SmartRestaurant
A Report on the Development of a NFC-Based Mobile Application

Vanda Rosa, Isabel Brito and João Paulo Barros
Instituto Politécnico de Beja, Escola Superior de Tecnologia e Gestão, Beja, Portugal

Keywords: Requirements, Mobile.

Abstract: This paper presents as a case study our experience gathered along the development of a mobile application prototype to be used by restaurant clients. The application, named SmartRestaurant, uses Near Field Communication (NFC) connectivity and runs on Android. It can be used in restaurants to load its menu, and other information provided by the restaurant, through scanning a NFC tag with a mobile phone or other NFC enabled device. Additionally, the application should allow users to store and share data with others. The paper presents the lessons learned during the development process used in this case study, including the functionalities already implemented, the future work, and also the process successes and failures.

1 INTRODUCTION

In this paper we present our experience on the development of a mobile native application prototype for restaurants, named SmartRestaurant. This application uses Near Field Communication (NFC), which is a set of short-range wireless technologies, typically requiring a distance of four centimetres or less to initiate a connection(Android, 2012). It makes life easier and more convenient for consumers by simplifying transactions and the exchange of digital content through the connection of electronic devices with a touch (NFC, 2013). NFC allows the communication of small amounts of data between a NFC tag and a NFC enabled device, or between two NFC-enabled devices. The driving goal behind this prototype is to take advantage of NFC, namely to allow restaurant clients to share data in a useful way. Presently, it is an Android application for smartphones and tablets. To illustrate the application, information about a Portuguese restaurant is used. The restaurant has to provide its information, namely recipes, menus, and suggestions. This information is loaded into the smartphone or tablet and can be shared with other users.

This paper is composed as follows: Section 2 presents some background and motivation. Section 3 shows the main technologies, languages, and tools used in the mobile application development, Section 4 presents the application prototype requirements, design, programming and testing stages, Section 5 identifies successes and failures and Section 6 the related work. Finally, Section 7 presents the conclusions and future work.

2 BACKGROUND

Since the AppStore store opened in 2008, there was been a huge growth in mobile application development, thus promoting new development practices for mobile application (e.g. (Wasserman, 2010; Morris et al., 2010; Alencar and Cowan, 2011)). Mobile applications development present specific aspects that may differentiate it from other applications, such as mobility and pervasiveness, sensor-based input, proximity to users including handling of personal data, potential interaction with other applications, and the large number of mobile devices and operating systems.

According with recent experience (e.g. (Wasserman, 2010)), mobile applications development frequently apply the following points:

- Use agile techniques especially when the application is relatively small, with two or three developers.
- Use the best practices in order to organising tracking of development efforts.
- Use specific development tools and framework, such as ADT for eclipse (ADT, 2013) or XCode for iOS (Xcode, 2013b), to simplify the task of development a small or medium size application.

Rosa V., Brito I. and Barros J..
SmartRestaurant - A Report on the Development of a NFC-Based Mobile Application.
DOI: 10.5220/0004567203220327
In *Proceedings of the 8th International Joint Conference on Software Technologies* (ICSOFT-EA-2013), pages 322-327
ISBN: 978-989-8565-68-6

- Make the best possible use of limited screen space and help the mobile users to quickly complete a simple task.
- Develop tests using testing platform, e. g. emulators, and real platforms, (e. g. (Xcode, 2013a)).

The above list is not exhaustive, but indicates the most important points for our study.

3 TECHNOLOGIES, LANGUAGES, AND TOOLS

This section presents the technologies, languages, and tools used to develop the prototype.

3.1 NFC

NFC is a wireless communication technology that allows data sharing between mobile devices in a close proximity, usually no more than a few centimeters away. The NFC standard includes two communication modes between mobile devices:

1. Active communication mode;
2. Passive communication mode.

In the active mode, the device is able to start a communication, in two specific ways: (1) the mobile device can read and even write NFC tags (reader mode) or (2) two devices can communicate and share data (peer-to-peer mode).

In the passive mode (or card emulation), the device is a contactless card and can only be read by other NFC readers (Mednieks et al., 2012).

In the present case study, we use both NFC communication modes between mobile devices. The NFC tags, used to store application and restaurant information, are in passive mode. The active mode, present in smartphones and tablets with NFC technology, can be used to read tags or share data with other active NFC devices.

