# Classification of Smart Card Operating Systems

Reza Asgari, Reza Ebrahimi Atani

*Department of Computer Engineering, Faculty of Engineering, University of Guilan*
*rezaasgari.68@gmail.com, rebrahimi@guilan.ac.ir*

## ABSTRACT

Today, smart cards are widely used in variety of applications of human life. The nature of these cards depends on theirs operating system, in other words the operating system determines that card can be used in what field. Hardware development on the one hand and increasing use of smart cards on the other, have led to that the operating systems have progressed dramatically. Moving from special-purpose operating systems with single application into multi-purpose operating systems with open architecture, is proof of claim. Increase the number of operating systems and variety of their applications, caused that the need to categorize and classify operating systems be felt more than ever. Smart card operating systems can be categorized based on different parameters such as file management, applications management, and historical generations. In this paper we will discussion about smart card operating systems and their classification based on these parameters.

**Keywords**: Smart Card Operating System, File Management System, ISO/IEC 7816.

## 1. INTRODUCTION

Smart card is a plastic card with size of credit cards (about 5/5 on 8/5 cm) that has embedded memory chips and processor for data storage and processing [1]. In other words, a smart card is a small computer that is mounted on a plastic card. Ability to provide security and ease of use are the most important reason for extension of smart card usages in different domains, these cards can processing and storing significant volumes of data. Increasing CPU power, memory capacity and ability of use cryptographic coprocessors in smart card microcontrollers led to smart cards can used in variety of applications such as mobile telecommunication, payment systems, identification, security, e-health, e-passport, e-ticket, and transportation [2].

The main core of smart card is its microcontroller, this microcontroller is composed of two parts: hardware and software. The hardware component includes all hardware modules that can be used in the card and may include the following components: a 8, 16 or 32 bit processor, communication interface, special-purpose hardware (such as cryptographic coprocessor, MMU, clock generator, timer and so on), and different memory types (that may including 256 B to 16 KB RAM, 4 to 400 KB ROM, 1 to 500 KB EEPROM, and 2 KB to 2 MB FLASH) [1, 3]. Software unit includes all applications and operating system that determine the nature of the card.

Smart card operating system is a set of codes which are executed on the card processor and is responsible for memory management, data exchange, instruction execution and cryptographic algorithms management [3]. The operating system must be able to perform these tasks with card limitations such as low memory, low-speed interfaces and unusual hardware features. In other words, the nature of smart card will be determined by its operating system. Therefore necessity of attention to operating system in card development process is inevitable.

Today according to various usages of smart cards, different operating systems are designed and implemented for them. Expansion of these operating systems has led to the need for a comprehensive classification of operating systems. Smart cards operating systems can be classified according to various parameters that different generation, file management, and application management are most important known parameters. The rest of the paper is organized based on this issue as follows. Section 2 present a definition for a smart cards operating system and describe briefly its tasks. Section 3 reviews different generations of operating systems. Operating systems are classified based on file management in section 4. Section 5 discussed about different architectures for application management. Section 6 and 7 present discussion and conclusion of the paper.

## 2. SMART CARD OPERATING SYSTEM

Based on German DIN 44300 standard, an operating system is “the programs of a digital computer system that together with the properties of the computing system form the basis for the possible operating modes of the digital computing system, and which in particular control and monitor program execution” [3]. In simple terms can be said: smart card processor running software that provides a standard interface to manage card's components, this software is called operating system.

Multi-Functional Card (MFC) operating system can be noted as a one of the first operating systems that was product in smart card area. MFC was introduced in 1990 by IBM. Update ability is main feature of this operating system that can be done via script protocols. One of the most MFC usages is in electronic payment systems, and particularly in electronic purse cards [4].

Usually, the operating systems codes are stored in ROM memory thus they have limitation such as error correction, therefore operating systems must have a very small error and high reliability. Of course it should be noted that from the perspective of operating system there is a bitter fact that implementation of a mechanism is affected by the used hardware. Another important point that should not be overlooked is that the implementation of cryptographic functions must be done in very little time.

