

# NFC on mobile phones: issues, lessons and future research

Vassilis Kostakos & Eamonn O'Neill  
Department of Computer Science  
University of Bath  
{vk, eamonn}@cs.bath.ac.uk

## Abstract

*In this paper we discuss some issues arising from our implementation and evaluation of two systems that enable mobile phone users to identify common address book contacts. These systems were implemented using Bluetooth and NFC. Here we discuss development and deployment issues, users' perceptions and preferences, and suggest promising future research directions.*

## 1. Introduction

In this paper we report lessons learned from our implementation of an application that uses either NFC or Bluetooth to allow users to identify the common contacts in their address books without revealing further information. First, we give a brief overview of our application. We then focus on the lessons learned from this project, including development and deployment, the difference in technological affordances between NFC and Bluetooth, and the user perceptions we recorded in a limited user evaluation.

## 2. Our address book application

Our system, described in [1], runs on mobile devices with J2ME, such as phones and PDAs, and allows users to encrypt and exchange address book information. This includes names, phone numbers and email addresses stored in users' mobile devices. For our prototype, these were stored directly in our application instead of tapping into the phone's native address book. Accessing the phone's native address book was not possible across all the devices for which we were developing, but is becoming increasingly achievable as J2ME is upgraded and phones evolve. Before a data exchange takes place, our system performs a one-way encryption (digest) of every entry in the address book. This ensures that the two-way exchange can reveal only information that is common to the users. Additionally, users have the option of

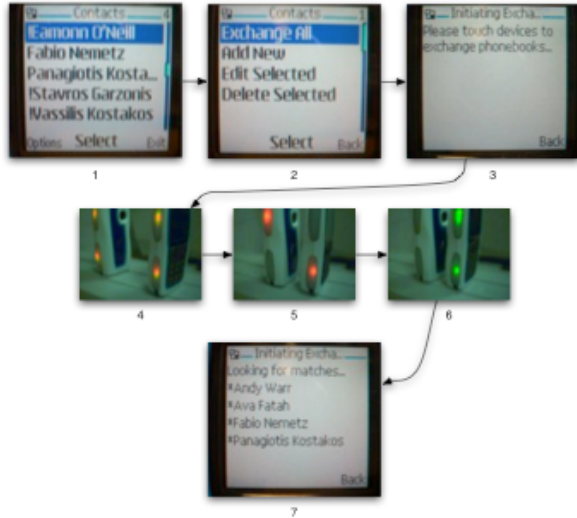
identifying entries as "Private", thereby withdrawing them from the exchange process.

The functionality of our system is shown in Figures 1 and 2. We developed two versions of our system – one using Bluetooth (on a Nokia 6680) and one using NFC (on a Nokia 3220 with NFC shell). Unfortunately a phone with both Bluetooth and NFC was not yet available, although the differences between the phones are negligible for our purposes here. The interface functionality was the same across both systems, but each device rendered the interface components differently. This is a feature of J2ME over which we had no control.

In addition to the development, we carried out a limited user study with 5 participants. In this study, participants were asked to use both systems and gave us their impressions of both systems. We present some of these findings here.



**Figure 1. Using Bluetooth, a user needs to activate the exchange mechanism (photos 1 to 4). The second user is alerted to the request for exchange (photo 5). If the second user agrees, the phones carry out a two-way exchange of digests. Upon successful completion of the exchange, the phones present the common entries (photo 6).**



**Figure 2.** Using NFC, both users need to activate the exchange mechanism (photos 1 to 3). The users place the phones next to each other, and wait for the exchange to take place while lights indicate the progress (photos 4 to 6). The phones then vibrate and display the common entries (photo 7).

### 3. NFC issues

In developing our applications we encountered a number of software and hardware issues with NFC. Here we discuss issues relating to the use of NFC and highlight implications that came up in our user study. Throughout our discussion we draw out pointers to future research directions.

#### NFC basics

The Nokia NFC shell operates at the radio frequency of 13.56 MHz. It supports both reading and writing tags. Although transparent to the programmer, the ability to both read and write tags is due to the fact that the hardware reader installed in the phone can change its behavior and emulate a tag. Thus, when switching from a read to a write, the hardware changes its configuration. By default, the read and write operations are one-way. In a sense, therefore, our address book application is novel in that it emulates a two-way communication between NFC devices.

