International Journal of Computers & Technology
Volume 5, No. 2, May -June, 2013, ISSN 2277-3061

# AN EFFICIENT MEMORY MANAGEMENT TECHNIQUE FOR SMART CARD OPERATING SYSTEM

Shardha Porwal, Himanshu Mittal
Jaypee Institute of Information Technology, Noida, UP
Shardha.porwal@jiit.ac.in
Jaypee Institute of Information Technology, Noida, UP
Himanshu.mittal@jiit.ac.in

## ABSTRACT

This research paper examines memory management issues associated with Smart card EEPROM and proposes a new technique for memory management for smart card files. The entire work concentrates to suggest a new methodology on memory allocation and de-allocation using pointers, which gives better memory utilization.

## General Terms

Memory Management, Smart Card, Memory, File Allocation table.

## Indexing terms

Smart Card, EEPROM.

## Academic Discipline And Sub-Disciplines

Smart Card, EEPROM, Memory Management.

## SUBJECT CLASSIFICATION

Operating System.

## COVERAGE

Memory Maagemnt of EEPROM for Smart Card Operating System.

## TYPE (METHOD/APPROACH)

Experimental- made a simulator for smart Card Operating System using C language.

## 1. INTRODUCTION

Smart card applications were developed as an alternative to magnetic cards shortcomings [1]. The primary aim of magnetic cards is to provide machine readable data. However, the magnetic cards fails in providing security against tampering. Smart Card integrates large data and provides better security on a single chip. It also provides data security against unauthorized access and any modification of data [5, 6].

Currently smart cards are used in different parts of the world for various applications including passport applications, health care applications, institute identity cards, payment systems, Telecommunication Applications, Financial Transaction and many more [9, 10].

The memory management issues are of great concern in today's modern applications. This work proposes a new technique for memory management for smart card EEPROM (Electrically Erasable Programmable Read Only Memory) by maintaining a list of free memory blocks to be preserve in the smart card EEPROM called the free space list. This is different from paging scheme in which whole memory is divided into number of page where a table is maintained to keep information about allocated and free pages which takes a fixed size in EEPROM memory [4].The result of this technique has proved to be more effective in terms of memory management of EEPROM.

## 2. EEPROM AND SMART CARD FILES

The EEPROM is a chip with non-volatile memory. Data and program codes can be written to and read from the EEPROM under the control of the operating system [3].

File management systems for smart cards have an object-oriented structure which means that all information about a file is stored in a file itself. Files in such object oriented systems are thus always divided into two parts. The first part, referred as the file header, contains information about the layout and structure of the file and its access conditions. The modifiable user data are stored in the second part. The file body is linked to the file header by a pointer [11]. Fig. 1 shows the internal structure of the file.

International Journal of Computers & Technology
Volume 5, No. 2, May -June, 2013, ISSN 2277-3061

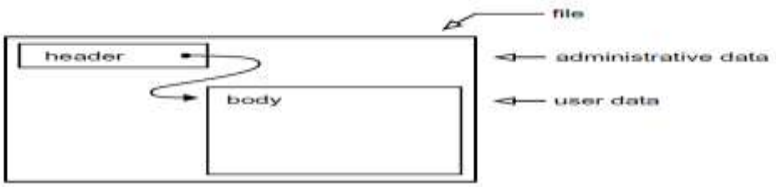


**Fig. 1 Internal structure of a file for a smart card files management system**

The Memory is allocated/ De-allocated for DF (Dedicated Files) and EF (Elementary Files). DF is the dedicated file just like a folders and EF is the Elementary file used to store the data [2]. Recently a lot of features are added with smart card such as multi application support, security (symmetric and asymmetric) etc for that we also need memory to maintain information [7]. As the size of memory increases the cost increases, therefore there is always a need to develop systems for utilizing the available memory in most effective and efficient way [8].

## 3. PROCESS FOR EEPROM MEMORY MANAGEMENT

As per the previous discussion the objective of this paper is to allocate the optimum memory for files with proper utilization of these allocated memory and merging the contiguous free memory blocks for further usage.

Following is the stepwise process of EEPROM Memory management as per the proposed technique. To demonstrate we have taken 20 bytes as header size for both DF and EF, the data area starting address is 1 and ending address is 3000 for simplicity.

We are maintaining a pointer variable (FS_Ptr) which contains information about the starting address of the free space list in the Last 2 bytes of EEPROM. The chain of free blocks is maintained from the end address of the memory that is 3000 minus the size of FS_Ptr. Fig. 2 shows the Structure of the node of Free Block List and Step by step procedure is also laid how to allocate the memory as per the scheme.

