

# Using Remote Memory Paging for Handheld Devices in a Pervasive Computing Environment

Arjuna Sathiaseelan and Tomasz Radzik

Department of Computer Science, King's College London, Strand, London WC2R 2LS

**Abstract.** Pervasive computing is the technology for the future. With efficient smart spaces and the proliferation of mobile devices and wireless networking standards, there are a lot of interesting potential solutions to many of the current problems. Due to the nomadic life of users, devices that are portable and handheld are highly useful in a pervasive computing environment. But there are severe problems that seem to be an obstacle for the usage of handheld devices in a pervasive environment. The major problem faced by hand held devices are the lack of memory space to run bigger application processes. In this paper, we define this problem and propose a possible solution for it.

## 1 Introduction

Pervasive Computing is maturing from mere fiction to fact. Pervasive computing encompasses the fundamental areas of mobile computing and distributed systems. In a pervasive environment, we have the notion of a smart space. Smart space is a combination of building infrastructure with technology embedded into it. These can range from sensors to wireless access points that connect to powerful remote servers. These smart spaces can range from stations, airport cabins, offices etc. We can also have personal smart space that accompanies a person everywhere. These personal smart spaces can be implemented on a body worn or hand held computer [1].

In recent years, there has been a proliferation of wireless technologies like IEEE 802.11b which supports 11Mbps transfer rate and the new arrival IEEE 802.11a which would support a transfer rate of 54Mbps and the proposed IEEE 802.15.3 which is a wireless PAN supports up to 54Mbps transfer rate and consumes less battery power. Also workstations and personal computers have become extremely powerful and have seen impressive increases in storage capacities. The cost effectiveness of these systems combined with the emerging high-speed wireless network technologies have led to the development of high-performance data processing environments based on networks of workstations connected to wireless networks by access points.

But the major obstacle for the use of handheld devices such as personal digital assistant, 2 way pagers, mobile phones etc in a pervasive environment is that the handheld devices have very little memory. The RAM is used for storing programs and also for executing them. Unlike desktop machines they do not

have a typical hard disk to store their application files. Thus the RAM also acts as the virtual memory. Thus there is a severe limitation in their usage.

*Cyber foraging* is a term commonly used in a pervasive computing environment. Cyber foraging means coupling the resources of wireless devices with wired hardware infrastructure [1]. In this paper, we propose a model for cyber foraging using these smart spaces available in a pervasive computing environment coupled with the latest wireless technologies for providing distributed memory management between devices with less memory and devices that have enough and unused memory. Section 2 presents a case study of the memory architecture of the two main PDA OS available in the market. Section 3 presents the Aura architecture. Section 4 presents our model. Section 5 presents the design issues involved in implementing the system. Section 6 gives the related work and finally Section 7 gives the conclusion.

## 2 Memory Architecture of Personal Digital Assistant

Mobile devices such as Personal Digital Assistants (PDA), 2 way pagers, mobile phones have very little memory. The available RAM is used for both storage and program execution. The following are the memory architectures of the two main PDA OS available namely Windows CE [9] and Palm OS [13].

### 2.1 Windows CE Memory architecture

Windows CE has a small memory footprint. The RAM in a Windows CE-based system is split up into two functional areas: one area is used for storing the objects and the other is used for executing the programs. Unlike other RAMs which are not persistent, the RAM on which all the objects are stored are persistent i.e. the files are always retained even when the system is suspended. That is the reason a backup battery is provided for Windows CE systems so that when we replace the main batteries, the back up battery provided by the system helps to retain the files stored in the object store. The next half of the RAM is used for program execution. This is where the application's heaps and stacks run. The boundary between the object store and the program RAM can be moved by the user by using the system control panel. When the memory conditions are low the system requests the user to use some part of the object store's RAM to be used for program execution.

