Beyond Device Pairing: New Interactions on NFC Enabled Mobile Phones

Yaw Y. Anokwa
Department of Computer Science and Engineering
University of Washington
Seattle, WA 98105
yanokwa@cs.washington.edu

May 18, 2007

Abstract

Near field communication (NFC) is a short-range wireless protocol that allows users to connect devices and access content and services by simply holding enabled devices near each other. Many of the existing applications (ticketing, purchasing, device configuration, etc.) use NFC as a method to transfer unique identifiers which then inform a larger system. While useful, these applications do not focus on the strengths of NFC technology or go beyond single function scenarios. This problem is addressed with a user interaction model for NFC enabled applications. The model draws parallels with the GUI model, leverages the existing knowledge users have about physical items in their environment, and can support a variety of applications tied together in a simple and intuitive way. To realize the model, custom hardware and software tools which allow researchers to build a variety of applications is provided. The tools exploit the unique capabilities of NFC and take the technology beyond pairing.

1 Introduction

Mobile devices have become the primary platform of ubiquitous computing. Recent research has shown that over two billion users worldwide [21] carry mobile phones with or near them [11]. As these devices have grown in popularity, a rich and vibrant research community has been spurred by the promise of more natural interactions between users, devices, and their environments.

One method of enabling this interaction between users and their environments was brought forth by Want et. al [20]. That work presented the idea of augmenting everyday objects with radio frequency identification (RFID) tags. Actions were triggered by scanning the augmented objects. For example, scanning a book could automatically purchase it from an online book seller or scanning a business card could add its data to a computer.

One could also imagine a scenario where users may, instead of carrying around various tagged objects, carry a single device that has the same functionality as those objects. For example, instead of a tagged car key and a tagged house key, a tagged mobile phone could be authorized to open your car and your house.

1.1 Background

In RFID technology, *tags* respond with a unique identifier when interrogated by *readers*. While some tags are active (battery powered), most tags are passive and thus require the reader's interrogation signal for the

power required to generate a response.

Near field communication (NFC) is a technology evolved from RFID that enables most tagged objects to communicate with each other. NFC's notions of tags/readers and active/passive are similar to those of RFID. Given an active NFC reader and a passive NFC tag, NFC behaves like common RFID systems.

Similarities fall away when readers communicate with other readers. When two NFC readers are within range, they can communicate like other radio technologies (WiFi, Bluetooth, ZigBee, etc.). If both devices are active, each generates its own radio field for communication. If one of the devices is passive, the active device generates a radio field to power both. In either case of reader to reader communication, bi-directional data transfer can occur at 106 kbits/s, 212 kbits/s or 424 kbits/s. RFID readers have no such functionality.

NFC also distinguishes itself from RFID by its extremely short range. While the NFC standards limit range to 8 inches, the effective range of most deployed systems is around 2 inches. Essentially, NFC provides a bi-directional channel that is open when devices practically touch and a channel that is closed when those devices move apart. Although limited range could be a hinderance, it also affords enabled devices inherent security – a user knows which device he/she is exchanging data with because physical proximity is required.

The most widely deployed NFC system is Sony's FeliCa card – the money/transit card in Japan's public transportation system. When using the card at a transportation terminal, it responds to readers with an identifier which is tied to an account. If that account has a positive balance, a user is authorized to travel, and the travel fees are deducted appropriately. If the balance is not sufficient, the user is not authorized to travel.

Future applications of NFC technology have been addressed in a number of places. A study by ABI Research [15] predicts that 50% of new mobile phones will support NFC by 2009. The study suggests that point of sale applications and facilitating Bluetooth discovery will be key applications in the mobile deployments. Additionally, the NFC Forum [8], the organization responsible for the NFC specifications, has laid out a framework for future application functionality.

One of the applications the NFC Forum envisions is the "smart poster". An enabled movie poster would respond to NFC scans with a URL to a ticketing site. The user could use the mobile phone's 3G or WiFi connection to purchase and download a movie ticket to the phone. That ticket could then be checked when the user arrived at the movie theater.

While useful, the smart poster ignores many of the actions a user might want to take at the poster. Perhaps the user would like to read reviews about film, download a copy of the poster, or get directions to closest theater. As long as the data was under 1MB, much of this information could be embedded in the tag and immediately transfered to the mobile device.