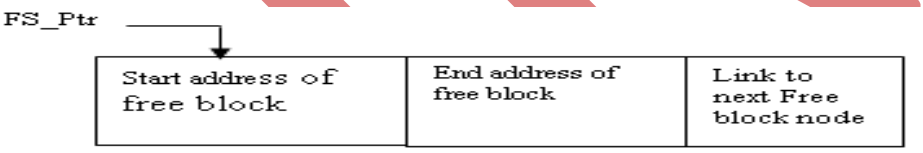


**Fig. 2 Structure of the node of Free Block list**

**Step1:** Allocate the space for MF:

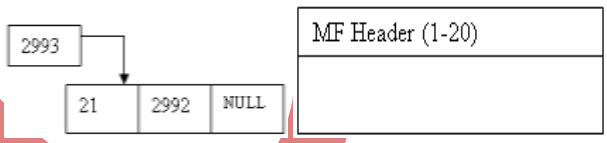


**Fig. 3 Allocate the memory for MF**

**Step2:** Create 3 different DFs (allocate the memory for Header):


**Fig. 4 Allocate the memory for DF1**

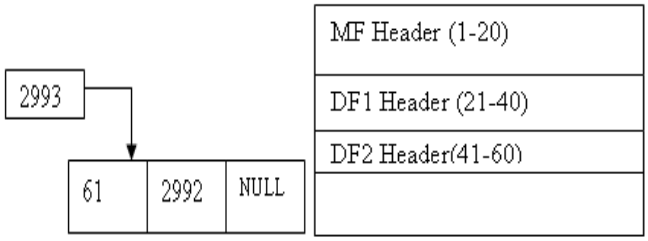


**Fig. 5 Allocate the memory for DF2**

International Journal of Computers & Technology
Volume 5, No. 2, May -June, 2013, ISSN 2277-3061