The entire operating system is stored in the ROM. Also the applications that come with the system are also stored in the ROM. These applications are run directly from the ROM rather than being loaded in the RAM for execution. Thus the programs that run in ROM do not take the space in RAM and thus, the speed of execution of these programs are increased.

**Executing a Program** Each program is stored in its own RAM slot. There are about 33 RAM slots in the Windows CE memory architecture. The current running program is always stored in the first slot. These 33 slots are all composed

of virtual memory. Also the Pocket PC does not have more than 10 applications running at any given time.

**Virtual Memory** The Virtual memory management systems for Windows CE is similar to the paged virtual memory management implemented by the Win 32-based operating systems like Windows NT, Windows 98 etc. A page in a Windows CE can be either 1,024 or 4,096 bytes depending on the microprocessor. These physical pages can be in any one of the three states namely free, reserved, and committed. A free page is free and is available to be allocated. A reserved page is a page that has been set aside so that it cannot be allocated by the operating system or by any other thread in the process. A committed page is a page that has been reserved by an application process.

## 2.2 Palm OS architecture

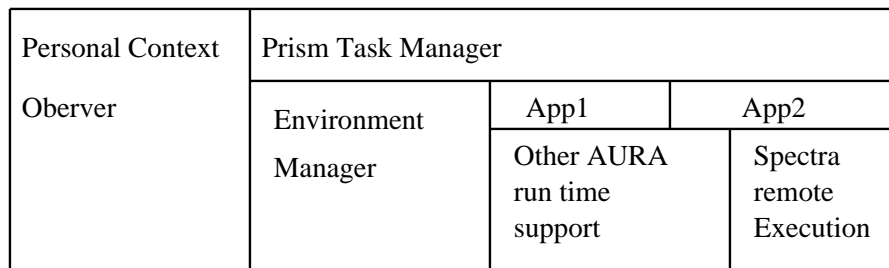
The Palm OS memory is part of the memory cards. These cards are removable. Both RAM and ROM are part of a single card. The OS supports theoretically up to 256 cards.

The ROM contains the operating system, the built in applications and the default databases. ROM sizes vary from 512K to 2M.

The RAM is used to store the add-on applications, user data and the run time storage. Their size varies from 128K to 2M. Palm OS can support up to 12M of RAM on a single memory card.

## 3 Aura Approach

The Aura is an architectural framework for ubiquitous environments developed by the Aura group at Carnegie Mellon University [12]. The Aura Client can be implemented into a body worn or hand held computer [1].



**Figure 2: AURA Architecture**

*Figure 2* shows some of the components of the Aura architecture. The personal Context Observer is used to interpret the physical context of the user, and is responsible for identifying a user identification, user's locations etc. The