Instead of powering up a 3G or WiFi radio and launching a browser, users could even download tickets embedded in the poster over NFC and transfer those tickets to other mobile devices by physically bringing the two devices together. After attending the movie, the user could keep remnants of the ticket as a virtual stub in their device. The NFC chip's ability to act as both a tag and a reader makes these scenarios possible.

1.2 Problem

Despite the array of possibilities, many of the proposed NFC enabled applications are simply RFID (active reader, passive tag) style single function scenarios. This work's goal is to enable more exploration and more peer to peer scenarios as well as ensure usability and adoptability.

A parallel can be drawn with the personal computer. In the same way that the "window, icon, menu, pointer" model made the GUI successful, similar models can enable a positive user experience across a number of applications in this space. Can these scenarios be enabled without breaking usability and flexibility? Can NFC hardware and software go beyond the design goals of the technology?

These questions are answered with a user interaction model for NFC enabled applications. The model draws parallels with the GUI model and leverages the existing knowledge users have about physical items in their environment. This allows for supporting a number of different applications tied together with simple and intuitive interactions. To realize the model, custom hardware and software which allow researchers to exploit the capabilities of NFC are also provided.

The remainder of this paper surveys related work and describes what the contributions are. The interaction model is then discussed in more detail along with a description and evaluation of the hardware and software provided. Lastly, the contributions of this work and future research challenges are presented.

2 Related Work

Want et. al's [20] work presented the idea of augmenting everyday objects with RFID. A similar project, Cooltown [5], linked real world objects with corresponding services in a URL-centric model of interaction. Devices in Cooltown required a web server making it difficult to implement on small mobile and powerless devices. There has also been work on RFID middleware [4, 16] that could enable scenarios mentioned above, but nothing that explicitly mediates NFC interactions across a variety of applications.

Valkkyen and Tuomisto [19] define physical browsing as object selection when performed by either pointing, scanning or touching. Pohjanheimo et. al [14] demonstrate the feasibility of such a system on a mobile phone. Rukzio and Leichtenstern [17] conduct a user study which demonstrates that for a variety of scenarios, touch is the preferred interaction method when intuitive and error resistant interactions are required. Much of the above work favorably compares touch with other methods of selection, but does not consider what happens after the touch.

Broll et. al [3] introduce a mobile interaction paradigm where devices use tags embedded in physical object as parameters and controls for the interface. While innovative, the evaluation shows subjects did not find the interface intuitive or natural.

Kostakos and O'Neill [6] argue that users find NFC more engaging than Bluetooth and more efficient than 2D barcodes. They identify full two-way communication in NFC as a requirement for superior usability and state that by tagging everyday objects with NFC, researchers could enable a full range of applications like Tag-o-scope (probe everyday things for hidden information) and PhotoSending (link personal objects to trigger photo sending to specific people). Makela et al. [7] demonstrate that most users "make sense of the world by developing a mental model based on prior relevant experience." In general, they have vague notions of how to interact with mobile RFID technology.

This work addresses the challenges posed by previous work. Specifically, the interaction model proposed is built on the previous experience that users have had with the tagged items. Additionally, the work enables full two-way communication and making the applications proposed by Kostakos and O'Neill [6] possible.

3 User Model

The proposed model addresses the problems posed above by specifying a series of constructs that leverage the existing knowledge users have about certain objects. By relying on the physicality of NFC as a technology

paired with the intuition that users already understand everyday items, additional functionality can be mapped to those items when represented digitally.

3.1 Model Definitions

An *item* is defined as any physical NFC enabled device (both passive and active). *Objects* are all the virtual things the physical item represents. For example, an item may be a movie poster, but that poster may represent objects ranging from a copy of the poster, to tickets you can buy or a trailer you can view. Objects can also be the context or properties of the item.

Actions are things that an object can do or can be done to the object. For example, an action for a ticket might be 'give as gift' but for a trailer, an action might be 'watch now'. *Interactions* are the process of using an object or doing an action and *events* are the results of an interaction.

Objects and actions can be placed in *bags*, virtual entities that define a set of objects and actions. Bags can also be used as access control. For example, reading a private bag requires authorization through sharing of a key (perhaps another object), but reading a public bag requires no authorization.