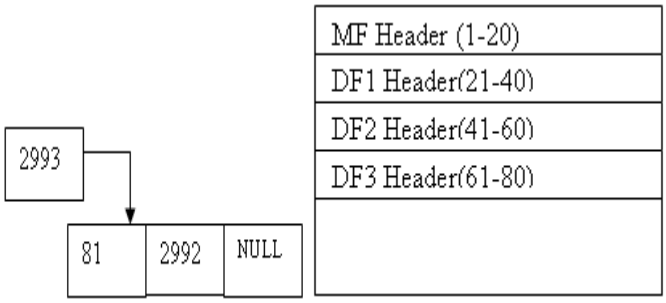


**Fig. 6 Allocate the memory for DF3**

**Step3:** Now Create 2 different EFs sequentially (allocate the memory for Header, body of given size 100 bytes):

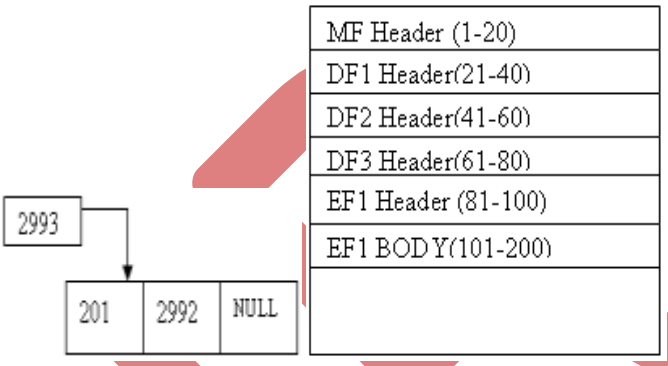


**Fig. 7 Allocate the memory for EF1**

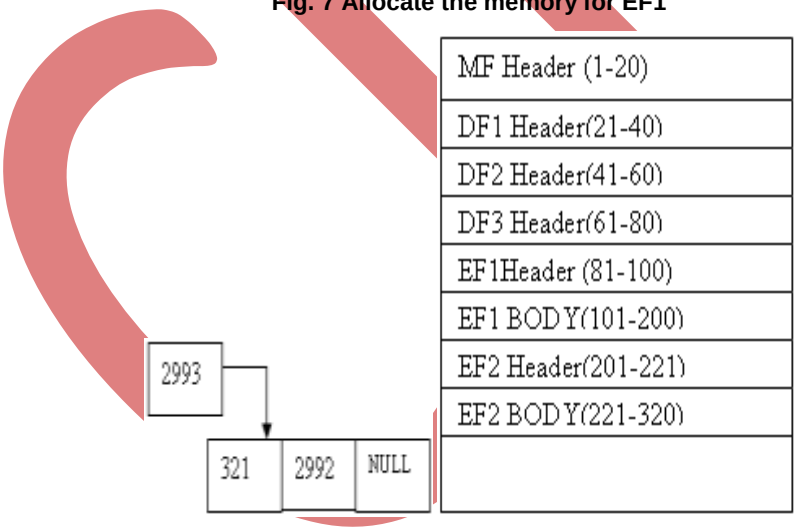


**Fig. 8 Allocate the memory for EF2**

**Step4:** Delete EF1 (de-allocate the memory for Header, body of given size 100):

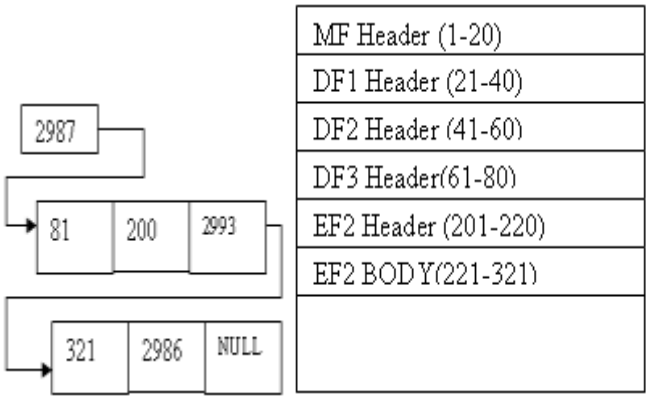


**Fig. 9 Free the memory occupied by EF1**

International Journal of Computers & Technology
Volume 5, No. 2, May -June, 2013, ISSN 2277-3061

**Step5:** Delete DF1, DF3, and DF2 (de-allocate the memory for Header, body of given size):

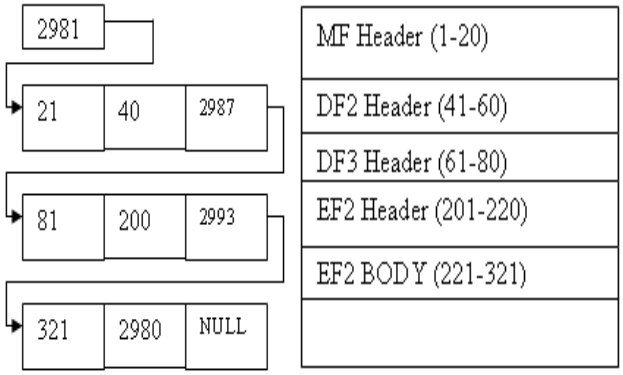


**Fig. 10 Free the memory occupied by DF1**

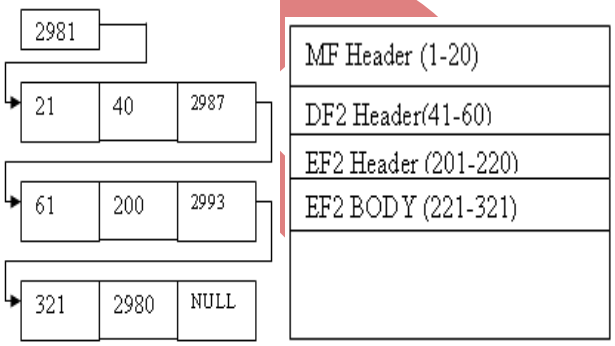


**Fig. 11 Free the memory occupied by DF3**

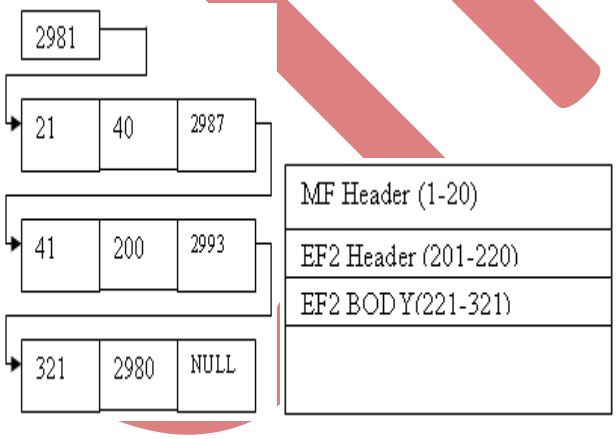


**Fig. 12 Free the memory occupied by DF2**

Now there will be two continuous free memory blocks, therefore OS will combine them and will make a single block.

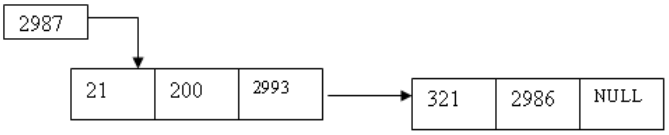


**Fig. 13 Combining continuous free memory blocks**

**Step6:** Allocate memory for a DF (DF1).

International Journal of Computers & Technology
Volume 5, No. 2, May -June, 2013, ISSN 2277-3061

