# ADAPTIVE FILE TRANSFER MIDDLEWARE FOR MOBILE APPLICATIONS

Mario A. Gomez-Rodriguez, Victor J. Sosa-Sosa and Ivan Lopez-Arevalo
*Laboratory of Information Technology (LTI), Cinvestav-Tamaulipas*
*Parque Científico y Tecnológico TECNOTAM - Km. 5.5 carretera Cd. Victoria-Soto La Marina*
*C.P. 87130. Cd. Victoria, Tamaulipas, Mexico*

Keywords: Wi-Fi, GPRS, MMS, Mobile File Server.

Abstract: Current mobile devices such as mobile phones and PDAs are able to run applications that can demand a considerable storage space. When these devices run out of local memory, they require backing up their files in an external storage device, which could restrict the user mobility. This paper presents an Adaptive File Transfer Middleware (AFTM) for mobile applications. This middleware eases the transfer of files between a mobile device and an external storage server by accessing the best wireless connection (WiFi, GPRS/UMTS) available, considering quality and cost of the service. AFTM is also able to use the Multimedia Messaging Service (MMS) as another option for transferring files. A File Backup Service (FBS) was built on top of the AFTM. The FBS will detect when the device ran out of local memory and will automatically send selected files from the mobile device to an external storage server, freeing the mobile storage memory. To decide which files should be backed up, FBS implements several file replacement policies. Results showed that the selection of one replacement policy will be a trade-off between the efficiency of the algorithm and the cost of the wireless service available when a file needs to be backed up.

## 1 INTRODUCTION

The cell phone is one of the most popular mobile devices. It has constantly evolved since its invention. It's common that current mobile devices have two or more communication interfaces such as USB, infrared, Bluetooth, GPRS and the IEEE 802.11, popularly known as Wi-Fi. Wireless networks improve the utility of portable computing devices, enabling mobile user's versatile communication and continuous access to services and resources of the terrestrial networks. Mobile phones have also been endowed with more computing power. Intelligent mobile phones (a.k.a. smartphones) integrate the functions of PDA (Personal Digital Assistant) in conventional mobile phones. These technologic developments allow current mobile phones to run applications that generate a large number of files and, as a consequence, require a greater storage capacity. Unfortunately, the storage capacity on these devices has not grown at the pace that mobile applications require. Once a mobile device exceeds its storage capacity, it is necessary to download its files (such as text, images, music, videos, etc.) in an external storage system (e.g., a computer), backing up the information. This process usually limits the mobility because users have to connect their mobile devices to a fixed external storage using a cable (e.g., USB), or through a short-range wireless network such as Infrared or Bluetooth. These storage limitations reduce the mobility and storage capacity of users, particularly in situations when users are travelling and they need more space, for instance, to keep their pictures or videos and there is not available a nearby external storage device. These situations motivate the development of supporting tools that allow files to be exchanged between mobile devices and external storage systems in a transparent way, taking into account LAN or WAN wireless connectivity, cost of the service and available storage. Since there are different mobile devices that work with various computing platforms or operating systems (e.g., Symbian, Blackberry, Windows Mobile, Android, etc.), it is not feasible to develop a proprietary tool for a specific platform, because it would limit its usability. That is why the design of a solution to this problem must include



A. Gomez-Rodriguez M., Sosa-Sosa V. and Lopez-Arevalo I. (2010).
ADAPTIVE FILE TRANSFER MIDDLEWARE FOR MOBILE APPLICATIONS .
In *Proceedings of the International Conference on Data Communication Networking and Optical Communication Systems*, pages 45-52
DOI: 10.5220/0002990100450052

multiplatform considerations that allow better coverage in different mobile devices.

The situations described above were our motivation to build an Adaptive File Transfer Middlewar (AFTM) that can be capable of running on different operating systems and allow mobile applications to send and receive files from an external storage server, connected to Internet through different wireless technologies, taking into account the cost and quality of the service. The AFTM was tested by developing a File Backup Service (FBS) for mobile phones, which offers a service of swapping files between the mobile device and an external storage server in a transparent way. Users of this application perceive a virtual storage space, which is higher than the real memory space included in the mobile device. FBS is similar to a file caching service, reason why it integrates an efficient replacing policy to optimize the file access time.