NFC Model	**Explanation**	**Example**	**GUI Equivalent**
Items	Actual NFC enabled item	A movie poster with a tag	Screen
Objects	Virtual things an item represents	Tickets, trailers, directions	Icons
Actions	Things that can be done to objects	A trailer can be played	Operations
Bags	Virtual groups of objects and actions	Tickets on a phone	Window
Interactions	Using objects or doing actions	Playing a trailer	Interactions
Events	Results of an interaction	A trailer is played	Events

Table 1: NFC model definitions with examples and GUI equivalents

Many of these terms are similar to concepts in the GUI model. For example, a menu could be seen as a list of actions and objects. Double-clicking an icon to open an application is an action. The objects users interact with are icons. Windows or folders could be seen as bags. The physical interaction with an item is similar to moving a cursor around on a screen. An overview of these definitions and the GUI equivalents are shown in Table 1.

3.2 Model Detail

Model usage is separated into two categories, scanning an item and using an object/action. Scanning explains how two mobile devices gather objects and actions and using describes how those objects and actions are used.

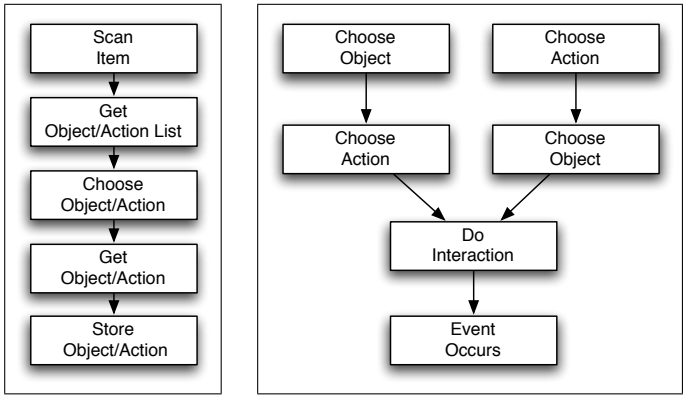
3.2.1 Scanning an Item

When a mobile device scans a physical item, the item returns the relevant information needed to describe itself. This can be a list of objects and their associated actions. The user specifies which actions to take and/or objects to retrieve. The mobile device then executes the choice, possibly including the transfer of object data into the mobile device. For example, scanning the smart movie poster gets a list of actions/objects such as: buy a couple of movie tickets, download a trailer, read reviews, or download directions to the theater.

Each of these actions may cause an object (the trailer, reviews, etc.) to be downloaded into the device. The interface on the device can reflect this interaction by showing an animation of the action or objects involved. Alternatively, the item could only contain objects which could be transfered to the device. For example,

moving the ticket object to your phone could also automatically trigger a financial transaction between the phone and the movie theater. The process of scanning an item is shown in Figure 1a.

Once the objects are on the device, another series of actions can be associated with each object. These actions either come from the information read from the item or can be added by the mobile device. The actions should reflect what could be done with the virtual object, but more complex actions should also be available for advanced users.



(a) Scanning an item gets and stores an object or action on a device.

(b) Once on a device, users select an object and/or an action. Interactions can be as simple as another scan or as complex as using an application.

Figure 1: Retrieving and using objects and actions

3.2.2 Using an Object/Action

To use an object or an action on a device, users can either choose an object and pair it with an action (e.g., 'select a ticket' and do the 'send to friend' action) or choose an action and pair it with an object (e.g., 'play media' then select the trailer). Users can also select just an object or action and rely on defaults. Whichever choice is made, a user can then do an interaction. This could be a gesture, some function on the device or even another scan. The desired event then occurs. This process of using an object/action is shown in Figure 1b.

3.3 Model Scenario

The following scenario describes the discovery, purchase and transfer of movie tickets and is an illustration of how the concepts presented so far could be combined.

John is walking through the mall when he sees a movie poster for the new Drake Stone movie has been placed near the food court. The promotional poster boasts a new NFC based technology called MovieTouch. John knows his Motorola ROKR E2 phone is NFC enabled so he touches the poster with his phone. Immediately, the phone beeps and displays a list of objects and actions that the poster has sent to the phone. Unknown to John, the poster also sends reviews, directions and other static content to his phone. The content will timeout if John does not use it soon.