The operating system enables the microprocessor to manage and control card memory. Providing a standard way to exchange data between the card, card reader, and/or applications is one of the main tasks of the operating system. In addition to these tasks, card operating system is responsible for access control, authentication, and information security. In summary, a smart card operating system must perform the following tasks [3, 5]:

- Data exchange between card and terminal
- Control command execution
- Files management
- Management and execution of cryptographic algorithms
- Management and execution of program’s codes

Smart card operating systems can be categorized based on parameters such as, file management systems, how to manage the programs, and historical generations.

In the following of this paper, we will review the smart card operating systems categories.

## 3. HISTORICAL CLASIFICATION

Research experience in the field of smart card operating systems goes back to the early1980s, so can be said that in the early 1990s, there were few operating systems that is designed specifically for smart cards [3]. The development process of smart card operating systems is as well as other operating systems, it is started from a special-purpose program for a particular application and consecutive has been developed. The result of this development is a structure, namely a general-purpose operating system that can simply be used and configured in different platforms. Smart card operating systems developments can be classified into three different generations, which in the following will pay to brief review these generations.

The first generation of smart card operating systems were library based operating system. These operating systems were used until 1987 in the smart cards like C-NetZ and D-NetZ in German telecommunication network [3]. This type of operating systems by providing a set of predefined functions allows applications to communicate with hardware efficiently. First generation operating systems are usually designed and implemented for specific applications therefore they were gradually replaced with next generation of operating systems.

The next step in the development process of the operating system moves a specific way toward an open architecture which it results was a monolithic operating system. An example of the application of this system was the primary GSM cards. The second generation operating systems were multi-functional and conceptually are more open than the previous generation. At this time, there were only standards for data structure and the basic instruction set, it makes up each manufacturer use its own special standards [6]. Excessive hardware dependency in the second generation led developers to moving toward operating systems that can be implemented with minimal changes on different hardware. The result was the design and implementation of a layered operating system that are largely independent of the hardware.

Layered operating system is a modern operating system that has functions such as memory management, multiple file tree and state machine [3]. Features that these operating systems provide are very similar to common operating systems in personal computers. Layered operating system that can handle independent multiple programs and prevent interference between programs. Many of these operating systems have complex state machine and wide instruction set, so that some of them can support multiple transport protocols. Today, all modern operating systems have a layered structure with an interface layer to communicate with the hardware. Figure 1 shows three generations of these types of operating systems. Multos [7, 8], CardOS [9, 10], STARCOS [11, 12], JCOP [13], TCOS [14],Cyberflex [15], and Payflex [16] can be cited as examples.

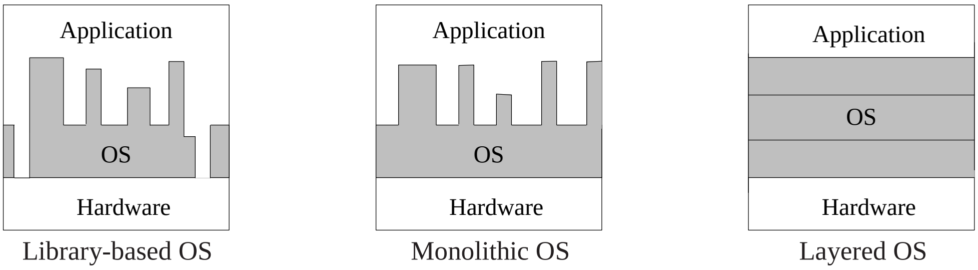

FIGURE 1. Simple architecture of three different generations of smart card operating systems [3]

## 4. FILE MANAGEMENT BASED CLASIFICATION

Based Modern file management systems are typically designing object-oriented. This means that description data is kept inside the file and each file before can be used must be selected. The file structure is divided into two parts. The first part, named header file, kept the file information like internal file structure and access conditions. The second part, named body, is responsible for storing user or programs information. These two parts usually due to security issues are stored in different pages of memory and linked together by a pointer [3, 17].