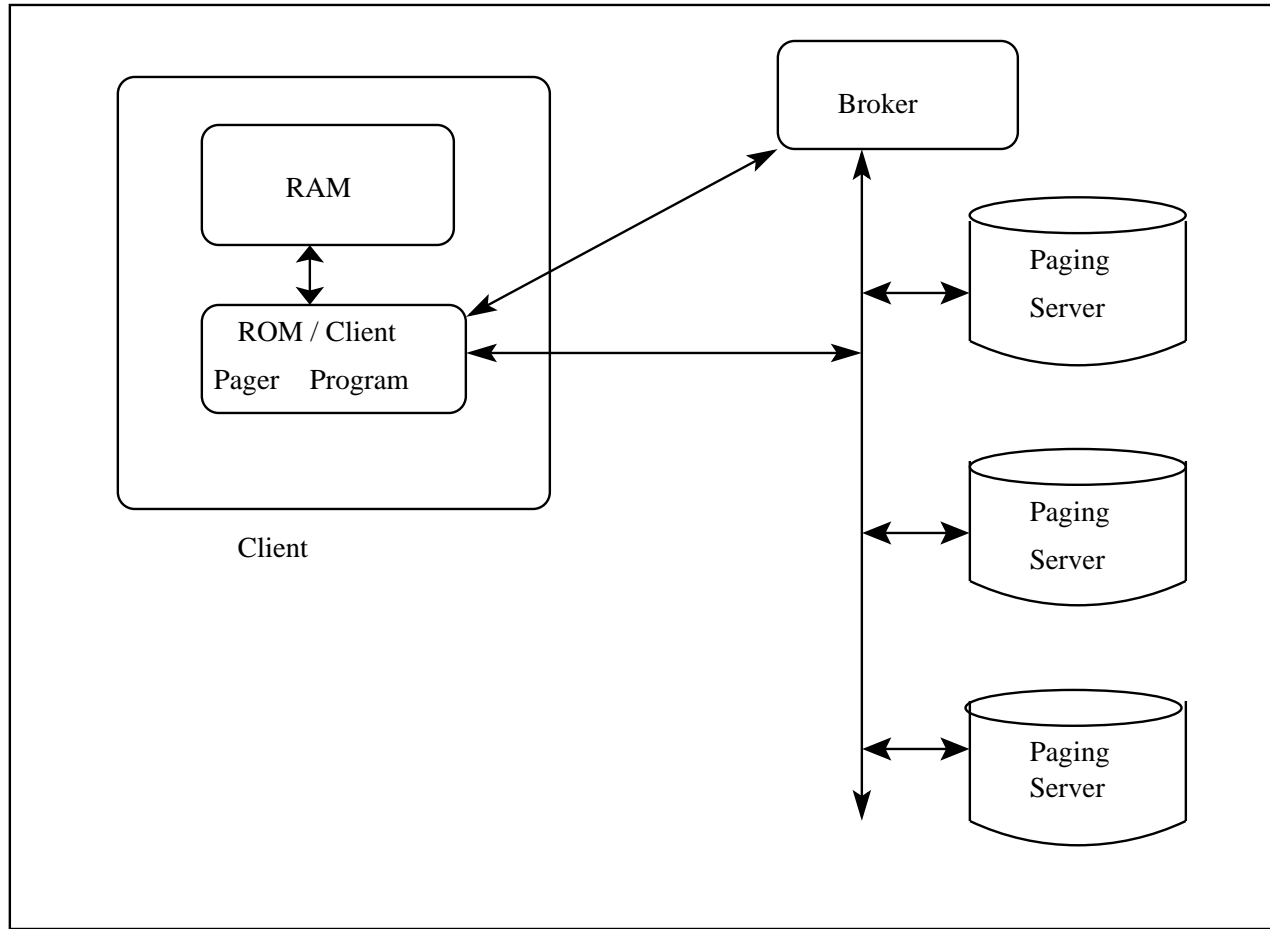
Task manager takes care of the context changes, change of environments, task changes etc. Finally, the Environment manager manages the computational environment by having knowledge of the computational environment, and assembling components to complete a task. The Environment manager also recognizes the resources available in an environment and utilizing those resources when needed.

## 4 Our proposed Solution

Our proposed model to solve the problem of insufficient memory in handheld devices to run bigger application process would be to use the concept of remote memory paging described in [2, 3, 4, 5] within a pervasive environment. This is done by

- Using the unused memory of the remote workstations, personal computers etc in the smart space to store the unused processes/pages of the handheld devices like PDAs over a wireless network through access points provided in the smart space.
- Storing the program files (like small word documents, excel documents etc) from the handheld device to the remote server's hard disk i.e. the program files that are stored in the PDA RAM are transferred to the remote server's hard disk (optional).