John doesn't know much about this new movie, so he selects the 'View Reviews' action on his phone. While browsing the reviews, he learns that the movie is highly rated and so John's girlfriend Alice (a huge Drake Stone fan) and would probably want to see the movie tonight. John returns to his 'Objects and Actions'

menu and this time selects 'Buy Tickets'. He enters the quantity on the phone and touches the poster again.

The phone's location service selects the nearest theater and sends this information to the poster during the touch. The poster is connected to the theater's ticketing system and so downloads the tickets and charges the cost of the tickets to John's phone bill. If the poster had no Internet connectivity, John could have still purchased tickets embedded in the poster's tag.

Looking through the 'New Objects' list on his phone, John confirms that he has both tickets and moves one to his public bag and the other to his 'Alice Bag'. John texts Alice and asks her to meet at the theater and promises a surprise.

When they meet, John allows Alice to touch her phone to his. The movie ticket in Johns Alice Bag is transferred to her phone over NFC. She places her ticket in her public bag. When Alice and John enter the theater, they touch their phones to the ticket booth. The ticket booth authorizes their access to the Drake Stone movie and leaves a virtual movie stub in their private bags that Alice and John can keep as a souvenir.

4 System Components

Researchers who wish to use mobile NFC systems must consider the tradeoffs between custom and off-the-shelf systems. Custom solutions require significant engineering effort and may not always produce realistic hardware for users. Additionally, if researchers intend on adding the custom solution to a commercial device, they must often modify the software and hardware of that device as well, potentially changing the user experience negatively.

Off-the-shelf system such as the Nokia 3220 [9] or Nokia 6131 NFC [10] are not necessarily better options. These systems do not allow for exploration of advanced use cases like file transfer over NFC, using NFC to start Bluetooth and WiFi connections or sensors (accelerometers, ZigBee sensors, etc.) paired with NFC. Additionally, the provided SDKs often do not expose of much of the low level functionality needed to exploit all parts of the NFC standard.

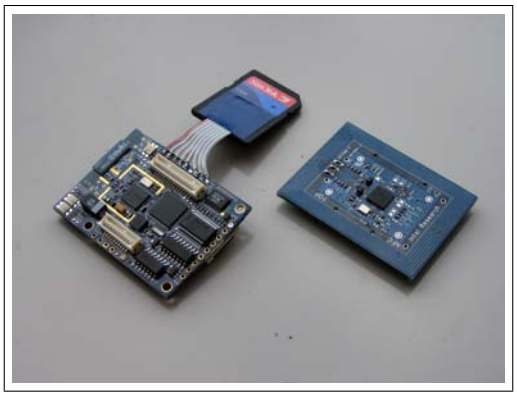
Rather than be constrained by a commercial solution, a custom solution which could be made available to the research community was built. This system enables a variety of applications and through experimentation can answer basic questions about what a custom system can offer. For example, how fast is the NFC transfer protocol in exchanging information? Is it more effective (both power and bandwidth) to use NFC to bootstrap another communication technology (Bluetooth, WiFi) for data transfer? How fast is the interaction and what are the tradeoffs?

4.1 Hardware

The system hardware is built on the Motorola Linux phone platform. The devices, the E680i and ROKR E2 (shown in Figure 2a), are based on an Intel PXA-27x processor (ARM architecture) and run up to 400Mhz. Internally, they have up to 48MB of DRAM and up to 80MB Flash memory. They are also equipped with Bluetooth 2.0, USB 2.0, and an MMC/SD expansion slot that supports 2GB SD cards. The phones boot the Linux 2.4 kernel and run binaries built with a C cross-complier as well as Java ME midlets.

In addition to the host phone, a Phone System Interface (PSI) board is used. The PSI board is a module designed to add expansion capabilities to phones. The PSI board fits into the MMC/SD slot of the host phone and allows for transparent access to a MMC/SD card, a MSP430 microcontroller, a 3-axis accelerometer, an expansion connector and a 802.15.4 (ZigBee) radio. The MSP processor runs a custom program that manages the communication between the phone and the PSI's sensors. The PSI can multiplex the phone's

(a) Motorola ROKR E2 Linux phone



(b) PSI board (left) and NFC board (right)

Figure 2: NFC system (components)

SD slot with all sensing to allow SD card support even when the PSI board is inserted. On top of the PSI board sits a Philips PN531 based NFC board that implements ISO 18092, ISO 14443A, MIFARE and FeliCa modes. The hardware components are shown in Figure 2b and the technical details of the implementation is in earlier work [13].