The rest of the paper is structured as follows. Section II describes related work; some of the systems described in Section II gave technical support to the Adaptive File Transfer Middleware (AFTM). Section III presents the AFTM architecture, which is divided in three main parts: Client-side application layer, Core connection layer and Server-side application layer. In Section IV, a mobile application named File Backup Service (FBS) is presented. This application was developed as a use case for testing the AFTM. Section V includes final comments and mentions ongoing work.

## 2 RELATED WORK

Mobile file systems like Coda (Satyanarayanan et al., 1990)(Kistler et al., 1992), Odyssey (Satyanarayanan, 1996), Bayou (Demers, 1994) and Xmiddle (Mascolo et al., 2002), worked with the data sharing-oriented problem in distributed computing environments. This problem could be directly related to the file transfer problem in mobile phones. Although with different focus, all of these systems try to maximise the availability of the data using data replication, each one differing in the way that they maintain consistency in the replicas. Coda provides server replications and disconnected operations; it allows access of data during the disconnection period and focuses on long-term disconnections, which more often occurs in mobile computing environments. Odyssey is the successor of Coda, which has been improved introducing context-awareness and application-dependent behaviors that allow the use of these approaches in mobile computing settings. The Bayou system is a platform to build collaborative applications, its emphasis is on supporting application-specific conflict detection and resolution. It has been designed as a system to support data sharing among mobile users and is intended to run in mobile computing environments. The system use a read-any/write-any replication scheme, thus the replicated data are only weakly consistent. Unlike previous systems, Bayou exploits application knowledge for dependency checks and merges procedures. (Lui et al, 1998) propose a mobile file system, NFS/M, based on the NFS 2.0 and the Linux platform. It supports client-side data caching in order to improve the system performance, reducing the latency during weak connection periods. (Atkin et al., 2006) propose other file system that, like NFS/M, supports client-side caching. Some applications like (GSpaceMobile, 2009) and (Emoze, 2009), enable the file transfer between mobile devices and external storage servers. However, these applications only consider a proprietary storage server.

(Boulkenafed and Issarny, 2003) 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. These middleware enable to share and manipulate common data in a collaborative manner (e.g, working meet, network gaming, etc.) without the need for any established infrastructure. They implemented their middleware service within a file system in order to evaluate it. The result was a distributed file system for mobile ad hoc data sharing. It is worth mentioning that the performance measurements were done on a platform of ten laptops, and they only use IEEE 802.11b WLAN in ad hoc mode, unlike AFTM, which is able to use Wi-Fi, GSM, GPRS or UMTS networks. (Belimpasakis et al, 2008) propose a content sharing middleware for mobile devices using different protocols (UPnP, Atom and WebDAV), providing interfaces for applications, in order to allow 3rd party developers to create applications with sharing capabilities. The middlewares mentioned above make use of both Wi-Fi and GPRS wireless networks. However, they consider neither transferring files through a messaging system like MMS nor the portability issue. We have decided to develop AFTM using J2ME, offering portability.

# 3 SERVICE ARCHITECTURE

In this section, the AFTM architecture is described in more detail. The architecture is mainly divided in three components or layers: Client-side application layer, Core connection layer and server-side application layer. Figure 1 depicts this architecture whose components are presented in the following sub-sections.

## 3.1 Client-side Application Layer (API-ESS)

This layer deals with file transferring operations (send/receive) required from mobile applications. One main function is to make transparent the selection of the available wireless connection (WiFi, GPRS or UMTS) or messaging service (MMS) provided by the lower layer (Core) to the mobile application. The main component of the client-side application layer is the External Storage Services Module (API_ESS). This module offers a set of wrappers for connections with different wireless networks. It uses a connection selector that indicates which wireless network will be used. When connectivity is limited, the connection selector offers the MMS services as the best option for file transferring.
The API ESS includes both messaging services SMS and MMS. Due to the limitations of SMS (about 150 bytes by message), MMS is the default option chosen by the connection selector.