Suppose the user enters a room, which is a smart space with his hand held device and the user switches the PDA device. If the user starts running a bigger application over the Internet (like the latest.NET applications), the memory manager of the PDA tries to find out whether there is enough space to run the application process. If there is no enough space, the memory manger contacts the client pager program which immediately invokes the service discovery program that finds out whether there is any remote paging service available. If the device senses that the remote service is available, it asks the user, whether he wants to use this remote service. When the user says yes, then the device gets ready to form a wireless connection through the access points to the available workstations in the network. These machines are called *surrogate* machines. Wireless connections are established between the handheld device and these devices that are ready to allocate the required memory resource. Security measures are undertaken. When a wireless connection is established between the remote server and the local PDA, according to the memory needs, unused or idle pages in the RAM are sent to the remote server for temporary storage. If more memory is required even after transferring the idle processes then the program files (these are the files that have been created and stored in the RAM, similar to the files stored in a hard disk for desktop machines) are sent to the remote server where they are stored in the hard disk of the remote server. This is just an optional case left at the discretion of the user. Now, the RAM has got enough space to run processes with larger memory space. As soon as the user finishes executing the application process, the stored pages and files can be retrieved back in to the client machine.



**Figure 1: Client Pager, Broker and Paging Server**

Our proposed system will run 3 services analogously as proposed in [3],

**Client Pager Service** The client pager service is responsible for establishing connections and to transfer the pages from the local RAM to the remote RAM. This forwards the paging requests to a remote server using sockets over the wireless LAN over TCP/IP. This client program is stored in the ROM. Thus the program need not be loaded into the RAM. The client pager maintains a data structure called memory map table. This table consists of the list of servers, which are currently serving the client, with the details of the remote partitions and the current allocation of the pages. The client pager service has also the facility to transfer program files to the remote server.

**Remote Paging Server** This service is run on the remote servers. These servers are used to accept connections by client machines and are used to store the client's pages in its own memory. When the client wants to send a page the server reads the page through the designated socket. When the client requests back the page the server transfers the requested page back to the client through the socket. The forwarding of paging objects involves the following steps. First a new paging object is created at the remote host. Second, pages are written to the remote paging object. Also to provide reliability, all the pages that are stored in the RAM are also stored in hard disk as a backup.

**Network Broker** A broker service is run in each network. This broker service is in charge of figuring the available resources i.e. servers available in the network and requesting the server, which is currently available to allocate the server's available RAM to the client machine. Every server periodically updates the broker's record of its capacity. The broker maintains a record of the capacity and allocations of each paging server on its network. The broker is also responsible for issuing access rights called read/write port permits to each server that services a particular client machine. These access rights are used to regulate the access of the port through which the server receives or writes the pages.

#### 4.1 Establishing connections

Initially when the user enters the room, and if the user wants to run a bigger process, the handheld device must first establish a wireless connection with the wired network through the available access points. Thus the network must provide all the security measurements for the handheld device to connect to the network and transfer information to and fro through the network.

#### 4.2 Run time transfer of idle process

When the handheld device is wirelessly connected to the network, the client pager program communicates with the broker to find out the available remote servers to store the program files. The broker maintains the list of all the available servers and informs the client pager about the availability of a server to host the client's files. The client then updates its memory map table with the details provided by the broker. The client pager program picks up the pages to be migrated and requests the broker to arrange the destination. Swapping algorithms like LRU algorithm [8] can be used to find out which pages to be swapped to the remote memory server. The client pager program learns from the broker of potential servers. The broker sends the list of all the potential servers to the client. When the client knows which server is going to act as the surrogate, the client sends a message to the corresponding remote server to arrange the required space for placing the object. The message encompasses the required paging size and the corresponding paging object. The server on request from the client creates the paging object according to the specified size in their RAM and stores the page.

When the client needs these pages, the client can request the server to page out the pages.

### 4.3 Transferring unused files (optional)

If more memory is required by the system then the client pager transfers the unused program files that are being stored in the RAM to the remote server's hard disk. These files would be restored only after the current application process finishes running. This is left at the discretion of the user. The client pager program notifies the user when it starts to transfer the files. The user can always abort this transfer if the user feels that it is not safe to store the program files. This option of transferring the unused program files are seldom used by the client pager, since by transferring the idle processes there is always enough memory to run the required application process and thus there is no need for the client pager to transfer the program files to the remote server.

