Benefits from realizing a location-based mobile hotel service with Java Card^{TM} 3.0

Rebekka Oeters*
University of Paderborn,
Software Quality Lab
Warbuger Str. 100
33098 Paderborn, Germany
roeters@s-lab.upb.de

Uwe Kastens
University of Paderborn,
Software Quality Lab
Fürstenallee 11
33102 Paderborn, Germany
uwe@upb.de

Carsten Rust
Sagem Orga GmbH
Heinz-Nixdorf-Ring 1
33106 Paderborn, Germany
carsten.rust@sagem-orga.com

Lars Schnake
Sagem Orga GmbH
Heinz-Nixdorf-Ring 1
33106 Paderborn, Germany
lars.schnake@sagem-orga.com

ABSTRACT

The leverage of Java Card^{TM} 3.0 enables a more comfortable development of more sophisticated and secure web applications on smart cards. Based on our experiences with the realization of a location-based mobile protoype hotel service with Java Card^{TM} 3.0 we highlight the benefits arising through the use of Java Card^{TM} 3.0 on mobile devices for Java Card platform and basic service providers as well as for other service providers in this paper.

Categories and Subject Descriptors

D.2 [**SOFTWARE ENGINEERING**]: Software development

General Terms

Languages, Standardization, Security

Keywords

Java Card^{TM} 3.0, mobile service, location-based service

1. INTRODUCTION

Java Card^{TM} 3.0 [1] (JC 3.0) is the upcoming Java Card^{TM} standard recently specified by Sun Microsystems and the Java Card Forum [4]. The following highlights of JC 3.0 will be demonstrated by a location-based mobile hotel service (see subsection 2.3) that we have prototypically implemented in order to gain first experiences with JC 3.0 which we will employ on our new JC 3.0 smart cards:

- **Internet connectivity of smart cards:** A web server and a servlet container will be available on JC 3.0 smart cards in order to serve multiple client requests, received over HTTP via TCP/IP, at a time by delivering static or dynamic content (see subsection 2.2).
- **Standardized mobile services:** Mobile services can be developed by service providers and deployed within servlet containers of multiple vendors in the form of standardized web applications (see subsections 2.2 and 3.1).
- **Concurrency in Java for smart cards**: Programming concurrent JC 3.0 applications enables a novel development of highly responsive web applications and a multi-threaded web server and servlet container (see subsection 3.1).
- **Mashup of data:** The mashup [5] of personal and trusted data stored on smart cards with large amounts of content requested from a remote server through the use of AJAX [5] enables an excellent use of smart cards as trusted storage media, and remote storage servers as content providers for the realization of dynamic and highly responsive web applications (see subsection 3.2)
- **Security enabler and complier:** Both the web server and the web applications deployed in the web server will serve and comply respectively security standards (see subsections 2.2 and 3.3)

The features and the underlying concepts of the location-based mobile hotel service are introduced in section 2. The features and the functionalities of JC 3.0 easing the service's development and distribution are presented in section 3. We conclude with the benefits we envision for our company as a Java Card platform and basic service provider as well as for other service providers arising through the use of JC 3.0 on our new JC 3.0 smart cards in section 4.

2. USING SMART CARDS FOR MOBILE SERVICES

Smart cards are well established in the mobile network industry. The SIM (Subscriber Identity Module) was primary defined in 1988 as a smart card for user authentication. Over the years the SIM has evolved, hardware and software has

*The work described in this paper has been done in cooperation between the smart card company Sagem Orga GmbH and the Software Quality Lab (s-lab) of the University of Paderborn.

become more powerful and the memory size has increased. Today's SIM-Cards do not only form the security token for network authentication and secure data storage for parameters and personal user data like the phonebook. It is also utilized by network operators as a service platform enabling differentiation and the provision of value added services.

The introduction of Java Card^{TM} technology by Sun Microsystems in the late 90s has also become very popular in the mobile sector. The usage of Java supports the software design with modularity and security concepts and brings a popular programming language to the smart card. The European Telecommunications Standards Institute (ETSI) [3] exploits the new technology of smart cards for the mobile sector and defines many useful technologies for the SIM-Card device including an application programming interface to make the platform extensible with further applications.

After describing the evolution of mobile services before JC 3.0 in subsection 2.1, the features and the underlying concepts of mobile services in JC 3.0 are introduced in subsection 2.2 and exemplified by a location-based hotel service in subsection 2.3.

