Smart cards integration in Distributed Information Systems : the Interactive execution model

Sebastien JEAN, PhD Candidate

University of Lille 1, LIFL Lab, Cité Scientifique, Bat M3, 59655 Villeneuve d'ascq cedex ,France

Tel : (33) 3 20 43 47 14, Fax : (33) 3 43 47 12, Email: jean@lifl.fr

Didier DONSEZ, Assistant Professor

Sylvain LECOMTE, Assistant Professor

University of Valenciennes, LAMIH Lab, Le Mont Houy, 59313 Valenciennes Cedex 9, France

Tel: (33) 3 27 51 19 50, Fax: (33) 3 27 51 18 29, Email: {donsez,slecomte}@univ-valenciennes.fr

Abstract

Wide distributed applications involve several partners, this requires security features when the software components are deployed between the partners. Moreover, the end-user is more and more nomadic and uses several terminals to browse his services. We believe smart cards can be used to meet these new requirements. However, smart cards acts mainly as isolated servers. We propose to improve the smart card execution model in order to pilot the application as a client and to serve some external requests as a server. We discuss about the benefits of such an approach, uses cases and implementation issue.

Keywords

Information systems, smart cards , client / server applications, nomadic applications

1 Introduction

In a Client / Server architecture, the server software components of the application are managed by an enterprise (the enterprise, an administration, …) which have deployed them. All the software components developed within the same enterprise are supposed to be confident with each other.

At opposite, in the WWW environment or in the mobile agent context, clients and servers may have to download and run software components written by a potential hacker. For example in a typical electronic commerce (EC) application, the customer browser may download an applet and the merchant server may receive a mobile agent (named aglet) from anonymous customers **[NY97]**. In this EC application, four organizations provide a component which contribute to the application global execution. These four organizations are the customer, the merchant, and their respective banks. The application is the result of the cooperation of these components. But each component of this application is not confident with the others since one (a customer or the merchant) may try to hack the other one or the banks. In this context, the design of the EC application must take into account confidence relationship besides the functionality of involved software and hardware components.

In another way, the computing is becoming ubiquitous **[W93]** as the end-user uses various kinds of terminals to browse his favorite services (mail, news survey, portfolio management, …) all long the day. These terminals can be always or transiently connected to the network: With xDSL links, PC, SetTopBoxes or Game Consoles are in the first case but mobile phones are in the second category even if cellular networks guarantee more and more coverage. Since terminals are more and more nomadic, such as cellular phones, PDA or even Nomadic Game Console, the end-user may start to browse a service on a first terminal, stop browsing to roam and continue with an another kind of terminal (i.e from his PC to his mobile phone if he leaves his office after work or from his phone to his STB in his hotel room). We propose in **[CD97]** to follow the end-user by using a mobile agent when he roams in order to maximize response times to user network environment (mailbox, push information…). In this context, the end-user must carry his environment (address book, bookmarks, credentials, …) from one terminal to another. Finally, the application designer has to take into account three specifics needs in new application design: the security, the mobility and the availability of the components.

We believe smart card offers a solution to secure application deployment and terminal mobility. This requires to improve the execution model of smart card in applications.

In section 1, we have presented the requirements in the design of large distributed applications such as EC applications. The next section quickly presents an overview of smart card technology and their main properties. The section 3 presents the actual execution models to execute embedded application code in smart cards. Thus, we propose in section 4 the interactive model, a more flexible execution model which enables embedded components to work both as client and server. We also discuss about implementation issues. Then we concludes with related works in the section 5.

2 Smart card in information systems

2.1 Technological aspects

Microprocessor cards, also called smart cards **[ISO87]**, provide both hardware and software security mechanisms to ensure embedded data security. A security block prevents against physical attacks providing mechanisms like light or abnormal voltage detection, and smart card operating system controls access to the information. On the top of that, data security in the card is also ensured through the use of various coding techniques. Thanks to these techniques, smart card is the best support to protect personal data in distributed environments. Consequently, smart cards can be found in various domains like banking, mobile phone, TV, healthcare, money games, … The range of their applications is continuously growing and evolving.

The main problems of smart cards are hardware constraints such as size of RAM, ROM and EEPROM, and low-bandwidth communications (19200 Bds). Moreover, accessing services or data held by a smart card or remotely sending a message to it is only possible when the card is slotted into a terminal connected to a network. Unfortunately, this situation is quite rare because an individual decides to use his smart card only when he needs to access an on-line service and when he can use a suitable terminal. Thus, a connection to the network is always initiated by the user, never by the system.