3.2 Languages and Tools

To develop the Android application prototype we have used the most common tools, including the Java programming language: namely, the Integrated Development Environment (IDE) used to build, test, and debug the application was Eclipse with the Android Developer Tools (ADT) plugin to streamline Android development. In the Eclipse installation, we include some Android SDK tools, Android platforms tools, and the emulator AVD Manager (Android Virtual Devices) to develop and test the SmartRestaurant prototype application (DevTools, 2012). For testing the application, we also used NFC tags (NFC forum, 2012) and a smartphone with NFC technology.

In this sense, we have developed Java code that writes application and restaurant information in a NFC tag. This information, which includes the application package name and, for example, restaurant ID or even table ID, can then be read by an active NFC enabled device to load SmartRestaurant application data. In this NFC Data Exchange Format (NDEF) message (Nokia, 2013), we add an Android Application Record (AAR), which has the package name embedded inside an NDEF record, to make sure that SmartRestaurant application is started when the NFC tag is scanned. If SmartRestaurant is not present on the mobile device, Google Play is launched to download the application (ADev, 2012).

To test the application we used the AVD Manager, the Android system virtual device emulator, but we also used a mobile phone, an Android-powered device that supports NFC connectivity.

4 SmartRestaurant PROTOTYPE – DEVELOPMENT ISSUES

The SmartRestaurant prototype development process was inspired by the mobile App Development model in (Morris et al., 2010). This model recommends agile development initiatives and the use of prototyping and testing activities early in the development process. The model is composed of three phases:

1. Feasibility and economic efficient analysis;
2. Software product realization;
3. Distribution.

This paper shows the application of the second phase, which was accomplished by three developers in collaboration with the project manager from Smartobject company. The project manager was responsible for the first and third phases. The selected case study can be applied in the context of any common restaurant. Next, following the stages of software product realisation, we present the requirements, design, programming and testing stages.

4.1 The SmartRestaurant Requirements

The requirements are collected and maintained by early prototypes taking in consideration the project

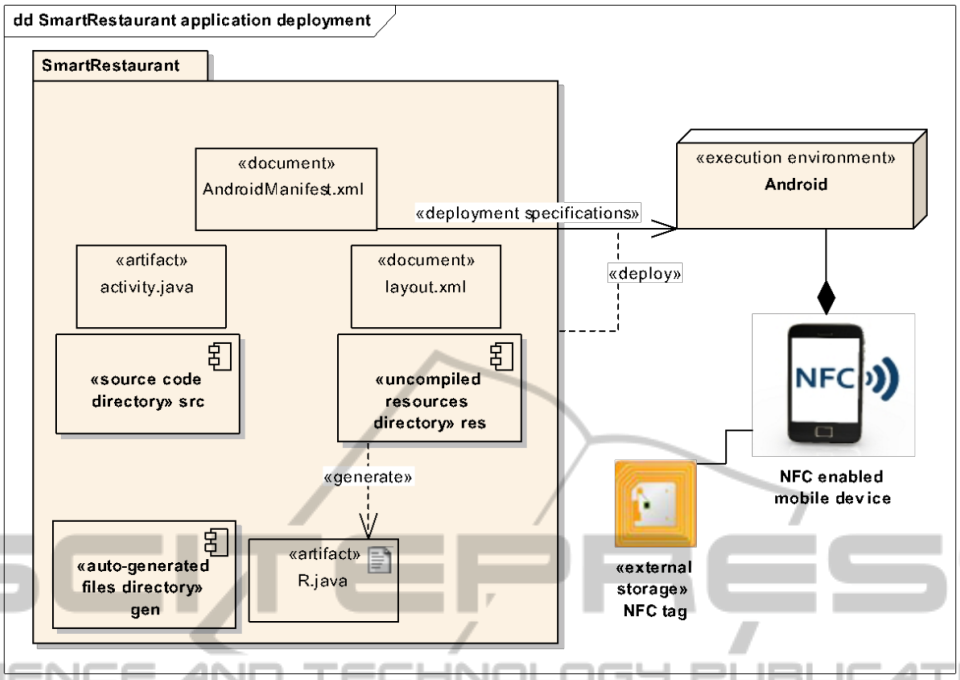

Figure 1: Deployment.

manager point of view. The requirements review is the responsibility of the project manager in collaboration with restaurants owners and managers. The application first requirement is to read a NFC tag to identify the restaurant and other provided information. When the tag is scanned, if the application is not installed on the mobile device, the same tag provides the information to download the SmartRestaurant application. If the application is already present on the device, restaurant data is loaded.