The API_ESS could include a configuration file that indicates the priorities assigned to the wireless connection services. The first in the list indicates the cheapest service in terms of cost or bandwidth consumption. The API_ESS functions are divided into upper level functions and lower level functions. The upper functions are typically used by mobile applications, e.g., selectRoot( ), retr(file) and stor(file). These functions send and receive files from an external storage service making totally transparent the type of wireless o messaging service used. Lower level functions deal directly with wireless connection and messaging wrappers. Examples of these functions are: autoDetectConnection(), getConnection() and setConnection.

The current implementation of the API_ESS includes the following wrappers:

- *Wi-Fi/GPRS*: It enables to send/request files via the IEEE 802.11 and the GPRS protocols.

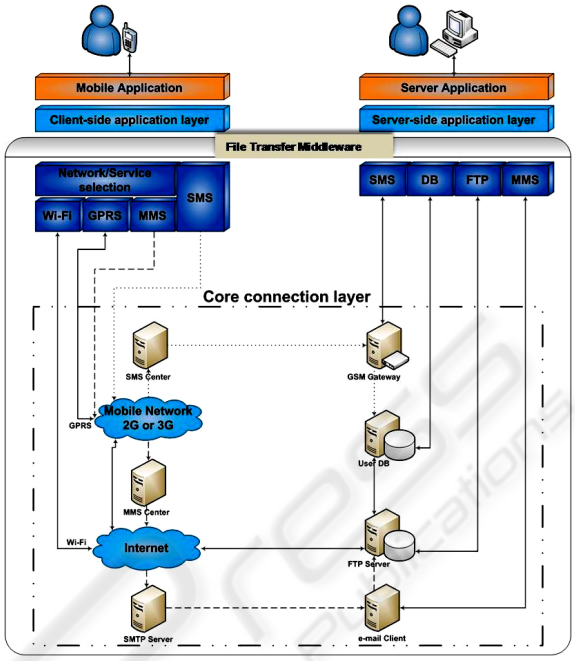


Figure 1: Layers of the File Transfer Middleware (AFTM) for Mobile Phones with Limited Connectivity.

- *MMS*: It represents an alternative option for sending and requesting files using the Multimedia Messaging Service (MMS). MMS has to be supported by a mobile phone carrier; otherwise it will not be available. This wrapper will commonly be used when the Wi-Fi and GPRS (or UMTS) networks are not available.
- *SMS-M*: The purpose of this wrapper is to provide mobile applications with one more option for sending information to and/or receiving from the external storage server in future applications. The mobile application FBS uses this wrapper for registering user accounts in a web storage server that is supported by the server-side layer of the AFTM.

## 3.2 Core Connection Layer

This part of the architecture represents the communication services needed by our middleware. It includes modules to deal with the most popular wireless technologies and messaging services. A brief description of these modules is as follows:

- *Center (SMSC)*: The SMSC is responsible for relaying messages between short message entities (SMEs, elements that can send or receive

text messages) and store and forward messages, if the recipient SME is not available.

- *MMS Center (MMSC)*: The MMSC is a key element in the MMS architecture, and it is composed of an MMS relay and an MMS server, which are responsible both to store and manage incoming and outgoing multimedia messages. It also ensures interoperability with other messaging systems (Bodick, 2005) by means of different communication interfaces (e.g., MM3 interface).
- *SMTP Server*: It receives the files that have been sent via multimedia messaging (MMS) to an email account. The data travels across the wireless network available (2.5G or 3G) and are routed by the MMSC to the Internet, then get to our SMTP server. This option is activated when socket connections fail.
- *GSM Gateway*: It receives text messages (SMS) that are sent by the client for complementary services, e.g., to create an account.
- User DB: It is an optional database that could contain information of subscribers registered by mobile applications based on AFTM.
- *FTP Server*: It is the process responsible for receiving and storing the files. In the FBS application, users have to be registered before obtaining a storage space in the external storage server. Files managed by the FTP Server can come from direct socket connected clients (using a wireless connection) or e-mail clients (using a MMS connection). This server is one of the key parts of the architecture as it controls all the files received in the external storage server.
- *e-mail Client*: This process is used by the server-side layer when the original server-side MMS receiver fails. In this situation, files that are sent by clients using MMS are redirected to a mail box defined in the server-side application layer. The e-mail Client process is in charge of obtaining these files from the indicated mail box using the Post Office Protocol (POP3).

