rely on an organizational framework whose hosts can be trusted [13]. This is the case in our system, since the services offered by the GSN are provided within a trusted cellular phone company. Similarly, users are not unknown to the system, since they have signed a contract with the company for renting the services, and they have been assigned an ID and a set of passwords. Thus, the system can authenticate and validate authorized users and their agents.

In particular, the security of hosts against agents can be achieved satisfactorily by using existing techniques [13]. The central issue is to prevent uncontrolled access. In our system, the only agents that come from the outside, i.e. are created from classes unknown to the GSNs, are the *majordomos*. It must be ensured that such agents represent authorized users. To achieve this, credentials are associated with the agents. Credentials correspond to the identity of their user, access level and key. Such credentials are used for the authentication and authorization of the majordomo agents, and for controlling their access to services and data. This is achieved by checking their credentials against the information the system maintains about the registered users. When an agent creates another agent, it passes part of its identification information to the new born agent by issuing it a "passport" containing the necessary information to identify the agent and its creator. That "passport" is sensitive information that needs to be protected against copies, thieves and modifications.

Another important issue is preventing uncontrolled use of the host resources. The programming languages and the agent platforms used provide mechanisms to prevent agents and hosts to access the data or code of others such as separated memory address spaces for the host and each agent. Also, they provide authentication and authorization mechanisms, as well as mechanisms to avoid uncontrolled copying and cloning of agents. To authenticate code and its origins, cryptographic techniques can be used, as for example signatures [12].

Apart from the use of mobile agents, in the *Locker Rent Service*, data stored in a private locker belong to the user and are protected against unauthorized access from other users or their agents. In a shared locker, members of the group owning the locker are allowed to access the data that is specifically stored as common, while each user in the group has a private area where he can store information that nobody else has the right to access. This is achieved by not allowing the *locker agents* access any other disk space but their own. When a *locker rent agent* makes a request to a *locker agent*, the *locker agent* checks the authenticity of the *locker rent agent* and decides to which data it has access to.

3.4 Mobility of the Locker

Private lockers are typically created in the GSN under the coverage of which the MU is located. In our system, the *majordomo* agent is located as close as possible to the MU it belongs to. This means that when a MU moves to an area covered by a different GSN, its *majordomo* agent also moves to the new GSN carrying with it its data and (some of) the agents that are working for it. When deciding to occupy a locker, the user can specify whether he wants: (a) the locker to move following his movements, (b) the locker to remain stationary, or (c) the system to decide when to move the locker. When the user specifies that he wants the locker to follow his movement, there is a possibility that neither the corresponding GSN nor any GSN in the neighborhood can accommodate the locker. In this case, the system informs the user or his majordomo that the locker can not be moved.

The system decides on whether to move a locker or not based on the cost-benefit of the transmission. Prior to the computation of the cost of the transmission of the locker and its contents, the *Locker Place* must negotiate to find another *Locker Place*, located closer to the new GSN of the *majordomo*, which can accept the locker being transmitted. Briefly described, the cost of moving the locker (C_{Move}) consists of the cost of transferring all data stored in it (C_{Transf}) plus the cost of interacting with the *majordomo* in the second GSN ($C_{CloseInt}$).

The cost of not moving the locker is the cost of the interaction between the *majordomo* and the locker located at different GSNs ($C_{NotMove}$).

The following parameters are involved in the computation of $C_{CloseInt}$ and $C_{NotMove}$: (1) $T_{stay}(s)$: an estimation of the time the *majordomo* will stay in the new GSN. The *majordomo* can estimate this from the previous behavior of the user, or the user can explicitly provide such information. (2)$R_{Int}(Kb/s)$: an estimation of the rate of interaction between the *majordomo* and its locker that can also be induced from previous interactions. (3)$C_{FarCm}(s/Kb)$: the cost of communications between the *majordomo* in the new GSN and its locker in the current position. This can be calculated by the system. (4)$C_{NearCm}(s/Kb)$: similar to the previous cost but for the case of the locker being located in a position near to the new GSN.