The restaurant data is collected and classified as restaurant information, menu, multimedia gallery, ingredient, satisfaction, and favourites. In the multimedia gallery, the restaurant can provide images of dishes or ingredients. An option is available for customers reviews/satisfaction about the restaurant service or dishes. Clients can also save their favourite dishes and share this information with other NFC enabled device users.

Inside the restaurant, NFC should be used to identify the customer table so that the orders (and later the bill) are delivered to the respective place.

4.2 SmartRestaurant Design

In Android application development, each layout (screen design) is usually related to an activity, i.e., the activity is the source code behind the user interface layout and defines its behaviour. An activity can also be seen as an application component that provides a single screen that the user sees on the device at one time and with which he can interact in order to do something. Just like a website consists of multiple pages, so does an Android application typically has multiple activities between which the user navigates (Gargenta, 2011).

The SmartRestaurant application design mostly uses linear, relative, list view, and grid view based layouts. List view layout is used in many screens to display menu options as a list of scrollable items and grid view layout is used in the image gallery.

Each activity corresponds to a Java file and the layout to an XML file. The correspondence between them is declared in the AndroidManifest.xml file. In other words, the Manifest File is the ”file that explains what the application consists of, what all its main building blocks are, what permissions it requires” (Gargenta, 2011), as well as other specifications about the application structure. Another important file in the SmartRestaurant design is the R.java. This is an auto-generated file that organises the res directory contents, such as images, layouts, formatting styles, strings, arrays, and other XML files added to that resource directory. As a database system has not yet been added, this file assumes especial importance to organise application data sample content.

As UML deployment diagrams can illustrate how the hardware and software components work together, in Figure 1 we use this type of diagram to illustrate the SmartRestaurant architecture (Ambler, 2012).

Figure 1 shows SmartRestaurant main composition, namely the deployment specification (XML files), directories for source code, uncompiled resources and auto-generated files. There is a node that represents the execution environment, in this case Android (UML, 2013). The SmartRestaurant application starts when the mobile phone (with NFC enabled and the SmartRestaurant app) touches a restaurant object containing a NFC tag.

4.3 SmartRestaurant Prototype Programming

The SmartRestaurant application interface is simple, user friendly, and respects the usability requirements. For example, we had special concern about colours, typography, writing style, iconography and other images, main and contextual navigation buttons, and a hierarchical navigation system to provide a clear and easy user interaction. This is shown in Figure 2.

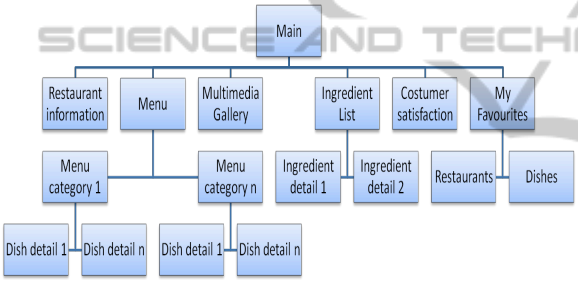

Figure 2: Navigation diagram.

The application has a main layout, which can be seen in Figure 3. The user cannot change this layout.

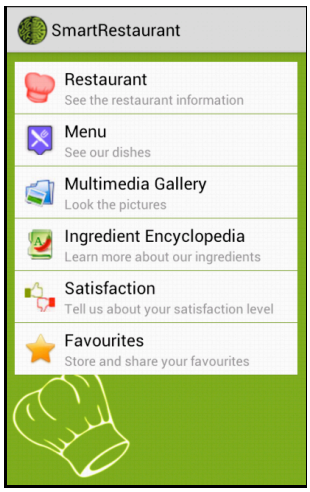

Figure 3: Main layout for SmartRestaurant.

However, all the remaining layouts should be customisable, since the available information depends on the specific restaurant data. The restaurant can provide information similar to its traditional menu or adapt the information to the mobile application. The available information should be attractive and suitable for a mobile application, so it is important to provide images and more complete information about the dishes and ingredients, as exemplified in Figures 4 and 5.