### 4.4 Termination

The termination of the remote service is done by the following possible ways:

- When the current executing process finishes running, the remotely stored processes are retrieved back.
- When the user aborts the executing process, the remotely stored processes are retrieved back.
- If the environment senses that the user plans to leave the smart space, then the client pager asks the client whether he plans to leave the space. If the user says no, the process is resumed. If he says yes, then the current executing process is stopped and the remotely stored processes are retrieved. The sensing of user's intentions can be done by the following ways:

1) The pervasive environment senses the user's intentions of leaving the smart space by looking up the schedule of the user. For example if the user enters the airport cabin which is a smart space at 7:30 am and starts using the service. If the user has entered a schedule saying that he plans to catch the flight at 8:30 am, the client will know that the user plans to leave by 8:30 am. Thus it has to alert the user at least by 8:15 am asking whether he plans to leave. If the user says yes then the process is exited and the remotely stored processes can be retrieved. The client should also notify the user about the time taken to retrieve all the stored processes.

2) If there is no schedule, then the pervasive environment senses the user's location by the signal strength. If the signal strength becomes very less, then the environment assumes that the user is planning to leave and asks the user whether he has any plans of leaving. If he says yes, then the running process is exited and the stored processes are retrieved back.

In all these cases, the user has to be in the smart space to retrieve the stored processes back. The time taken to transfer back the stored processes depends on the user's location from the access points. Thus it is assumed that the smart

space has sufficient access points and antennas that can increase the distance coverage. The user has to wait or move around the smart space to retrieve all the stored processes back.

**Implementation Remarks** Our proposed model can be a component in an Aura Client that could be used by the environment manager to identify the remote memory service and utilize the service to send and receive remote pages. The Aura client can be implemented into the handheld PDA [1].

## 5 Design Issues to be considered

### 5.1 Reliability

The main concern in a distributed system environment is reliability. Some of the main problems in a distributed environment are network failures and the workstation crashing unexpectedly. When the network failure occurs, the clients cannot access the workstations and thus all the pages that the client had transferred to the server gets lost. Thus the client has to wait for the network to recover to get back its saved pages. The other problem is that a workstation may crash unexpectedly. This can be either because of electricity failure but this can be avoided by having UPS installed with the system. The other type of crash is the OS getting crashed or the server program getting crashed. Some workstations save the pages in RAM to their memory to prevent any kind of data loss in the event of these type of failures [6]. Suppose the broker fails then a new broker could be regenerated using the election and regeneration algorithm [3]. To ensure reliability there are different reliability policies available. They are called mirroring and parity logging [5].

**Mirroring** Mirroring is the simplest form of reliability. When a client sends a page to a server, the client sends copies of the page to two more servers. When the main server, which is currently serving to the client fails, one of the other backup server can start serving the client. The major disadvantage of mirroring is that mirroring wastes the remote memory by replication.

**Parity Logging** Mirroring wastes half the main memory. In order to avoid this, parity logging scheme could be used. If the client uses  $Y$  servers to transfer  $Y$  blocks, then each paged out page is XORed with the page size buffer maintained by the client machine. Then this page is transferred to the server. Round robin policy is followed to transfer the pages to each server. After transferring the  $Y$  pages to  $Y$  different servers, the buffer is also transferred to a parity server. When a server crashes, all the pages can be restored back by XORing the pages within the same parity group.

## 5.2 Security

Since wireless network is a shared medium, everything that is transmitted or received over a wireless network can be intercepted. Encryption and authentication are always considered when establishing connections. The IEEE 802.11b standard provides a standard encryption mechanism to do this by encrypting the traffic and authenticating nodes via the Wired Equivalent Privacy (WEP) protocol [10].