### 3.3 Server-side Application Layer (API-ISS)

This layer gives developers an infrastructure (API_ISS) for building a storage server that considers mobile devices. API_ISS includes modules for receiving and sending files through different communication services such as WiFi, GPRS, UMTS and the Multimedia Message Service (MMS). Different types of servers can be implemented using this API. The particular behavior can be customized, for instance, by developing a web storage service or a file sharing storage server. The Internal Storage Service (ISS) module represents the main module included in the API_ISS.

The API_ISS is a module that offers a set of methods, which will be used by server-side applications. It contains functions for managing connections and user accounts as well as functions for receiving and transmitting data through different wireless networks. It includes similar wrappers like those located in the client side application layer. It implements the FTP service, which packs all the methods for transferring and receiving files going to or coming from mobile devices. A file storage server that is developed with the API_ISS could connect with other distributed storage servers building a big virtual disk. This distributed approach allows it to increase its virtual available storage space by integrating the storage space offered by other external storage.

## 4 USE CASE

This section describes a use case for the Adaptive File Transfer Middleware (AFTM). A mobile application named FBS that uses our AFTM was developed. The purpose of FBS is to send files from a mobile phone to an external storage server when the mobile phone runs out of memory. This process is transparent for users that can always see the complete list of files, even when some of them are not physically kept in the memory of the mobile phone. When a file is required by the user, FBS requests the missing file from the storage server using any wireless network available or the MMS service, depending on the service priority defined in the configuration file. Automatic swapping allows mobile devices to free storage space in a transparent way. A web storage server was also developed based on AFTM. To use our web storage server, users have to be registered before sending files. The registration process can be done through a SMS message or by filling in a web form. FBS and the AFTM were developed using the J2ME platform. The configuration, profile and basic libraries needed for our development are: CLDC 1.1 (Connected Limited Device Configuration 1.1.), MIDP2.0 (Mobile Information Device Profile 2.0), WMAPI 2.0 (Wireless Messaging API 2.0), SATSA-Crypto (JSR 177) and PDA Profile (JSR 75).

Due to the fact that FBS works like a caching system, a file replacement police was included in its implementation. A benchmark based on metrics

taken from different real mobile phones was also created. The main metrics were: the average storage space in a mobile phone, the average file size, and the average data transfer rate in wireless LAN and WAN networks such as WiFi y GPRS. Table 1 includes measures taken from different mobile phones and PDAs such as: Nokia5530, 7020, N95 and HP iPAQ HW6945, Blackberry 8220.

Table1: Common measures in limited mobile phones and PDAs.

| MEASURE | SIZE |
|---|---|
| MAX_MEM_SIZE | 2GB |
| MAX_FILE_SIZE | 15MB |
| MIN_FILE_SIZE | 469KB |
| AVG_FILE_SIZE | 1.3MB |

A collection of more than 1000 files taken from different mobile phones revealed an average file size of 1.3MB. More of them were of music, photos and videos. Transmitting files of 1.3MB from different mobile phones to an external storage server connected to Internet using WiFi revealed an average time of 8.3s (about 1.3Mb/s). Using GPRS for the same transmissions showed an average time of 148s (about 73Kb/s).