To compute C_{Transf}, the following must be considered: (1) $T_{CrLAg}(s)$: the time to create a new *locker agent* in the new *Locker Place*, (2) $T_{Send}(s/Kb)$: the time to transmit to the new *Locker Place* the contents of size $S(Kb)$ stored in the locker, (3) $T_{UpdDB}(s)$: the time to update the DB of the new *Locker Place* with the N entries about the contents of the locker being transfered, (4) $T_{MoveLRAg}(s)$: the time to move the *locker rent agent* from the old *Locker Place* to the new one.

Without going into details and without considering the cost of interchanging few messages among the agents in order to coordinate the movement, as well as the deletion of what remains of the old locker, the two costs under consideration are:

$$C_{NotMove} = T_{stay} \times R_{Int} \times C_{FarCm} \ . \tag{1}$$

$$\begin{aligned} C_{Move} = C_{CloseInt} + C_{Transf} = (T_{stay} \times R_{Int} \times C_{NearCm}) + \\ +(T_{CrLAg} + S \times T_{Send} + N \times T_{UpdDB} + T_{MoveLRAg}) \ . \end{aligned} \tag{2}$$

The locker is transferred if $C_{NotMove}$ is greater than C_{Move}. In general, moving the locker from one GSN to another is an expensive operation. Nevertheless, the possibility of moving a locker is useful since the system can choose to move a number of lockers from one GSN to another to balance the system load, while taking care of not deteriorating the interaction between the lockers and their *majordomos* by overly incrementing the cost of their communications.

The shared lockers are associated with a group of users whose movement is in general independent. Their location is the result of a decision of the group or of the representative of the group, so shared lockers do not necessarily move following their users.

4 Implementation and Performance Results

A prototype implementation of the *Locker Rent Service* using the platform "Aglets Workbench" [8] is under development. Currently, our prototype allows the creation of private lockers, the storage of files in the lockers, authorized access to the files, and the release of the lockers. Also, we have fully implemented the interaction among all components involved in the service: the agents (*MU static agent*, *majordomo*, *locker guardian agent*, *locker rent agent* and *locker agent*), the places (*MU*, *Inventory* and *Locker*) and the database in the *Locker Place*.

We have performed a number of experiments to validate our approach. In these experiments, the mobile device was a portable computer running Windows 98, equipped with a 233 MHz Pentium processor and 64 MB of RAM. We used a 9,600 bps GSM card as the communication device. For the GSN, we used a stationary computer running Linux Red Hat 6.1, equipped with a 400 MHz Pentium II processor, and 128 M of RAM. We have performed several experiments[3], in which we compared the time that the MU needs to be connected in order to get, for example, a file or a web page, according to three different approaches.

The first approach corresponds to the traditional Client/Server(CS) model: the MU opens a connection directly to the address where the file is located and the connection remains open until the file is downloaded. The second approach corresponds to the Client/Agent/Server(CAS) model: the MU sends a message to its *majordomo* agent located in the GSN, specifying the address of the file to be downloaded, and delegating the task of obtaining the file or web page to the *majordomo*. Then, the MU can close the connection. Once the file is obtained, the *majordomo* sends it to the MU in one message. Finally, the third approach corresponds to the Client/Agent/Locker (CAL) model: this is similar to the CAS model, except that the *majordomo* after obtaining the web page or file, sends it to the user's locker, where it is stored until the MU reconnects and the user asks for the file or web page.

Notice that we are only interested in the MU connection time and not in the total cost. That is, for the CAS and CAL approaches, we only have to consider the time needed to interchange the messages between the MU and the majordomo in the GSN. Saving connection time also means that the MU saves battery power and money. The CAS and CAL approaches make it possible to work in an asynchronous way, i.e., the MU submits a task to be performed and then disconnects or continues working on other tasks, while the submitted task is being executed in the GSN. When the task finishes, the MU receives the results.