## 5.3 Robustness

Our proposed model supports high degree of mobility. Wireless standards like IEEE 802.11a and IEEE 802.11b support roaming with their mobile IP protocol. Our model would work fine when the user roams anywhere inside the smart space. The performance of the IEEE 802.11b and IEEE 802.11a are given below [14],

The data link rate of IEEE 802.11b is 11Mbps for an indoor range up to 107.5 feet, from 107.5 feet to approximately 177 feet the data rate is about 5.5Mbps, from 177 onwards up to 225 feet the data rate falls to 2Mbps.

The data link rate of IEEE 802.11a is 54Mbps up to 23 feet, then the data rate reduces to 48 Mbps from 23 feet up to approximately 43 feet. The data rate keeps on dropping when the range is increased and at approximately 173 feet the data rate is about 6Mbps. Up to 225 feet the data rate is maintained at 6Mbps. These data rates are highly acceptable considering the fact that the handheld devices have no hard disk facility to utilize the concept of virtual memory.

When the user decides to leave the smart space, the client notifies the server. When the server receives the notification, the server sends back all the files and pages out the pages back to the client machine. Thus the client restores back the remotely saved program files and the pages. When the client restores back all the files and pages, it notifies to the user that the whole process is complete. All the links between the client and server are broken.

## 5.4 Latency

The time taken to perform each read/write operation over the wireless network varies on the various parameters. Some of the parameters that play a vital role in determining the transfer time of the pages are given below:

- 1) Underlying network
- 2) Network hardware interface
- 3) IPC overhead
- 4) Machine's I/O Bus

Parameters like the network hardware interface and the machine's I/O Bus are vendor specific. IPC overhead depends on the underlying operation system. The IEEE 802.11b wireless network has bandwidth ranging from 2.4 to 2.4835GHz and can provide transfer rates from 1, 2, 5.5 and 11Mbps. IEEE 802.11a wireless

network the bandwidth ranging from 5.15-5.35GHz, 5.725 – 5.825 GHz and can provide transfer rates from 6, 9, 12, 18, 24, 36, 48 and 54 Mbps [14] and the IEEE 802.15.3 has 2.4 GHz bandwidth and can provide transfer rates from 11, 22, 33, 44 and 55 Mbps [14].

Thus the transfer rates for transferring a byte over an IEEE 802.11b are

1 Mbps	8 $\mu$ s/byte
2 Mbps	4 $\mu$ s/byte
5.5 Mbps	1.4 $\mu$ s/byte
11 Mbps	0.72 $\mu$ s/byte

The transfer rates for transferring a byte over an IEEE 802.11a are

6 Mbps	1.33 $\mu$ s/byte
9 Mbps	0.88 $\mu$ s/byte
12 Mbps	0.66 $\mu$ s/byte
18 Mbps	0.44 $\mu$ s/byte
24 Mbps	0.33 $\mu$ s/byte
36 Mbps	0.22 $\mu$ s/byte
48 Mbps	0.166 $\mu$ s/byte
54 Mbps	0.148 $\mu$ s/byte

The proposed transfer rates for transferring a byte over an IEEE 802.15.3 are

11 Mbps	0.72 $\mu$ s/byte
22 Mbps	0.36 $\mu$ s/byte
33 Mbps	0.24 $\mu$ s/byte
44 Mbps	0.18 $\mu$ s/byte
55 Mbps	0.145 $\mu$ s/byte

These transfer rates vary according to the distance of the user with respect to the access points in the smart space. The transfer rate for transferring a byte over a IEEE 802.3 wired Ethernet LAN is 0.8  $\mu$ s/byte [3]. It was found out that with the 10Mbps (0.8 /byte) transfer rate of the IEEE 802.3 network, the time taken to access a file was four times faster than accessing the same sized file on a magnetic disk [5]. Thus we see that from the rates given above, wireless

networks perform fairly well and with a network like the IEEE 802.11a or the IEEE 802.15.3 the time taken to transfer the pages would be very less compared to the time taken to transfer the pages over an IEEE 802.3 network. The total overhead of transferring the pages of handheld devices over a wireless network may be slightly slower than transferring the pages of desktop machines over wired links since the overall speed depends on many device specific parameters such as the inter process communication, processing power etc. But considering the fact that the handheld devices have very less memory and there is no hard disk to exploit the real power of the virtual memory concept, our proposed model will definitely be a good solution to this problem.