File tree (that is used in the card) is made logically based on header files. The header files typically have pre-defined fields that can be managed in two ways: static and dynamic. In static mode, information of file header is fixed and unchangeable. Using this structure increases the speed of file access but reduces flexibility. To fix this problem dynamic structure can be used, this structure provides the modify ability in the file tree. Smart card operating systems based on file management system can be divided into two categories:

- Static operating systems
- Dynamic operating systems

Static operating systems are operating systems that use a static file management system. Static file management systems are commonly used when operating system is used for a set of pre-defined targets so that possible changes in the structure of the file system (such as adding or deleting a file) will not exist. These file management systems for having a fixed header structure can act as a very effective and reliable and boost operating system performance in terms of speed. Drawback associated with this system is that after card production is not possible to load/delete files on/from the card. Most operating systems of first generation can be considered in this category.

In contrast, there are operating systems that able to load/delete files on/from the card after card production. This type of operating system, because they use a dynamic file management system, called dynamic operating systems. Dynamic operating systems provide ability for developers to load or delete files with high-security. Today, most multi-application operating systems are placed in this category, for example TCOS [14], Athena [18], JCOP [13], and STARCOS [11, 12] can be cited.

## 5. CLASIFICATION BASED ON APPLICATION MANAGEMENT

The One of the most important operating systems classification is based on applications management. Operating systems based on how manage applications can be divided into two categories:

- Closed (Native) operating systems
- Open (Global) operating systems

## 5.1 CLOSED OPERATING SYSTEMS

Most cards use operating systems that have been developed in order to achieve specific goals and they can directly call functions in the hardware. Application developers must use from low-level language that used in these operating systems for developing applications, it is the main reason that why the operating system is called closed. Because closed operating systems designed and implemented tailored to specific hardware and applications then they are highly effective performance and their memory consumption is optimized.

From the perspective of the developers closed operating systems are completely under control, and if that is required it can be changes or added new features. But, the drawback associated with this operating system is that applications must be written in low-level language that is recognized by the operating system.

Security of cards that use the native operating system fully depends on the operating system. Therefore, usually closed operating systems are under different evaluation and tests such as Common Criteria (CC) [19]. PayFlex [16], CardOS [9, 10], PROTOS [20], SCOSTA-CL [21], COMBOS [22] and Caernarvon [23, 24] are examples of closed operating system. Closed operating systems developer companies for security reasons often prefer to publish only general information about their operating system and usually avoid from expression of the details. Table 1 presents review of general characteristics of the closed operating system.

## 5.2 OPERATING SYSTEMS

On the opposite of closed operating systems, there are open operating systems. The reason of naming these operating systems is their open architecture that enables the developers to use high-level languages for writing their applications for the operating system. Open operating systems usually have an interpreter or virtual machine that can convert high level program codes into machine level language [26]. Due to variety features that supported by open operating systems, they are typically have more complexity and heavier code size than closed operating systems.

Because the use of the virtual machine or interpreter, open operating systems are considered largely independent of hardware. The concept of hardware-independent, meaning that developers can simply write their desired applications for operating system, by using high level known languages and without detailed knowledge about the hardware. Of course it is important to note that due to the complexity of the operations, such as running virtual machines, in this type of operating systems, often open operating systems require more sophisticated hardware than closed operating systems. For example, we can note to use of the 16 or 32 bit processor on open operating systems compared with the 8-bit processor used in most closed operating systems.

As mentioned, the remarkable thing about the open operating system is that allow to applications developers, even those who don't have a role in the development of the operating system, to write their applications in high level languages such as C, C++, Java, or Basic; And due to consideration of security policies load and run them on the card. In contrast to closed operating systems, developers of open operating systems usually publish comprehensive information about the operating system. In general, the main features of these operating systems can be expressed as follows [3, 26, 5]:

- Run the card applications on different chips
- Separating the operating system from applications
- Support from different communication interfaces and protocols that can be used appropriate to the user's needs
- Ability to update the files, programs, and even some modules of the operating system (if the operating system is stored on a writable memory such as Flash) after card production
- Various access control systems and security mechanisms that can be used appropriate to the user's needs
- Very strong firewall to isolate applications sources from each other

Open operating system concept was expressed first time in 1996 by Schlumberger, with the introduced the first platform-independent architecture for smart cards which able to execute programs that were written in Java [4]. Then in 1997 Java Card Forum [27] with membership of several companies in the field of smart cards was formed and formally undertook the responsibility of standardization and development of Java Card. If we want to present a definition of the Java Card must say that: Java Card is a smart card that its microcontroller can provide possibility of running java virtual machine and java card runtime environment [3].