2.1 Mobile services before JC 3.0

Especially the standardization body ETSI has defined several applications and techniques to enhance mobile communication with security or convenience features based on card technology (e.g. roaming). One frequently used ability is the SIM-Card Application Toolkit. It defines a set of commands and procedures to enable the card to contain applications specific to the issuer (the network operator). This allows the operator to deploy services including information and location-based services, banking and Internet access. Today, the smart card in a mobile device operates and manipulates menus and services and authenticates users for service access. Even applications for secure financial transactions can be managed via the tamper-proof SIM device, enabling mobile commerce.

Unfortunately, the graphical user interaction falls behind on today's standards. Interaction with the successful application platform J2ME on mobile devices proves difficult, even if both worlds are Java dominated.

2.2 Mobile services in JC 3.0

With the advent of JC 3.0 mobile services including location-based mobile services can be developed and distributed much more comfortably and efficiently than in the past. We have drawn this conclusion from our first experiences with JC 3.0 and will give support for it in section 3.

Mobile services are realized as JC 3.0 web applications and deployed in a JC 3.0 web server and servlet container. A web application is a collection of servlets, static content like HTML pages, and other classes and resources constituting a complete application or service. The web application runs in a servlet container and serves client requests or other applications and services (see figure 1). A servlet is a Java class interacting with web clients via a request / response paradigm and generating dynamic content. A servlet obeys a specified life cycle and is managed by a servlet container.

The JC 3.0 servlet specification defines the interface, the configuration, and the deployment format of web applications on the card.

The interface of a web application with respect to web clients requesting a static or a dynamic resource is specified by the service method of the web application's servlets, and the URI mappings for these servlets used to make the servlets accessible by appropriately built client requests. The interface of a web application with respect to other web applications residing on the same smart card is specified by explicitely listing objects accessible from the other applications.

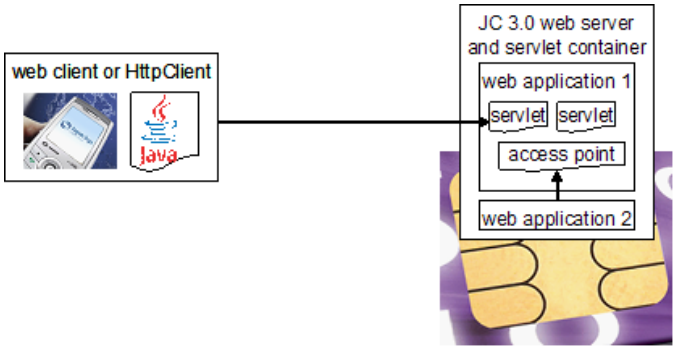

Figure 1: High level overview of the JC 3.0 web server's architecture

The configuration of a web application is declared by the web application's deployment descriptor, the Java Card platform specific application descriptor, and the runtime descriptor. The deployment descriptor of a web application lists the elements of the web application and configures the URI mapping used to access the servlets of the web application. Moreover, the session management available within the servlet container in order to track clients' information during several requests is configured within the deployment descriptor. The deployment descriptor also configures the security constraints of web application components with regard to the authentication used for web clients and with regard to the data transport, and defines the security roles of users as used for authentication.

The optional Java Card platform specific application descriptor and the runtime descriptor can be used to specify the access control for other web applications e.g. the access points, among other things (see figure 1). Thus, besides configuring a web application's components, the three descriptors listed above can also be used to configure the web application's authentication mechanism and access control.

2.3 Hotel service

We prototypically realized a hotel service that provides a mobile user, e.g. a businesswoman looking for a hotel to stay in, with a map of hotels dependent on her location, as well as her ratings or notices for the hotels.

After requesting the hotel service (see screenshot (a) in figure 2), the mobile's user must authenticate in order to gain access to her personal ratings or notices of the hotels close to her (see screenshot (b) in figure 2). Now, the mobile's user can request her personal data concurrently for selected hotels (see screenshot (c) in figure 2).

The location-based mobile hotel service mashes up large amounts of map data requested from a map server and small trusted items of personal data requested from the mobile user's smart cards (see figure 3).

The location-based mobile hotel service has the following structure:

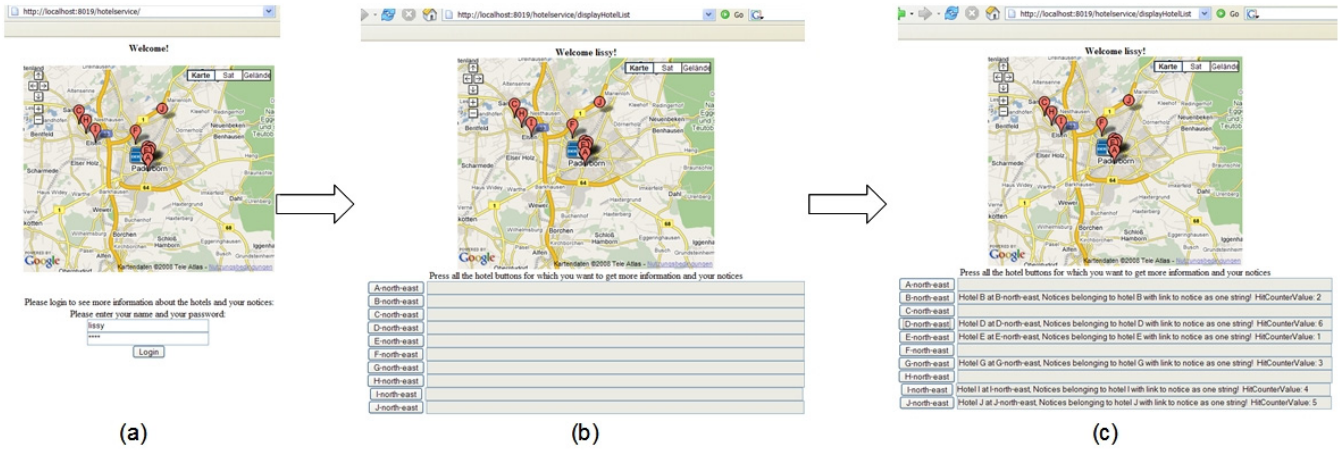

Figure 2: Behavior of the hotel service

Figure 3: Mashup of location-based map request and trusted personal data

On the client side, the mobile's web browser, the location-based mobile hotel service consists of a static HTML page used to display the result of the map request and to enable login. Moreover, two JavaScipt functions executed by the web browser's AJAX engine reside on the client side (see subsection 3.2).

On the server side, the JC 3.0 web server and servlet container, the location-based mobile hotel service consists of JC 3.0 servlets (see subclasses of class HttpServlet in the top-most segment of figure 4) which constitute the service's public interface and implement the service's request handling logic like Java Card^{TM} 2.2 (JC 2.2) applets. JC 3.0 web application specific initialization behavior and request handling logic is implemented within the filter and listener classes (see subclasses of class Filter and classes implementing the listener interfaces in the middle segment of figure 4). The persistent data that models of the hotel service in shown in the bottom segment of figure 4.

3. JC 3.0 FEATURES AND FUNCTIONALITIES EASING THE SERVICE'S REALIZATION

The features and functionalities of JC 3.0 that ease the development and the distribution of mobiles services on new JC 3.0 smart cards in our opinion are highlighted in this section. The standardized and concurrent JC 3.0 API (see subsection 3.1), the mashup of trusted personal and large amounts of freely available data (see subsection 3.2), and the security standards (see subsection 3.3) introduced with JC 3.0 are the main novelties, we have beheld when prototypically implementing our location-based mobile hotel service.

3.1 Standardized and concurrent JC 3.0 API

Both the JC 3.0 web server and the web applications deployed within the servlet container are Standard Java like applications with high programming comfort due to the well-established and rich programming concepts of Standard Java e.g. concurrency, the use of the Java String API, and more liberal use of objects, when compared with JC 2.2 [2].

The adoption of the well-known servlet and web application model from Standard Java within the JC 3.0 API eases the acquisition and development of mobile services. At the same time the concept of servlets is easy to grasp for todays JC 2.2 developers because of the similarity with JC 2.2 applets.

Due to the standardized web application archive (WAR) format the distribution of mobile services is simplified.

3.2 Mashup of location-based service in mobile browser

Within JC 3.0 web applications, data requested from remote servers can be mashed up with trusted data requested from mobile users' smart cards (see figure 3). Thus, complex mobile services can be composed using JC 3.0 while the mobile user's personal data is further protected by the