These experiments gave us a realistic average transmission time of wireless networks with different traffic loads. The traffic coming from Internet connections played an important role in these measures. (Enriquez, 2009) made several tests trying to find an optimal block size for transmitting files between a mobile phone and a data server (PC) using WiFi and GPRS networks, considering different traffic hours. The optimal block size obtained using WiFi was 4KB (Figure 2) and for GPRS was 96B (Figure 3). However, the GPRS transfer rate showed a high variability in hours with high traffic, resulting in many communications errors. Several block sizes were tested with high traffic to observe the GPRS behavior. GPRS showed less variability when a block size of 64B was used, resulting in a better behavior. This stable behavior motivated the use of a block size of 64B in AFTM, instead of 96B found in (Enriquez, 2009).

As we mentioned above, FBS is a mobile application that works like a caching service, reason why it includes a file replacement policy. The main task of a replacement police is to decide which file will be sent to the external storage server to free storage space in the mobile phone memory.

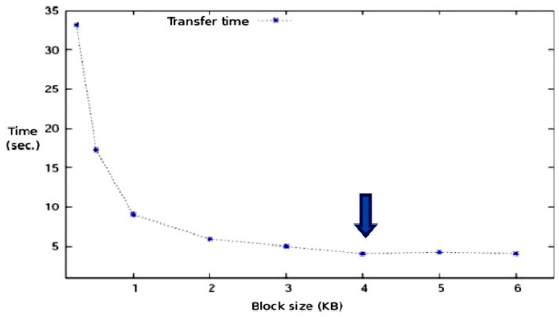


Figure 2: An optimal transmission block size in a WLAN network.

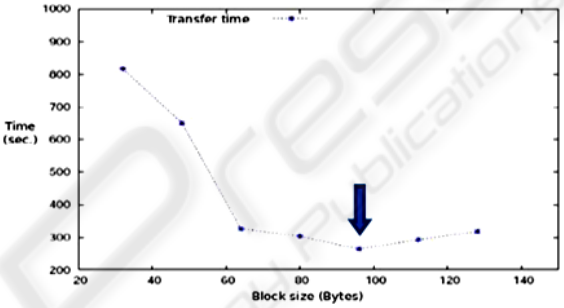


Figure 3: An optimal transmission block size for a GPRS network.

In order to find a good replacement policy for FBS, four algorithms were tested. These algorithms were: LRU (Least Recently Used), LFU (Least Frequently Used), LFS (Largest File Size), and SFS (Smallest File Size). As their names indicate, the LRU policy will send the least recently used file to the external storage server. LFU will send the least frequently used file, LFS the largest file, and SFS the smallest one. If there is more than one file as the largest or the smallest, a LRU police is applied. A benchmark that reproduces a sequence of send/receive operations, using a group of 120 files with sizes ranging from 512KB to 2.5MB was implemented. Thirty one experiments were carried out with different scenarios. Connections for sending and receiving files were randomly changing between GPRS and WiFi, emulating mobile users.

Figure 4 depicts the behavior of the average hit ratio in FBS using each replacement policy (in 31 experiments). Figure 5 shows the average transmission time.

As we can see, if FBS replaces files based on the largest file size (LFS), it obtains the best hit ratio (almost 8 of 10 files were in the mobile phone memory when they were requested by the user).

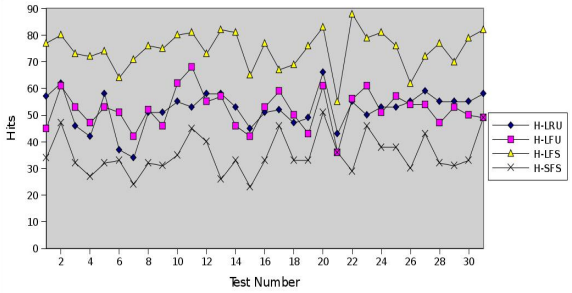


Figure 4: Average hit ratio obtained by FBS using different replacement polices.