Our experiments showed that when downloading a single file, the CS approach performs slightly better than the approaches using agents. The time the MU needs to be connected is determined by the time it takes to interchange data between the MU and the GSN. With the CS approach, this time is mostly the time it takes to send the file through the wireless media. However, in the CAS and CAL approaches there are some messages interchanged between the agents situated in the MU and in the GSN. Even if these messages are short, their transmission through the GSM media at 9,600 bps causes the small overhead of those approaches over the CS one. However, the bigger the size of the file, the more irrelevant the overhead. Nevertheless, these results do not constitute a proof against the use of the CAS and CAL approaches. Taking into account that the cost that influences the most the final cost is the one associated with the

[3] Each experiment has been performed more than 20 times.

low speed communication media, the approaches using the intermediary agent give the opportunity to reduce the communication cost by applying various optimizations such as (a) applying filters in the GSN that reduce the amount of data to be transmitted; (b) avoiding interactions between the MU and the GSN, i.e. the need of using the wireless media; and (c) performing tasks in the GSN that otherwise should be done in the MU (with the higher use of low speed communication media that this implies).

Experiments with such optimizations are reported in [15]. In one experiment, we applied *lossless compression* to the files before transmitting them to the MU. The results showed that even a small compression percentage suffices to make the approaches with agents perform better than the CS approach. In another experiment, to avoid interactions between the MU and its outside world, we run a test in which web pages (the .html file along with the files of the images incrusted in the page) were downloaded to the MU. The results demonstrated that the avoidance of interactions needed to download a single web page with few images is enough to make the approaches using agents perform better than the CS one.

These experiments show the benefits that the use of the approaches using intermediary elements can bring to the mobile computing framework. However, the performance of the CAL approach is slightly worse than the behavior of the CAS approach. This is because the use of the locker in the CAL approach introduces a time overhead due to the extra message sent from the MU to the *majordomo* in the GSN, once the MU reconnects. However, the ability of storing data for the MU in a persistent and safe way constitutes the strong point of the CAL approach over the CAS one. Moreover, we present an experiment that shows how the small overhead between the CAS and CAL is compensated and even inverted, by using a feature provided by the locker to facilitate dealing with complex requests, by storing their results in the locker for a period of time, and downloading them to the MU later.

For this test we assumed that the user requests, within a single *request message*, 4 different web pages to be obtained and stored in the locker. Table 1 shows the characteristics of the pages used. Later on, he requests (one *read message*) to download them together to the MU (one *answer message*). We compared this scenario with the following one in which the user request each page, one by one (one *request* and one *answer message* per page), to be obtained and downloaded to the MU. Notice that this last scenario corresponds to the case in which the user has not contracted a locker. Therefore, the *majordomo* agent can only keep the web pages with it for a short period of time before sending them to the MU.

In Fig. 2 we observe how the avoidance of messages between the MU and the GSN makes the CAL approach better than the CAS one. The total amount of data transmitted, belonging to the web pages, is the same in both cases, the only difference between both approaches is the number of messages interchanged. The analysis of the standard deviation also shows how the bigger the number of messages interchanged the bigger the instability.

Table 1. Characteristics of the set of web pages used

Page	No. of images	size of each image	No. of files to download	Data to download
Page 1	1	80 Kb	2	80 Kb
Page 2	4	20 Kb	5	80 Kb
Page 3	8	10 Kb	9	80 Kb
Page 4	16	5 Kb	17	80 Kb

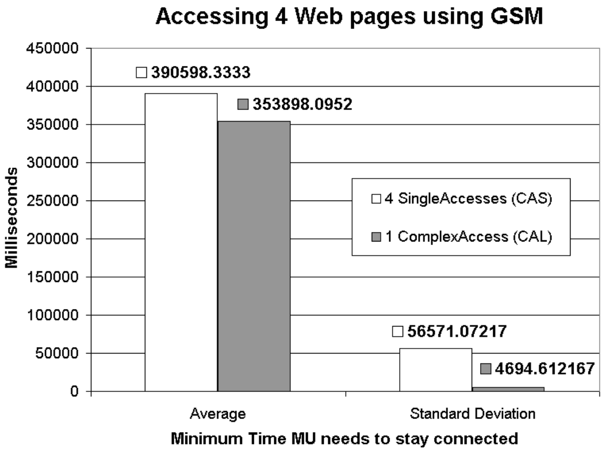

Fig. 2. Using GSM. Comparison of the minimum time the MU has to be connected in order to get 4 Web pages using the CAS and the CAL approaches