Java Card is a multi-application smart card that provides ability of management and implementation of several programs that are written in the java language (java applets). Although from a technical perspective (because of the lack a file management system) Java Card is not recognized as a real operating system for smart cards; but is mentioned typically as one of the smart card operating systems. Nowadays there are many operating systems based on Java Card, as examples some of these operating systems can be found in Table 2.

Table 3 briefly reviews the main features of three popular operating systems that have open architecture, namely MultOS, BasicCard, and Windows. MultOS [7, 8] is an open operating system that is used in the multi-purpose smart cards and first time developed by Netwest Bank of England to support electronic purse applications. Today's this operating system developed by various companies such as Samsung, who are known as Multos Consortium [8] and it can be used in all known applications of smart cards, such as payment systems, transport, access control, identification, communication, e-health and etc. According to the official announcement on 19 November 2013, more than 500 million cards worldwide are using this operating system [29]. Different versions of this operating system have been developed by different companies and have different characteristics, information about these releases can be found in Multos official website [8].

Basic Card [30] is an operating system with open architecture that was introduced in 1996 by German Zeitcontrol. Nowadays, there are several version of this operating system that can be used tailored to the needs of users and different hardware. BasicCard operating system use an interpreter for Basic programming language that occupies about 17 KB of ROM memory and support all of main commands and data units such as string and floating point number.

Basic Card in comparison other smart card operating systems with the interpreter, have compressed program code and relatively high execution speed. These two, caused by the fact that basic language can quickly and easily translate the code and in preparing the application does not require a complex security mechanism. So where is the need to develop applications on smart card quickly and easily, the basic card is a suitable alternative instead of other operating systems. Another positive feature of the Basic Card is that the basic programming language is very easy and low price to learn.

TABLE 1.

Important characteristics of five native operating system

| **Operating system** | **Company** | **Evaluation Assurance Level** | **Cryptographic algorithms (bit)** | **Important properties** | **Important usages** |
|---|---|---|---|---|---|
| CardOS [9, 10] | Atos (Siemens) | EAL 4+ | 3DES, AES (up to 256), RSA (up to 4096), SHA-1, -224, -256, -384, -512 | Multi-application, dynamic and flexible file system, possibility of adding new files or programs, ISO/IEC 7816 compatibility, secure massaging, and T=1 communication protocol | Identification, signature cards, e-government, e-passport, driving license, and e-health |
| Caernarvon [23, 24] | IBM | EAL 7 | DES, 3DES, AES, RSA , ECC | Multi-application, possibility of adding new files or programs, secure authentication mechanism, support contact and contactless cards, and high assurance level | Security, e-government, identification, transportation, and electronic payment system |
| SCOSTA-CL [21] | Indian institution of technology | - | 3DES, SHA-1 | Multi-application, secure massaging, support contact and contactless cards, T=1 and T=0 communication protocol, and ISO/IEC 7816 compatibility | Transportation |
| TCOS [14] | T-System (Germany) | EAL 4+ | 3DES (up to 168) , DES (112), ECC (320), RSA (up to 2048), SHA-1 | Secure massaging, support contact and contactless cards, T=CL, T=1, and T=0 communication protocol, and ISO/IEC 7816 compatibility | Signature card, e-passport, and tachograph card |
| SECCOS [25] | ZKA (Germany) | EAL 4+ | 3DES, RSA , SHA | Dynamic authentication, and EMV and ISO/IEC 7816 (part 8and 9) compatibility | Signature Card, e- health, e-banking and payment systems |