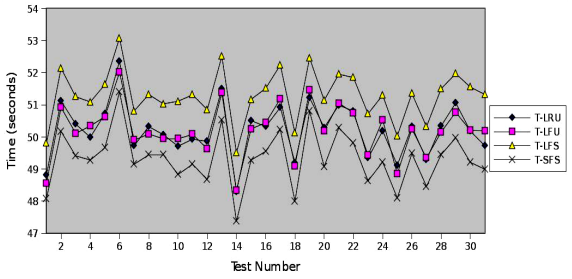


Figure 5: Average transmission time obtained by FBS using diferrent replacement policies.

However, it also obtains the greatest average transmission time, which could be a result of transmitting large files using a GPRS connection at the moment of a miss. These results give us some insights to determine if we should prioritize between the quantity of bytes transmitted and the total time consumption using the network, especially in the GPRS/GSM networks. In most of the cases the wireless communication service is charged based on bandwidth consumption. Figure 6 shows the total bandwidth consumption after running all of the algorithms. These experiments did not show a correlation between the total bandwidth consumption and the average transmission time. Even though the LFU algorithm had low total bandwidth consumption, it did not show low average transmission time. It happened so because most of the time that the mobile application had to send/receive files (using the LRU algorithm) coincided with a GPRS connection.

The evaluations using FBS gave us information for deciding which algorithm could be implemented as part of our Adaptive File Transfer Middleware (AFTM), and to consider if it would be better to include a fixed replacement policy or an adaptive one, considering if the cost of the communication service is based on time or bandwidth consumption. In the current implementation of FBS, users are able to define a cost-based priority list that best fits his or her requirements. This priority list has to be included in a configuration file. Information like depicted in Figure 4, 5 and 6 can help to decide the order of priorities based on the cost of the services in a particular region.

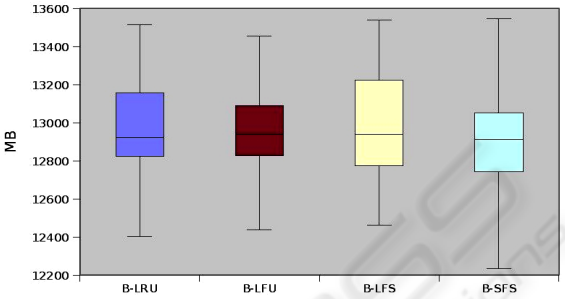


Figure 6: Total bandwidth consumption obtained by FBS using 4 different replacement policies.

Another important topic considered in FBS was information privacy. This topic becomes an important factor when transmitting files from a mobile phone to an external and public storage server. In cases where the storage server does not offer a private place to keep the user information, FBS can encrypt the files before sending them to the server. In these cases, the external storage server will keep only encrypted files, offering privacy to the information. The encryption process is carried out by using the Java Specification Request 177 (JSR 177, Satsa Crypto). FBS uses the Advanced Encryption Standard (AES), which is a symmetric key algorithm with a key and block size (it could vary) of 128bits. Experiments for evaluating the impact in the transferring time when using encryption were conducted with FBS. The objective was to find out the impact in terms of the cost of extra bandwidth and transferring time needed when encryption is applied. For this experiment, the use of the MMS service as another option (included in AFTM) to transfer files was tested. To execute this experiment, our test bed had to be redesigned, because the largest public MMS service provider in Mexico has defined a maximum file size for each MMS message of 100KB. FBS was tested using a group of 100 files with sizes ranging from 80 to 100KB.

Table 2 shows how the encryption process increases the average file transfer time until 3 times in some cases. We can see how WiFi connections are the most affected.

Table2: Average file transfer time with and without encryption using files with an average size of 90kb.

| Average File Transfer Time (AFTT) | **WiFi** | **GPRS** | **MMS** |
|---|---|---|---|
| AFTT without Encryption | 0.07s | 1.23s | 32.56s |
| AFTT with Encryption | 0.21s | 1.96s | 86.12s |
| Impact Using Encryption | 3 times more | 1.59 times more | 2.64 times more |

This situation is more evident when small files are transmitted. Since GPRS networks have a low bandwidth and low transmission rates, the resulting impact of the encryption process is lower. The transmission times obtained from the MMS service showed a high variability, because the public MMS server provider in Mexico does not guarantee real time transmissions.