5 Conclusions

In this paper, we present a new data service for the mobile user: the *Locker Rent Service*. This service allows its users to use extra storage space located at the fixed network, outside of their mobile computer but close to them. Besides providing an extension of the storage space of the mobile computer, the service gives to its users more autonomy. Autonomy means that users can decide when to work disconnected or connected to the wireless network, since they can always count on the locker space for storing both the messages sent to them by other users and the results of their requests. Moreover, the use of lockers allows reducing data traffic in the wireless network, and provides an alternative for storing sensitive data.

For the implementation of the *Locker Rent Service* we use the agent technology. The induced overhead of using agents is greatly compensated by the advantages they provide with respect to performance, flexibility and adaptability. Our experiments show the feasibility of the implementation when compared to the traditional Client/Server approach. Furthermore, our experiments show how the performance of our approach is further improved when complex tasks are considered.

References

1. Milojicic et al. MASIF, the OMG Mobile Agent System Interoperability Facility. In *Proceedings of Mobile Agents '98*, September 1998.
2. Yair Amir, Baruch Awerbuch, and Ryan S. Borgstrom. A cost-benefit framework for online management of a metacomputing system. *The International Journal for Decision Support Systems, Elsevier Science. To appear.*
3. R. Gray, D. Rus, and D. Kotz. Agent TCL: Targeting de needs of mobile computers. *IEEE Internet Computing*, 1997.
4. A.D. Joseph, J.A. Tauber, and M.F. Kaashoek. Mobile computing with the Rover toolkit. *IEEE Transactions on Computers: Special Issue on Mobile Computing*, 46(3), March 1997.
5. A. Joshi, S. Weerawarana, and E. Houstis. On disconnected browsing of distributed information. In *IEEE RIDE97*, pages 101–107, 1997.
6. J.J. Kistler and M. Satyanarayanan. Disconnected operation in the Coda file system. *ACM Transactions on Computer Systems*, 10:3–25, 1992.
7. E. Kovacs, K. Röhrle, and M. Reich. Mobile agents OnTheMove – integrating an agent system into the mobile middleware. In *Acts Mobile Summit. Rhodos, Greece.*, June 1998.
8. IBM Aglets Workbench Home Page. http://www.trl.ibm.co.jp/aglets/.
9. S. Papastavrou, G. Samaras, and E. Pitoura. Mobile agents for WWW distributed database access. In *Proceedings of the International Conference on Data Engineering*, 1999.
10. E. Pitoura and G. Samaras. *Data Management for Mobile Computing*. Kluwer Academic Publishers, 1998.
11. G. Samaras and A. Pitsillides. Client/Intercept: a computational model for wireless environments. In *Proceedings of the 4th International Conference on Telecommunications (ICT'97)*, April 1997.
12. Bruce Schneier. *Applied Cryptography. Second Edition.* John Wiley & Sons, 1996.
13. Christian Tschudin. Mobile agent security. In *Intelligent Information Agents. Forthcoming LNCS*. M. Klusch.
14. Y. Villate, D. Gil, A. Goñi, and A. Illarramendi. Mobile agents for providing mobile computers with data services. In *Proceedings of the Ninth IFIP/IEEE International Workshop on Distributed Systems: Operations and Management (DSOM 98)*, 1998.
15. Yolanda Villate, Evaggelia Pitoura, Arantza Illarramendi, and Ahmed K. Elmagarmid. Extending the data services of mobile computers by external data lockers. In *Proceedings of the Third International Workshop on Mobility in Databases and Distributed Systems, MDDS'2000.* IEEE Computer Society Press, September 2000.
16. X:drive. http://www.xdrive.com/.