The NFC SDK provides programmers with an extensive event-driven architecture which notifies applications when a tag has been touched, when data has been read, when data has been sent, and so on. It is interesting to note that because the whole system runs on a phone and is thus battery-driven, the system stops working when the battery level falls below

approximately 20%. Below this threshold, the system returns error messages to any hardware operation request, such as sending data.

### Interoperability

A certain degree of compatibility across NFC devices has been achieved by the introduction of standard data formats. The NFC Transfer Interface Packet (NTIP) is the first such standard. The NFC Transfer Interface Packet defines the NTIP data structure format as well as rules for constructing a valid NTIP Packet as a collection of NTIP records. Furthermore, it defines the unique identification of NTIP record types by different parties. An NTIP record can contain, for example, information about an SMS, such as the phone number and the text of the message.

### Physical hyperlinks and actions

This standard allows for tags to be assigned actions. A user can, for example, create a tag to call ‘Home’, or to SMS ‘Joe’ that ‘I’ll be late’. In this way, NFC tags can act as physical hyperlinks or physical shortcuts for specific actions. For instance, users can spread a number of tags on a desk or a wall and annotate them with various actions that will be executed when the phone touches the tag. Thus, physical address books become an interesting application to explore in further research.

This physical hyperlinking feature is greatly enhanced by the presence of another J2ME feature: the ability automatically to launch a midlet when the phone touches the tag. The Nokia NFC shell enables a midlet to register itself for automatic launching when a tag or another NFC device is touched. The midlet registers itself by calling an SDK method which allows it to specify the types of tags it can handle. The types of tags supported, and for which a midlet can self-register, are shown in Table 1.

Tag type	Max Read/Write per operation	Max storage on tag
Mifare 4k	1000	3360
Mifare 1k	720	720
Mifare ul	48	48
NTIP container	48-1000	48-3360
NFC device	1000	-
ISO14443-4	1000	-

**Table 1.** The types of tags supported by NFC.

The first column defines the tag name while the second column defines the maximum amount of data (bytes) that can be written to or read from a target instance in a single operation. The third column shows the maximum amount of user data (bytes) that can be stored on the tags.

The first three types of tags are the Mifare<sup>1</sup> 4 kilobyte, 1 kilobyte, and ultralight. Each of those tags can also be formatted as an NTIP container. Additionally, other NFC phones are compatible, as well as standard ISO14443-4 devices. In the case of two devices communicating with each other, as in our application, read and write operations are limited to 1000 bytes. Furthermore, when two devices exchange information, one of them needs to emulate a tag, and therefore a two-way communication channel is not possible to achieve.

### Malicious parties

Another interesting research issue to explore is that the combination of automatic launching of midlets and the assignment of actions to tags can lead to unwanted situations. For example, when standing on a bus or the underground, phones could accidentally come in contact and establish communication. This is something we considered while building our application, and it also came up in our discussions with participants in our user study. Our solution was to disable this automation, and require explicit user input. This stops a malicious party from exploiting the automation provided by NFC and J2ME by causing people's phones to dial or SMS any number, or by injecting other people's tags with advertisements, spam or malicious hyperlinks.

Thus, the same mechanisms that can allow physical address books and other everyday objects to act as containers of actions can also leave their users vulnerable to malicious parties. Further research should explore the appropriate balance between automation that allows us to experiment with physical objects as shortcuts to actions and security mechanisms that do not expose their users.

One security mechanism that we can put to use is tag authentication. NFC tags do not have any lock bytes to make them write-protected, but instead we can use authentication keys to make an NFC tag write-protected or even read-protected. This can be achieved by writing the authentication keys to the tag.

Currently, the NFC SDK provides only the means to set the two keys (keys A and B) when formatting a tag as an NTIP container. The format command will

write access bits to all user data sectors as follows: read access for key A, read and write access for key B.

Users can specify their own authentication keys to read or write to tags when they format them. If no user keys are specified the set of default authentication keys is used, shown in Table 2. The user-defined key, if provided, is always used first when authenticating a tag. If it does not match, the default keys are subsequently tried.

Key	Description
0xD3F7D3F7D3F7	The default public NTIP key. Used as the A key for reads and the B key for writes.
0xFFFFFFFFFFFF	Manufacturer default key 1. Always used as the A key.
0xA0A1A2A3A4A5	Manufacturer default key 2. Always used as the A key.