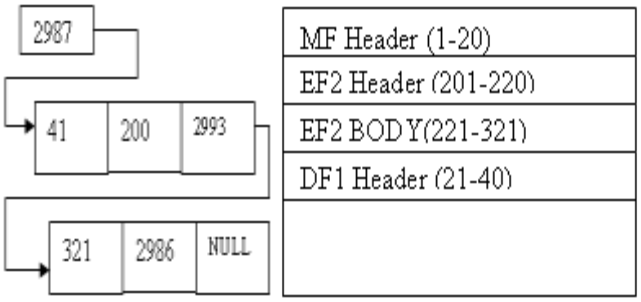


**Fig. 14 Allocate the memory for DF1**

**Step7:** allocate memory for an EF, which require 300 bytes for its body part.

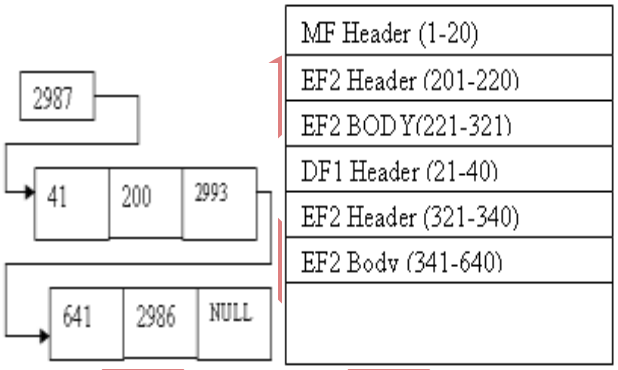


**Fig. 15 Allocate the memory for EF3**

## 4. RESULTS AND CONCLUSION

Memory Requirement comparison is shown for 4K EEPROM. If we divide the memory into pages and each page is of 32 bytes then numbers of pages will be 128. If we consider that page table contains page number, page status and next page fields only then it needs 128 x 3= 384 bytes, whereas our scheme requires:

| Number of free blocks | Memory required |
|---|---|
| 1 | 8 bytes |
| 2 | 14 bytes |
| 10 | 62 bytes |
| 32 (avg.) | 194 bytes |

**TABLE I Memory Usage**

It was also observed that memory management by using the proposed scheme overcomes issues such as Easy Allocation/De-allocation, No Internal Fragmentation, Optimal Utilization of Memory, No Page-Size Restriction, Dynamic free block List, and Compaction.

## 5. ACKNOWLEDGEMENTS

The authors would like to acknowledge G. L. Bajaj Institute of technology and management, Gr. Noida for providing the platform for the Project on smart card OS, Prof. Naresh Maheshwari for his valuable guidance, Mr. Anil Kumar Singh, director of Amity s/w inc. for the fruitful discussion and Mr. Alok Mishra for his support.

## REFERENCES

**Advanced card System Ltd. http://www.acs.com/**

[1] Deepak Nagawade, Implementation of SCOSTA-CL based Smart Card Operating System (SCSOS), Indian Institute of Technology, Kanpur, India June 2008.

[2] IBM Deutschland Entwicklung GmbH Information Development, *MultiFunction Card 4.1 Family Programmer's Reference Version 1.0*, Dept. 3248 Postfach 1380 71003 Boeblingen Germany.

[3] IM Y. Jung, Sung I. Jun, Kyo I. Chung, *A Persistent Memory Management in Java Card*.

[4] Mastercard home page. http://www.mastercard.com/index.html/.

International Journal of Computers & Technology
Volume 5, No. 2, May -June, 2013, ISSN 2277-3061

[5] PrEN 1545-4. *Identification Card Systems – Surface Transport Applications – Part 4*: Vehicle and Driver Licencing, 1997.

[6] R. Shankesi, *Developement of an Operating System for Smart Card. Master's thesis*, IndianInstitute of Technology Kanpur, Dept. of CSE, India, April 2002.

[7] Simple operating system for smart card education.http://www.mbsks.franken.de/sosse/.

[8] Visa International, USA home page. http://www.visa.com/.

[9] Visa, *Visa Smart Debit/Credit Certification Authority Service Description (VSDC) version 2.1*, December 2005.

[10] Wolfgang Rankl et al; *Smart Card Handbook Third Edition*, Giesecke and Devrient GmbH, Munich, Germany, John Wiley & Sons, Ltd publication.

## Author' biography with Photo

### Ms. Shardha Porwal

Shardha Porwal obtained her M.Tech. in Digital Image Processing from Maulana Azad National Institute of Technology, Bhopal. She got her Bachelor's degree in Computer Science & Engg. from Anand Engineering College, Agra, affiliated to Uttar Pradesh Technical University, Lucknow. Her research areas are Digital Image Processing, Operating System, and Algorithms.

### Mr. Himanshu Mittal

Himanshu Mittal obtained M.Tech. in Software Engineering from Delhi Technological University, Delhi (DTU). He obtained his B.Tech. in Information Technology from Hindustan Institute of Technology, Greater Noida affiliated by Uttar Pardesh Technical University, Lucknow. His area of research is Software Engineering, Operating System, Wireless Sensor Networks and Cryptography and Network Security.