TABLE 2.
Characteristics of some operating systems that used in java cards

| Operating system | Communication protocol | Cryptographic algorithms (bit) | Important properties | Important usages |
|---|---|---|---|---|
| Athena [18] | contact and contactless cards, T=0, T=1, and T=CL protocols | AES (up to 256), DES and 3DES (2 and 3 Keys), RSA (up to 4096), SHA-1, -512, ECC (up to 384) | Multi-application, EMV, Global Platform, and ISO/IEC 7816 compatibility, secure massaging, dynamic file management system, EAL 4+, and garbage collection | Electronic payment system, e-government, e-passport, e-health, driving license, and national ID |
| IDCOR [28] | T=0 and T=1 protocols | 3DES, AES (128 to 256), SHA-1, -224, -256, -384, -512, RSA (up to 2048), ECC(up to 512) | Multi-application, garbage collection, ability for working with Flash memory, EAL5+, FIPS 140-2 Level 3, and Multiple Logical Channels | Suitable for identification and access control usages |
| JCOP [13] | contact and contactless cards | 3DES (56 up to 168), AES (256), RSA (up to 2048), ECC (320), SHA-1,-2 | Multi-application, EMV, Global Platform and ISO/IEC 7816 compatibility | Transportation, e-government, identification, national ID, e-health, e-passport, and electronic payment system |

TABLE 3.
Three popular operating system that have open architecture

| Operating system | Supported platform | Supported languages | Communication protocol | Cryptographic algorithms (bit) | Important properties |
|---|---|---|---|---|---|
| Multos [7, 8] | .NET and java | MEL, C, Basic, and java | T=0 and T=1 protocols, Contact and contactless cards | AES, DES, SHA, 3DES, ECC, RSA | Multi-application, multi-purpose, powerful firewall, support different standards such as EMV, ISO/IEC 7816, and ETSI (for SIM cards) |
| BasicCard [30] | Basic | Basic | T=0 and T=1 protocols, Contact (after V 7.5) and contactless cards | AES, DES, SHA, ECC, RSA | Multi-application, multi-purpose, and ISO/IEC 7816 compatible |
| Windows [31] | .Net | Microsoft visual Basic and C++ | T=0 and T=1 protocols, Contact and contactless cards | AES, DES, SHA, 3DES, RSA | Multi-application, multi-purpose, ISO/IEC 7816 compatible, garbage collection, and fat based file system |

## 6. CONCLUSION

Growth Need to modify and update of operating systems has caused that developers design operating systems that can also be implemented in EEPROM and Flash memory. Perhaps the most important negative point with this type of operating

ISSN: 2252-4274 (Print)
ISSN: 2252-5459 (Online)

systems is that the possibility of modifying OS codes is provided for attackers, which threaten the security of the operating system's codes.

One of the key points in the design and implementation of a smart card operating system is related to the reliability. Reliability means that the operating system will have very few errors, and has the ability to resist against intrusions. Operating systems for enhancing the security of smart card typically use various cryptographic algorithms. DES, 3DES, AES, RSA, SHA, and ECC are the most important cryptographic algorithms that are widely used in the field of smart cards. These algorithms based on the applications and cards features can be used by keys with different length. To enhance efficiency of cryptographic algorithms, cryptographic coprocessor can be used in smart cards.

Smart card operating systems to improve the security level and reliability among its users, usually evaluated by various criteria. CC is one of the most popular certifications in the field of smart card evaluations. Acceptable level for this standard in relation to the operating systems and cryptographic algorithms is usually EAL 4. Operating systems that are at this level or higher levels have high assurance level of security. Although the high-level of assurance 100% not guarantee the security of operating system, but it provide assurance that the operating system is in high level of security and can resistance against errors and possible attacks.

## 7. CONCLUSION

Growth of hardware microcontroller and smart card usages has led to increasing the size and complexity of operating systems. Smart card operating system is a set of codes which are executed on the card processor and is responsible for management issues. Evolution path of smart card operating systems is began with native operating system for special-purpose and single application, and continued to multi-purpose operating systems with open architecture. These operating systems can be classified in three library-based, monolithic, and layered categories. Smart card operating systems to manage files usually use a file management system; Based on the file system that is used, operating systems can classified to static and dynamic. Smart card operating systems can have open or closed architecture. Open operating systems can be used as a platform that enables programmers to write their applications hardware-independent; but closed operating systems used in order to achieve specific goals and they can directly call functions in the hardware.