2.2 Benefits of smart card use

Security

The smart card is a high security hardware component, which is often used to secure electronic payment (credit card, e-purse) and more sociologically to establish confidence between several organizations involved in a same application.. For example, the smart card can assume the user authentication for all the other application components. It can also be used to store signed requests between all components to assure non-repudiation from a request by an application component. At last, the smart card can secure generic terminals **[JG99]**, which can be used to start or continue an application execution.

Mobility

As evoked previously, the application user is nomadic, and is connected on several kinds of terminals that can be mobile (for example, mobile phone) or generic (for example, a terminal in a conference site).

This kind of terminal does not allow to the user to be connected on the network for a long time. The client part of the distributed application will not be connected to the other application components during all the application execution time.

In this case, when the user restarts the client part of the distributed application, the new terminal must access to the application context and the transaction identifier. These data can not be stored in the first terminal, which can be off-line at restart time. However, these data can be stored in a smart card in a persistent way, and will be used in the new terminal to restore the application context at the reconnection.

Availability

In many cases, one or more application components are unreachable when the end-user want to run or continue an application. The application designer has to define how the application works when a component is absent from the running environment. This application design is already used by payment application with a bankcard where two execution contexts are defined. In the Online context, the vendor contacts the client bank to ask for an acknowledgement to debit the bank account. In the Offline context, only the card and the terminal are involved.

In this case, a smart card must be able to embed the application components configuration, and the condition to respect in order to use the components (as a security device, the smart card has to verify that all the components are used in respect with the application design). Moreover, while Offline, the smart card can embed a part of application code and data which can be downloaded by all the terminals in order to continue the application execution.

3 Smart card typical interaction models

Regarding to the distributed systems problematic as well as the smart card technological evolution, we think that it is benefic to place smart card in the center of applications. After getting a glance at the two common smart card interaction models, we will present a new one, justifying his advantages through the example of a distributed transaction manager smart card.

3.1 Smart card as a server

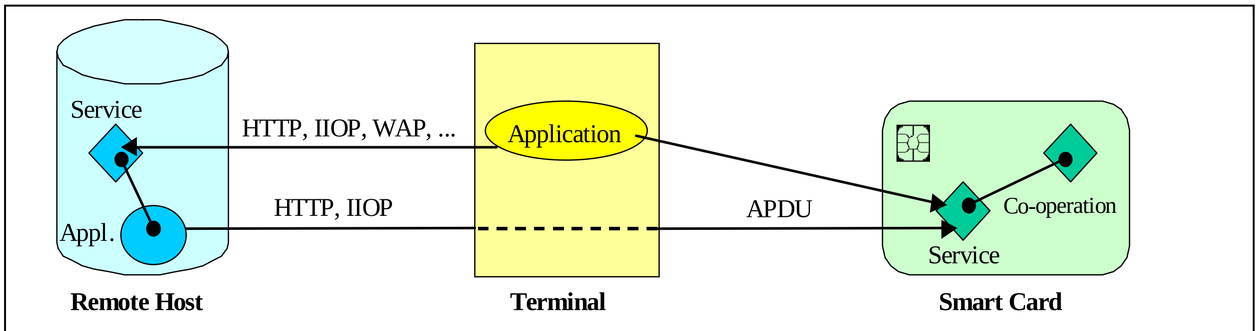

Figure 1. Smart card as server

The Figure 1 presents the traditional interaction model of the smart card. The arrows define the way the requests are sent. We consider the application as the "main", i.e. it controls the execution flow. A service is often be consider a server, but it can be also client of another service (it then seems like CORBA Common Object Services **[OMG99]**).

The smart card proposes here only services and the application is generally located on the terminal or on a remote host. The smart card can be server of data, as the CQL database smart card **[GG92]**, or server of code like Java Card is **[Sun99]**.

The dialogue between the terminal and the smart card is carried out via the normalized APDU protocol **[ISO87]**. This protocol is one-way as the terminal requests the card who computes and eventually gives a result, so server is the first and natural use of smart cards.

If the client is a remote host, the communication with the smart card is carried out through the terminal that is then considered as the smart card proxy. In **[B99]**, the card is seen like an embedded Web server and the "pages" are accessed by the way of the terminal using HTTP.