## 6 Related Work

The concept of remote paging had been done extensively [2, 3, 4, 5]. However these ideas were discussed for wired environments. Moreover the idea of remote paging was used for desktop and laptop machines. Bill N. Schilit and Dan Duchamp [3] use the concept of remote paging for mobile computers. They assume a symmetrical environment. This means that in their environment all participating machines are in some sense similar i.e. each machine can be both the client or server. In our paper we assume the environment is asymmetrical which means we have two distinct types of machines : stationary, powerful servers and a small mobile computing device. We use this concept of remote paging to propose a model to be used in the area of cyber foraging in a pervasive environment. Our proposed model considers handheld devices and wireless networks like the IEEE 802.11a, IEEE 802.11b and the WPAN IEEE 802.15.3 for the underlying network.

## 7 Conclusions

In this paper, we propose a model to solve the problem of insufficient memory for handheld devices. This was done by using the unused memory of the remote workstations, personal computers etc in the smart space of a pervasive computing environment to store the unused processes/pages of the handheld devices like PDAs over a wireless network through access points provided in the smart space.

## References

1. Satyanarayanan M.: Pervasive Computing: Vision and Challenges. IEEE Personal Communications (2001)
2. Evangelos P. Markatos, George Dramitinos: Implementation of a Reliable Memory Pager. Proc. USENIX Tech Conf., San Diego, California (1996)
3. Bill N. Schilit, Dan Duchamp: Adaptive Remote Paging for Mobile Computers. TR CUCS-004-91, Department of Computer Science, Columbia University, New York (1991)

4. Eric A. Anderson, Jenanna M. Neefe: An Exploration of Network RAM. CSD-98-1000 (1998)
5. Michail D Flouris, Evangelos P. Markatos: The Network RamDisk : Using Remote Memory on Heterogeneous NOWs. (1999)
6. T.E. Anderson, M. D. Dahlin, J. M. Neefe, D. A. Patterson, D. S. Roselli, R. Y. Wang: Serverless Network File Systems. ACM Transactions on Computer Systems, 14(1): 41-79 (1996)
7. M. Accetta, R. Baron, W. Bolosky, D. Golub, R. Rashid, A. Tevanian, M. Young: Mach: A new kernel foundation for UNIX development. USENIX Association Summer Conference proceedings, Atlanta (1986)
8. Milan Milenkovic: Operating Systems : Concept and Design. Tata McGraw-Hill, Second Edition (1997)
9. Douglas Boling: Minimizing the Memory Footprint of Your Windows CE based Program. [www.microsoft.com/msj/0598/memory.htm](http://www.microsoft.com/msj/0598/memory.htm) (1998)
10. Sultan Weatherspoon: Overview of IEEE 802.11b security. Intel Technology Journal, Network Communications Group, Intel Corporation (2000)
11. James C. Chen, Jeffrey M. Gilbert: Measured Performance of 5-GHz 802.11a Wireless LAN Systems. White paper, Atheros Communications, Inc, California (2001)
12. Joao Pedro Sousa, David Garlan: Aura: an Architectural Framework for User Mobility in Ubiquitous Computing Environments. IEEE/IFIP Conference on Software Architecture, Montreal (2002)
13. Technical Report: Palm OS Memory Architecture. Palm Inc. <http://oasis.palm.com/dev/kb/papers/1145.cfm>
14. James D. Allen: Blue Tooth and Beyond: Wireless Networks for Multimedia applications. Network Interop (2001)
15. Technical Report: IEEE 802.15 WPAN Task Group (TG3). [www.ieee802.org/15/pub/TG3.html](http://www.ieee802.org/15/pub/TG3.html)