**Table 2. The default keys used for tag authentication.**

## 4. Using NFC

One of the first obstacles we had to overcome with NFC was that users could not easily use their phone's keypad while touching another phone. This is also true when tags are used. This meant that our system could not ask for user input while an NFC communication was taking place. Additionally, we found that users could not easily read the phone's screen while touching another phone. This meant that we had to use the phone's lights and vibrator to notify users of the progress and status of the exchange. The notification capabilities of this specific phone include vibration, a set of four lights embedded in the sides of the phone, and sound. We believe that a combination of all three of these can provide useful feedback and an enhanced experience when using NFC. This is a direction that our ongoing research has taken.

Another issue we identified was that the NFC exchange, although taking much less time than a Bluetooth exchange (about 5 seconds as opposed to about 30 seconds), was still slightly uncomfortable. This was because users had to hold their phones in an awkward position. The delay in the process was related to the way the communication took place.

Our application had only one interface option — "Exchange". Both users had to select "Exchange", and then touch the devices. At this stage, both devices

<sup>1</sup> See <http://www.mifare.net>

would be attempting to send as well as receive information. Apparently randomly, one of the devices succeeds in transmitting the data first. At this stage, the devices have to rediscover each other (whilst still touching) and carry out the second part of the exchange. The bottleneck in this process was the second part of the exchange, where both devices had to alter their hardware configuration and rediscover each other.

Another way of carrying out the two-way digest exchange would be to have two interface options: "Send information" and "Receive information". This would make the NFC exchanges themselves very fast, but would require extra user input as well as physically separating and retouching the phones.

The underlying communication technology had an effect on the user experience. NFC provides a much more engaging physical experience than Bluetooth, which is reinforced by the symmetry of users' physical actions. With NFC, the physical act of touching the phones is symmetric between users, and they receive the same feedback, at the same time, from their respective phones. With Bluetooth, the joint experience is not so strong since Bluetooth technology imposes a request-reply model that makes the experience asymmetrical. Furthermore, there is little or no physical interaction using Bluetooth.

Participants seemed to prefer the NFC application for face-to-face interactions. When asked to elaborate on this, they claimed that it was easier to carry out the exchange using the NFC system because it involved fewer steps.

Verifiability was clearly identified as an issue of concern to the participants, in particular the problem of verifiability of Bluetooth contacts. While one participant preferred the Bluetooth system, claiming that it would be useful in getting to know new people, another participant claimed that she would be very reluctant to respond to Bluetooth exchange requests from someone unknown. She claimed that with Bluetooth everyone could "see" her, while with NFC only friends could "see" her.

From our discussions with participants, we conclude that the responsiveness of NFC is an advantage, provided that no user interaction with the phone is required while NFC is being used. Additionally, if the users are having a face to face conversation and are close enough to choose between touching their phones together or using the Bluetooth system they will prefer the former.

We had to explain to the participants that the purpose of our system was *not* to exchange phone numbers. All our participants commented that the NFC technology would be very useful for simply exchanging phone numbers. This appears to be a user requirement that currently is not effectively addressed by mobile phones.

## Summary

In this paper we briefly described an application we developed using both NFC and Bluetooth. This application allows users to identify the contacts that they share. We discussed a number of technical and research issues that we identified while building and evaluating our systems, as well as in a subsequent evaluation study.

We believe that NFC is a technology that is perceived as "simple" by end users. Full two-way communication cannot be achieved using NFC, yet the speed and ease of its setup is a positive feature of this technology. Although some security issues still need to be addressed, NFC can make use of everyday objects as containers of hyperlinks and actions, thus opening up a whole new array of potential applications. The limited usability of the phone is a key limitation at the moment, but the augmentation of NFC use with feedback using lights, sound and vibration can offer a rich user experience.

## Acknowledgements

The authors' research is funded by the UK Engineering and Physical Sciences Research Council grant EP/C547683/1 (Cityware: urban design and pervasive systems). We thank Vodafone Group R&D and Nokia Insight & Foresight for their support of this work. We also thank Stavros Garzonis, Andy Warr, Kharsim Al Mosawi, and Tim Kindberg.

## References

- [1] Kostakos, V., O'Neill, E., Shahi, A. (2006). Building common ground for face to face interactions by sharing mobile device context. Workshop on Location and Context Awareness (LOCA 2006), Dublin, Ireland. Lecture Notes in Computer Science 3987, Springer, pp. 222-238.