3.2 Proactive smart card

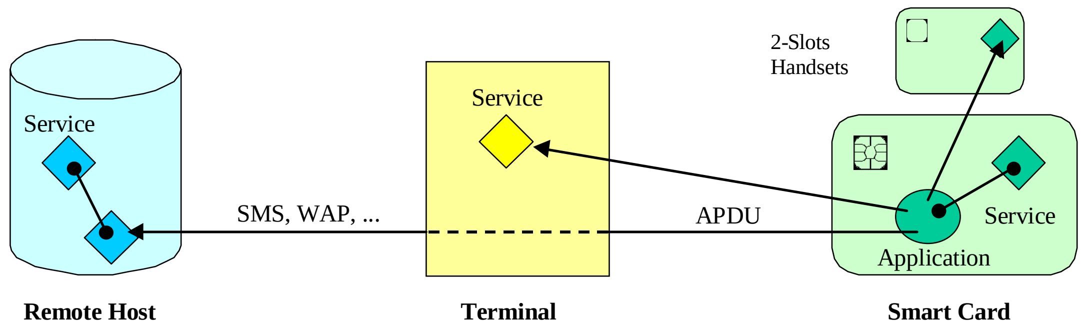

Figure 2. Pro-active smart card

The Proactive smart card model, presented on Figure 2, was proposed within the GSM **[ETSI97]** services extension framework with SIM Toolkit **[ETSI98]**. Compared to the previous approach, the roles are completely reversed. The smart card holds the application and is client of the services hosted by either terminal or remote servers.

This model is used in mobile phone in order to let the SIM card driving the Man-Machine Interface of the terminal (for example the smart card sends to the terminal the order "display this menu and return the number of the selected item). The smart card can also request remote services using SMS (Short Messages Service), or, only in 2-slots handset devices, requests services from another smart card. In the last two examples presented, the terminal is considered by the smart card as a proxy. In this model, even if the card seems to be able to send requests, the basic communication protocol is however not modified. It is only subtly circumvented in order to let the smart card give orders.

4 Toward a new model : Interactive smart card

4.1 A new interaction model

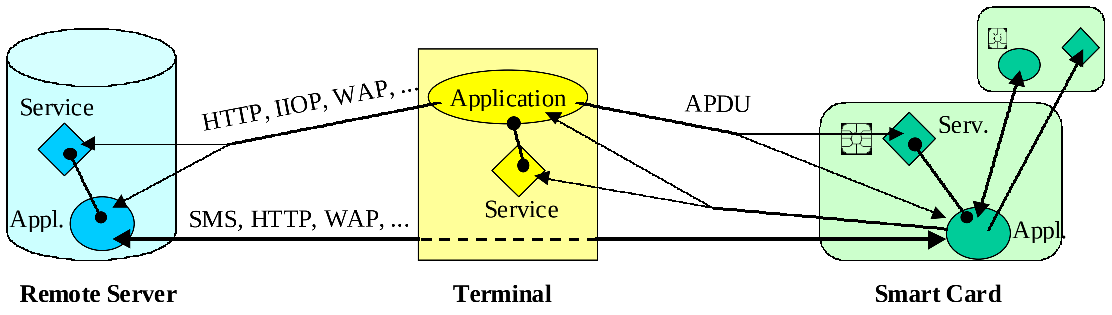

Figure 3. Interactive smart card model

Several reasons lead us to propose a completely interactive model for smart cards, such as presented on Figure 3. Multi-applications smart cards (as Java Card) raise the question "why is it necessary to execute services in a smart card?" whose major answer is the need of security. Indeed, potentially non-secure terminals cannot reasonably host kinds of sensible applications. In addition, it is also interesting to use the intrinsic security of the smart card so as to protect the code so that it cannot be examined during execution.

The potential security hole in the terminal (e.g. a Trojan Horse) justifies intending to give to the smart card the control of the application. Moreover, the characteristics of smart cards make of them a minimal core of security and confidence. If one wishes to benefit from this function, it is also necessary to give smart cards the capacity of initiative.

4.2 Use case : smart card as a distributed transaction manager

Let us consider the example of a card who is a distributed transaction manager **[BN97]**, of which the interest is discussed in **[L98]**, used to make a combined commercial transaction on Internet (the user wishes to obtain both or nothing of 2 products from 2 different servers). The application, a virtual gallery browser including shared shopping cart, can be on the terminal or on a remote host (to increase ergonomy while using lightweight terminals like cellular phones). When the user wishes to validate his purchase, a request is sent to the card (then server) in order to start the distributed validation. The embedded transaction manager (then client) then sends a validation order to the various servers implied in the transaction.