# 5 FINAL COMMENTS

As mobile devices evolve, they require more storage capacity to keep the large amount of files that new mobile applications generate. This paper briefly introduced an Adaptive File Transfer Middleware (AFTM) for mobile applications. The middleware allows mobile applications to store and request files from an external storage server, taking advantage of available wireless networks and considering issues related to the cost of the service. As a use case, a mobile application that transparently swaps files between a mobile device and an external storage server was developed, increasing storage space in the former. In this application, different file replacement policies were tested. An important parameter that has to be considered when deciding about a replacement policy is the cost of the service. Results obtained in this work showed that it is not a trivial decision to select a specific replacement policy for a mobile device. Mobile devices do not present a correlation between the bandwidth and time consumption because they do not keep the same wireless connection any time. Service costs are usually defined by wireless WAN providers, and could be based on bandwidth consumption or time consumption, criteria that have to be included in the replacement algorithms. FTS is also prepared for transferring files using the Multimedia Messaging Service (MMS), an additional option to be considered for users when connections are limited. Our experiments revealed that the time for transferring encrypted files could rise the original time in a factor of three. This increase is more evident when small files are transmitted using wireless network with high transmission rates.

# ACKNOWLEDGEMENTS

This research was partially funded by project number 51623 from Mix Funds of the National Council for Science and Technology (CONACYT-Mexico) and the Government of Tamaulipas State.

# REFERENCES

Satyanarayanan, M., Kistler, J. J., Kumar, Okasaki, P., E. H. Siegel,E. H., and Steere.D. C. (1990) Coda: a highly available file system for a distributed workstation environment. *IEEE Transactions on Computers* 39(4):447-459.

Kistler J. J. and Satyanarayanan M.. (1992) Disconnected operation in the Coda file system. *ACM Transactions on Computer Systems* lO(1): 3-25.

Satyanarayanan, M. Mobile Information Access. (1996) IEEE Personal Communications, 3(1):26–33.

Demers, A., Petersen, K., Spreitzer, M., Terry, D., Theimer, M., Welch, B. (1994) The bayou architecture: Support for data sharing among mobile users.

Mascolo, C., Capra, L., Zachariadis, S., Emmerich, W.: Xmiddle. (2002) A data-sharing middleware for mobile computing. *Int. Journal on Personal and Wireless Communications* 21(1). 77-103.

Lui, J. C. S., So, O. K. Y., Tam, T. S. (1998) NFS/M: An open platform mobile file system. In: ICDCS '98: Proceedings of the The 18th International Conference on Distributed Computing Systems, Washington, DC, USA, *IEEE Computer Society* 488.

Atkin, B., Birman, K. P. (2006) Network-aware adaptation techniques for mobile file systems. In: NCA '06: Proceedings of the *Fifth IEEE International Symposium on Network Computing and Applications, Washington*, DC, USA, IEEE Computer Society. 181-188.

GSpaceMobile: (2009) Available at: https:// www.ibomobi.com/home/.

Emoze™: (2009) Available at: http://www.emoze.com/.

Boulkenafed, M., Issarny, V. (2003) A middleware service for mobile ad hoc data sharing, enhancing data availability. In: Middleware '03: Proceedings of the ACM/IFIP/USENIX 2003 International Conference on Middleware, New York, NY, USA, Springer-Verlag New York, Inc. 493-511.

Belimpasakis, P., Luoma, J. P., Börzsei, M. (2008) Content sharing middleware for mobile devices. In:

MOBILWARE '08: *Proceedings of the 1st international conference on MOBILe Wireless MiddleWARE*, Operating Systems, and Applications, ICST, Brussels, Belgium, Belgium, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering). 1-8.

Bodic, G. L. (2005) Mobile Messaging Technologies and Services SMS, EMS and MMS. Second edn.

Enríquez, J.A.V. (2009) A vertical handover platform for applications on mobile devices. Master's thesis, CINVESTAV. Available at: http://www.tamps.cinvestav.mx/sites/default/_les/tesis 1.zip