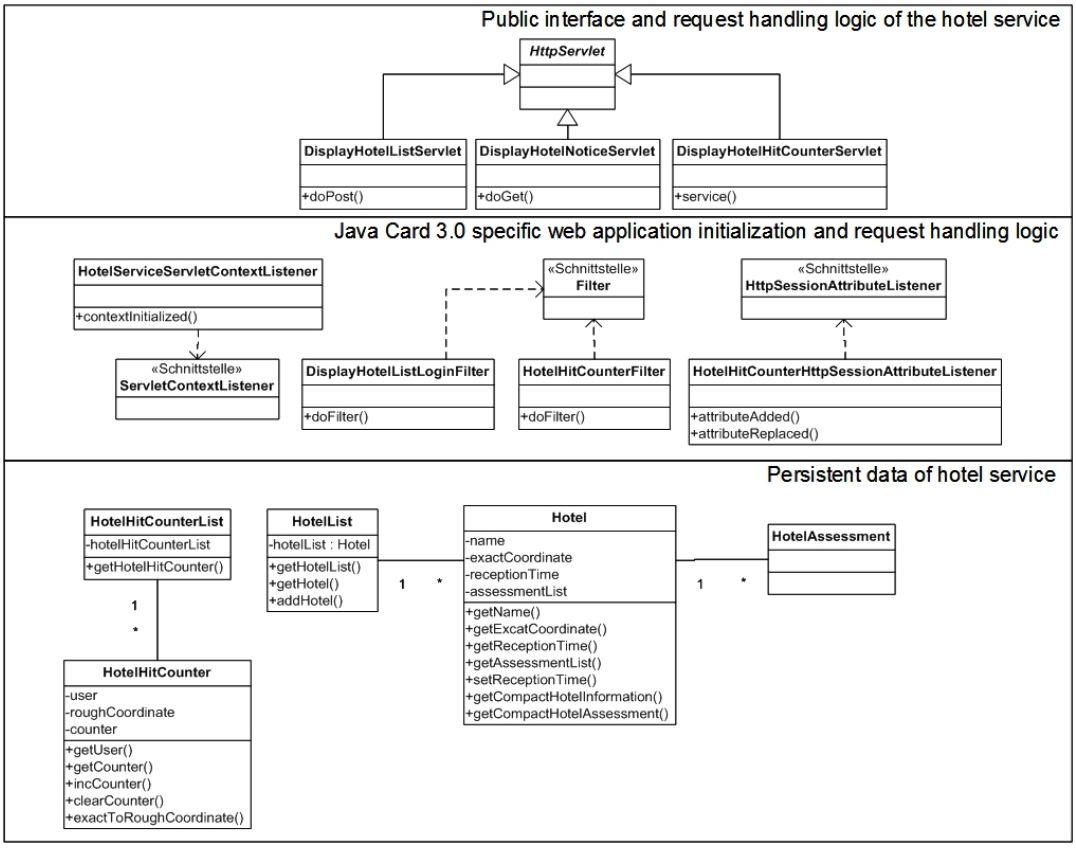

Figure 4: Structure of the hotel service

smart card and the web server's and web application's security mechanisms.

The injection of the mobile user's personal data into a complex mobile service is accomplished by the AJAX (Asynchronous JavaScript and XML 5) engine of the mobile device's web browser (see figure 5).

The use of AJAX and the concurrent processing of user requests within JC 3.0 web applications and the JC 3.0 web server lead to the highly responsive behavior of these mobile services.

3.3 Web server as security enabler

The JC 3.0 web server and servlet container provides the network services over which client requests and server responses are sent, and manages servlets throughout their life cycle. The JC 3.0 web server allows concurrent handling of multiple client requests at a time due to the multithreading capabilities of JC 3.0 virtual machines. Thus, mobile services can be accessed in parallel, e.g. a user of a mobile device and a service provider can access a web application at the same time.

The JC 3.0 web server implements the concurrent handling (see interface ThreadPool in figure 6) of multiple client requests (see classes HttpConnection, Request, and Response in figure 6) by dispatching requests to handlers for static resources, servlets, web applications, sessions, and security constraints (see all classes implementing the interface Handler). These handlers generate responses which are returned to the clients.

The JC 3.0 API provides further security implementations beyond the authentication and access control mechanisms mentioned in subsection 2.2 and shown in figure 6. These further security implementations e.g. protection domains and code isolation, can be used by service providers when developing mobile services and deploying them in the form of web applications within JC 3.0.

4. CONCLUSIONS

We have presented a prototype of a location-based JC 3.0 mobile hotel service and used the experiences we made during its realization in order to highlight the features and functionalities of JC 3.0, Sun's upcoming Java Card^{TM} standard. Moreover we have pointed out the benefits we envision through the development and distribution of JC 3.0 mobile services for Java Card platform and basic service providers as well as for other service providers, namely:

- **Independent development** of mobile services done by service providers **and deployment of mobile services** within every servlet container that satisfy the new architecture for mobile services due to the standardized interface, configuration, and deployment format of web applications (see subsection 2.2).
- **Provision of standardized security concepts** by every servlet container that satisfy the new architecture for mobile services which enables standardized usage of security concepts within web applications for service providers (see subsections 2.2 and 3.3).