The previous scenario shows how distributed information systems can take benefits of security and trust brought by smart cards, but in such case the hosts and the smart card need to be both client and servers.

4.3 Interactive model implementation

In order to implement this model, some problems have to be solved. First, a full-duplex communication protocol has to be defined. If not, the complete interactivity cannot be provided. Even if the classical normalized APDU protocol have been modified while designing SIM Toolkit, such a way is not efficient. A glance at the model presented previously raises the need of an asynchronous protocol in order to be able to provide necessary event notification mechanisms. A second problem is at a higher level. The smart card is characterized by its intrinsic security, particularly the fact that it is a highly secured data container. Let us now consider that an embedded application can simply send a request to a remote host via a system call provided by the way of an API. Then there's no easy way to control that this application do not send its embedded data to this host, and this problem is bigger when several cardlets share data. We also propose low level mechanisms, derived from another research **[DJL],** in order to provide fine-grain access control able to reduce those risks.

5 Conclusion

In the paper, we have presented new application requirements to trust software components in the WWW application involving several partners. This requires security features when the software components are deployed between the partners. The application must be deployed and redeployed at runtime since the end-user can change his terminal (e.g. a PC) for another with different features (e.g. a mobile phone). This requires mobility and adaptability features when the end-user roams from one terminal to another. We believe smart cards can be used to meet these new requirements. However in the actual execution model, smart cards acts as isolated server. The pro-active execution embeds a pilot of application in the card but the protocol between the card and the terminal is static. We propose to improve this execution model by the interactive execution which allows the card to simultaneously pilot the application as a client and to serve some external requests as a server. With the interactive model, distributed applications can take benefits of smart card intrinsic security. Like in the transaction coordinator example, this allow to use the card as a minimal security kernel. However, in order to keep the high security level of embedded data, some mechanisms have to be provided in the smart card operating system.

6 Bibliography

[B99] J. Barber. *The smart card URL programming interface*. in Proceedings Gemplus Developper Conference '99.(Paris, France). 1999.

[CD97] D. Carlier, D. Donsez. *Permanent Network Representation for Mobile User*. OPODIS, International Conference on Principles Of Distributed Systems, France. 1997.

[DJL] D. Donsez, S. Jean, S. Lecomte. *How to improve access control and application flexibility in smart card using the database principles.* ACM SIGMOD PODS'2000 submission.

[ETSI98] European Telecommunications Standards Institute (ETSI). *Digital cellular telecomm. system (Phase 2+); Specification of the SIM Application Toolkit for the Subscriber Identity Module - Mobile Equipment (SIM - ME)*. Technical specification. 1998.

[GG92] G. Grimonprez, E. Gordons. *A card as element of a distributed database.* IFIP WG 8.4 Workshop, Ottawa. 1992.

[ISO87] International Standard Organization (ISO). *Information Technology - Identification cards – Integrated circuit(s) cards with contacts*. ISO/IEC 7816-1,2,3,4. 1987-1995.

[L98] S. Lecomte. *COST-STIC : Carte Orientée Service Transactionnel et Systèmes Transactionnels Intégrant des Cartes*. PhD thesis, University of Lille. 1998.

[NY97] Y. Nakamura and G. Yamamoto. *An Electronic Marketplace Framework Based on Mobile Agents*. IBM Research, Tokyo Research Laboratory, Research Report, RT-0224. 1997.

[Sun99] Sun Microsystems Inc. *Java Card 2.1. Runtime Environment (JCRE) Specification*. 1999.

[BN97] P.A. Bernstein, E. Newcomer. *Principles of Transaction Processing for the systems professional*. M. Kaufmann Publishers. 1997.

[ETSI97] European Telecommunications Standards Institute (ETSI). *GSM 11.11: Digital cellular telecommunications system (Phase 2+); Specification of the Subscriber Identity Module - Mobile Equipment (SIM - ME) interface*. Technical specification. 1997.

[W93] M. Weiser, *Some computer science issues in ubiquitous computing* .Communications of the ACM 36,7 , p75-84. 1993.

[OMG99] OBJECT MANAGEMENT GROUP, *Common Object Request Broker Architecture - revision 2.3.1,* OMG document formal/99-10-07. 1999.