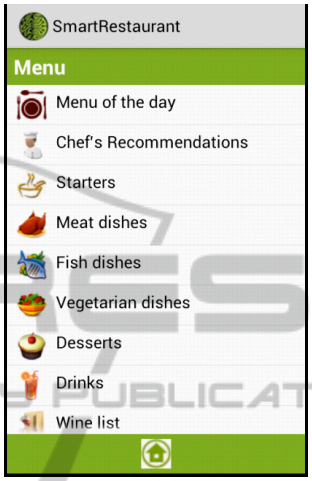

Figure 4: Menu layout example.

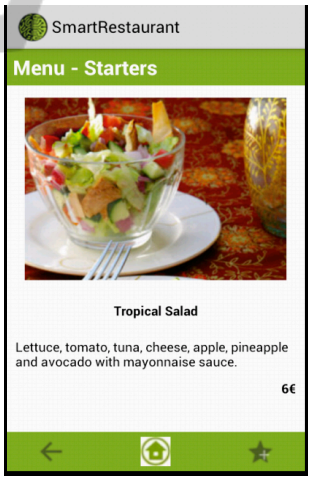

Figure 5: Dish layout example.

As stated before, SmartRestaurant has been installed on a tablet and the application was perfectly usable. Yet, for a perfect adaptation to the screen size, it would be necessary to make some changes. In fact, it is important to design an Android application to be compatible with different device types. Each screen size offers different possibilities and challenges for user interaction. As such, the application must be optimised for multiple screen configurations (Screens, 2012).

4.4 SmartRestaurant Prototype Testing

This stage was accomplished using a test platform, the AVD Manager, which is the Android system virtual device emulator, and an Android-powered device, which supports NFC connectivity, as a real platform, to test NFC tags and to evaluate the application design and functionality.

Also, as already stated, we have installed the application in a tablet, yet only for testing different screen sizes, as this was not a NFC enabled device. The real platform test was accomplished with the collaboration of the project manager, who identified the users and obtained their feedback.

5 SUCCESSES AND FAILURES

Along the development of the SmartRestaurant prototype the following tasks have already been conducted:

- Application prototype requirements, architecture and design.
- Source code development (main features only).
- Data sample writing.
- NDEF message writing in a NFC tag.
- Application installation in an android smartphone, which supports NFC connectivity, and application testing.

These tasks have allowed us to identify several successes and failures. This is also a demonstration that a functional prototype is a powerful vehicle to improve initial major goals and software requirements.

Regarding successes, we were already able to identify the following points as positive and important ones:

- For the identification of the initial general goals it is of utmost importance to gather the opinion, formally or informally, of many different users.
- Mobile applications open many more possibilities than traditional ones, due to the additional available sensors and the inherent anytime and anywhere usage possibility . Hence, users and developers opinions should be collected in brainstorm sessions where many of those possibilities can be identified and eventually filtered out, so as to abandon some of them.
- For the potential clients it is fundamental to see early functional prototypes, as they help clients decide what they want.
- Information storage in NFC tags is very simple to realise.
- NFC communications modes are simple and easy to use.

Regarding the failures, the following ones stood out as deserving more careful consideration:

- The interoperability with existent systems must be carefully evaluated. In the present case study, restaurants were less interested than initially expected also because of the addition of another system.
- The overlap with other similar tools can empty the initial goals: regarding the sharing of data, the NFC technology must become widely available to justify its use as a vehicle for sharing.
- A technology can be in a too preliminary state to be useful to some application contexts: presently, smartphones/tablets with NFC technology are too few and expensive.

6 RELATED WORK

We could not find any article on the lessons learned along the development of a NFC application. Yet, it is easy to find this type of application. In fact, it was possible to identify several NFC-based applications for restaurants. Apparently, if NFC becomes widely available, this kind of application can become very common.

Next, we point to some references about the use of NFC in restaurants.

The application, named NFC Connected Restaurant (CoRest, 2013), was created by CostumerIn, a Canadian software development company. It also supports menus and the ordering of meals using the phone. Enable Table is another app which seems to have similar functionalities (EnabelTable, 2013).

Some other NFC-based applications for restaurants are available in the market, sometimes for a specific restaurant or a specific mobile device and with distinct functionalities. For example, some emphasize menus and orders (Bangkok, 2013), payment and coupon offers (Enable, 2013), or menu translation (EREst, 2013). Finally, a recent column (Sabella, 2013) emphasizes the use of NFC in restaurants and presents several examples.