## REFERENCES

[1] G. Selimis, A. Fournaris, G. Kostopoulos, and O. Koufopavlou, “Software and Hardware Issues in Smart Card Technology”, COMMUNICATIONS SURVEYS & TUTORIALS, IEEE, Vol. 11 (3), PP. 143-152, 2009.

[2] E. M. Keith, and M. Konstantinos, *Smart Cards, Tokens, Security and Applications*, Springer, 2008.

[3] W. Rankl, and W. Effing, *Smart Card Handbook*, 4th edition, Wiley Publishing, 2010.

[4] Mike Hendry, *Multi-application Smart Cards Technology and Applications*, Cambridge University Press, 2007.

[5] Heng Guo, “Smart Smart Cards and their Operating Systems”, HUT, Telecommunications Software and Multimedia Laboratory.

[6] D. Damien, G. Antoine, G. Gilles, and J. Sebastien, ”Smart Card Operating Systems: Past, Present and Future”, 5th USENIX/NordU Conference, Sweden, 2003.

[7] “MULTOS Standard C-API”, Technical Report, MAOSCO Inc, 2013.

[8] http://www.multos.com/.

[9] “HiPath SIcurity CardOS V4.3”, Technical Data Sheet, Siemens.

[10] “CardOS V5.0 Multifunctionality”, Technical Data Sheet, ATOS, July 2012. Available at: http://atos.net/content/dam/global/we-do/cardos-v5-datenblatt.pdf.

[11] “STARCOS 3.0”, Technical Report, Giesecke & Devrient GmbH, 2004.

[12] “STARCOS 3.5 ID The smartcard operating system for next-generation ID documents”, Technical Report, Giesecke & Devrient GmbH, 2012.

[13] “Java Card OS for NXP’s SmartMX family of secure microcontrollers”, NXP Corporation, 2012.

[14] “Telesec Tcos 3.0 Smart Card OS”, Technical Report, 2010. Available at: http://www.telesec.de/tcos/LB_TCOS3.0_100318_dt.pdf.

[15] “Cyberflex Access The Java card for information security”, Axalto Corporation, 2003.

[16] “Payflex: Smart Cards for Community Electronic Purse ”, Technical Report, Schlumberger, 1997.

[17] Wolfgang Rankl, “Smart Card Applications: Design Models for using and programming smart cards”, Wiley & Sons, 2007.

[18] “Athena Smart Card”, Product Datasheet, Athena Smartcard Solutions Inc, 2012.

[19] http://www.commoncriteriaportal.org/cc.

[20] “PROTOS, Multipurpose PRofessional Token Operating System”, Technical report, United Access. Available at: https://www.united-access.com/protos.

[21] Deepak Nagawade, “Implementation of SCOSTA-CL Based Smart Card Opetating System”, Master of technology thesis, Indian institute of Technology, Kanpur, 2008.

[22] “Combos Cpa for Advanced Private Lable Projects”, teChniCal speCifiCations, Trüb AG, 2011.

[23] P. A. Karger, S. McIntosh, E. Palmer, D. Toll, and S. Weber, “Lessons Learned Building the Caernarvon High-Assurance Operating System”, Security & Privacy, IEEE, Vol. 1, pp. 22-30, 2011.

[24] P. A. Karger, D. Toll, E. Palmer, S. McIntosh and S. Weber, “Designing a Secure Smart Card Operating System”, IBM Research Report, 2008.

[25] “SECCOS: SEcure Chip Card Operating System ”, Technical Report, Zenteraler Kredit Ausschuss, 2006.

[26] “SmartCard Handbook”, Department of Finance and Deregulation, Australian Government Information Management Office, 2008.

[27] http://javacardforum.com/.

[28] “IDCore 40”, Technical Report, Gemalto Inc, 2013. Available at: http://www.gemalto.com/products/top_javacard.

[29] http:// www.multos.com/news/view/multos_celebrates_500_million_issued_devices.

[30] “Professional and MultiApplication BasicCard”, Product Datasheet, BasicCard Corporation, 2012. Available at: http://www.BasicCard.com.

[31] “Introduction to Windows for Smart Cards”, Microsoft Corporation. Available at: http://technet.microsoft.com/en-us/library/dd277375.aspx.