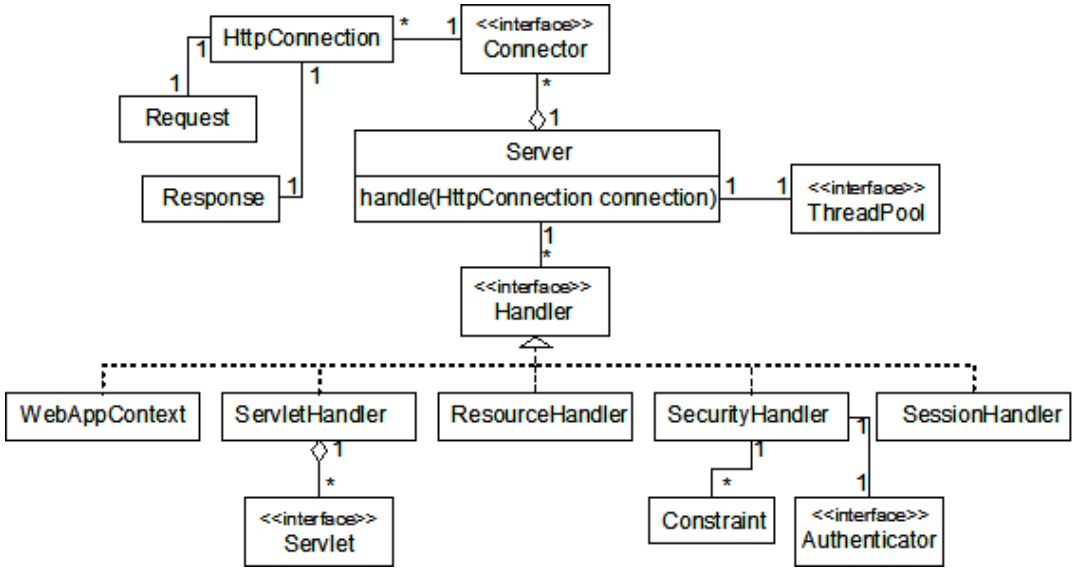

Figure 6: High level overview of the JC 3.0 web server's architecture

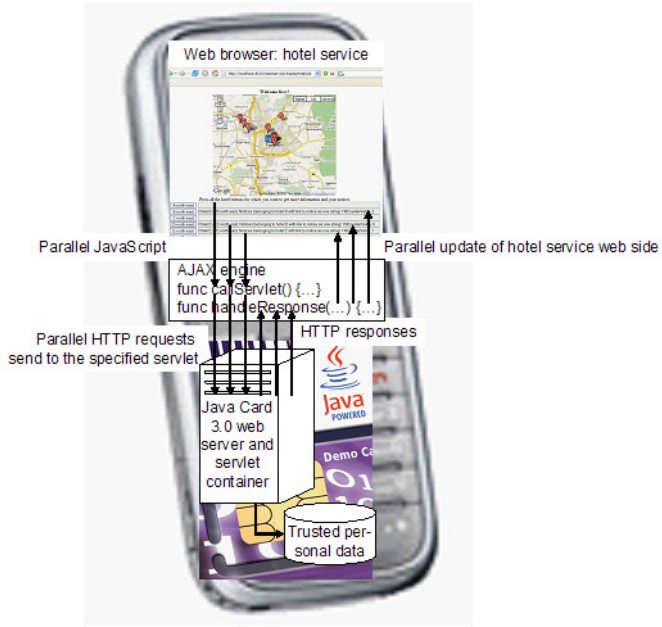

Figure 5: AJAX used for dynamic update of parts of the hotel service web side through injection of personal data

- **More comfortable and efficient development and distribution of JC 3.0 web applications** for service providers due to well-known, rich, and Standard Java like programming concepts comprised in JC 3.0 and the standardized distribution format specified in JC 3.0 (see subsection 2.2).

- **Highly responsive and dynamic JC 3.0 web applications** due to concurrent accesses of JC 3.0 web applications and the JC 3.0 web server (see subsections 2.2, 3.1, and 3.3).

- **Excellent use of smart card capabilities as trusted, resource-limited storage** media within the context of JC 3.0 web applications for smart card platform and service providers (see subsection 3.2).

5. REFERENCES

[1] http://java.sun.com/products/javacard/3.0/specs.jsp. Java card 3.0. 10.04.2008.

[2] http://java.sun.com/products/javacard/specs.html. Java card 2.2. 10.04.2008.

[3] http://www.etsi.org. Etsi. 24.04.2008.

[4] http://www.javacardforum.org. Java card forum. 10.04.2008.

[5] L. Lagosanto and J.-J. Vandewalle. Web 2.0 applications on a next-generation java card platform. Technical report, JavaOne 2007, TS-5203, Gemalto, 2007.