7 CONCLUSIONS

This paper presents a mobile application prototype. As this is a first prototype, some important features have not yet been developed. Nevertheless, it has allowed us to identify several successes and failures. As

future work, we intend to extend our project through the following activities:

- Addition of a database system to manage restaurant information and associated back-end development (for restaurant data management).
- Design optimization for multiple screens.
- Application testing in a restaurant environment.
- Support for bill payment.
- Connection between SmartRestaurant mobile application and the restaurant website or other restaurant software.
- Outside the restaurant, support for reservations and orders using restaurant flyers or cards with an embedded NFC tag.
- Support for other mobile operating systems.
- Definition of a test case.

ACKNOWLEDGEMENTS

We would like to thank Pedro Martins, who have brought up to us NFC tags and restaurant software idea. His suggestions and knowledge about NFC technology have been very useful for this project. This work was supported by ”Sistema Regional de Transferência de Tecnologia SRTT Operação: ALENT-07-0262-FEDER-001870”.

REFERENCES

ADev (2012). Android developers: NFC basics. http://developer.android.com/guide/topics/connectivity/nfc/nfc.html.

ADT (2013). Android developers: ADT plugin. http://developer.android.com/tools/sdk/eclipse-adt.html.

Alencar, P. and Cowan, D. (2011). *Handbook of Research on Mobile Software Engineering: Design Implementation and Emergent Applications (2 Volumes)*. Engineering Science Reference, 1st edition.

Ambler, S. W. (2012). Uml 2 deployment diagrams. http://www.agilemodeling.com/artifacts/deploymentDiagram.htm.

Android (2012). Android developers: Near field communication. http://developer.android.com/guide/topics/connectivity/nfc/.

Bangkok (2013). Bangkok restaurant offers pizza ordering via NFC. http://www.nfcworld.com/2012/08/27/317380/bangkok-restaurant-offers-pizza-ordering-via-nfc/.

CoRest (2013). The NFC connected Restaurant App. http://www.customerin.com/default.ht.

DevTools (2012). Android developers: Developer tools. http://developer.android.com/tools/index.html.

EnabelTable (2013). Enable Table. http://www.enabletable.com/EnableTable.com/Welcome.html.

Enable (2013). Enable Table™ NFC. http://www.prweb.com/releases/2011/01/prweb4963794.htm.

EREst (2013). E-Restaurant NFC. http://www.e-restaurantnfc.com/restaurateurs.php.

Gargenta, M. (2011). *Learning Android*. O'Reilly Media, Inc., 1st edition.

Mednieks, Z., Dornin, L., Meike, G. B., and Nakamura, M. (2012). *Programming Android: Java Programming for the New Generation of Mobile Devices*. O'Reilly Media, Inc.

Morris, B., Bortenschlager, M., Luo, C., Sommerville, M., and Lansdell, J. (2010). *An Introduction to bada: A Developer's Guide*. Wiley Publishing.

NFC (2013). NFC forum: About NFC. http://www.nfc-forum.org/aboutnfc/.

NFC forum (2012). NFC forum: Technical specifications. http://www.nfc-forum.org/specs/spec_list/#tagtypes.

Nokia (2013). Nokia developer: Understanding NFC data exchange format (NDEF) messages. http://www.developer.nokia.com/Community/Wiki/Understanding_NFC_Data_Exchange_Format_(NDEF)_messages.

Sabella, R. P. (2013). NFC, Next on the Menu for Restaurants. http://www.nfcbootcamp.com/nfc-next-on-the-menu/.

Screens (2012). Android developers: Designing for multiple screens. http://developer.android.com/training/multiscreen/.

UML (2013). UML deployment diagrams examples. http://www.uml-diagrams.org/deployment-diagrams-examples.html#deployment-example-android.

Wasserman, A. I. (2010). Software engineering issues for mobile application development. In *Proceedings of the FSE/SDP workshop on Future of software engineering research*, FoSER '10, pages 397–400, New York, NY, USA. ACM.

Xcode (2013a). This is what developing for android looks like. http://techcrunch.com/2012/05/11/this-is-what-developing-for-android-looks-like/.

Xcode (2013b). Xcode 4. https://developer.apple.com/xcode/.