The two board sandwich is plugged into the SD slot of the phone as shown in Figure 3a and wrapped around the back as shown in Figure 3b. The entire package is designed to sit on top of the battery compartment of the phone and can be covered with a modified battery cover. Despite all the modifications, the phone functions as originally designed.

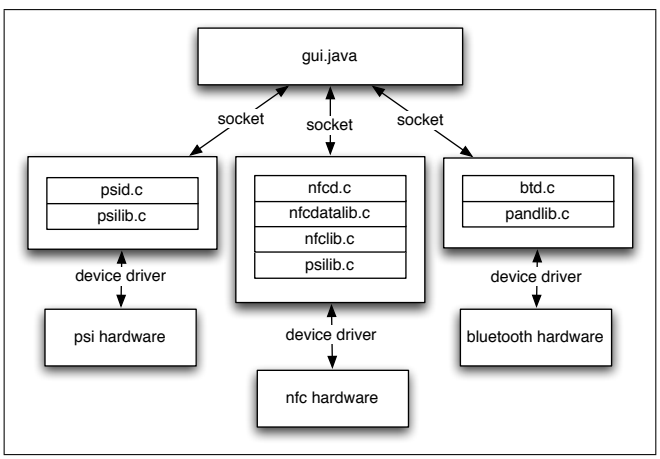
(a) The NFC board sits on the PSI board. The PSI board connects to the phone via the SD attachment.

(b) Both boards are then wrapped around the phone, and sit on the battery compartment.

Figure 3: NFC system (package)

4.2 Software

The system software stack is shown in Figure 4a. The PSI board's drivers are implemented as dynamically loadable kernel modules and software modifications were made to the phone to prevent the stock MMC/SD kernel drivers from loading on startup. Once the PSI kernel module loads, the PSI board can be controlled directly using standard POSIX read/write/select calls implemented in psilib.c.



(a) Commands initiated at the GUI are sent through sockets to daemons and then to the hardware. Data received is sent back through the same pathway.

(b) Screenshot of GUI application for sharing files between two phones.

Figure 4: NFC System (software)

NFC hardware is controlled through the PSI device drivers and basic NFC functionality is provided with POSIX read/write/select calls implemented in nfclib.c. Peer to peer data transfer is implemented in nfcdatalib.c. The ROKR E2's Bluetooth stack is replaced with BlueZ [2] and pandlib.c provides programmatic access to the hardware. A similar abstraction could be used to enable WiFi connectivity through the SD slot.

A set of daemons (psid.c, nfcd.c and btd.c) run at startup and expose localhost sockets. Developers who write Java ME applications can send commands and receive data from the hardware. Currently, nfcd.c only controls NFC hardware, but because psid.c controls the PSI it can also provide some NFC functionality. btd.c controls the Bluetooth hardware and allows for creating, destroying and verifying connections with Bluetooth devices.

To demonstrate the software stack's ease of use, accelerometer data sent from the PSI board is used in a basic gesture recognition system [12] built with Java ME. Significant change in device orientation are recognized and used as a selection mechanism for actions after an NFC scan. Another implemented application [1], a tool for file sharing using NFC and Bluetooth, is shown in Figure 4b.

The NFC initialization commands have been augmented with device configuration and connection information. This enables devices to instantaneously create a Bluetooth connection while the user is performing the selection of the tag. Bypassing the Bluetooth scanning process has been shown [18] to enable significantly easier use of the technology. By overlapping Bluetooth connectivity, the overall interaction time is reduced for large data transfers.

Without this optimization, users would either have to keep the phone near the tag for all communication or wait for the Bluetooth connection to be established to do any significant data transfer. Although NFC can do some data transfer, it requires the device to stay at the point of interaction during the entire transfer. Because users are likely to move away either as a result of a gesture or moving the screen closer for viewing, secondary communication channels are useful.

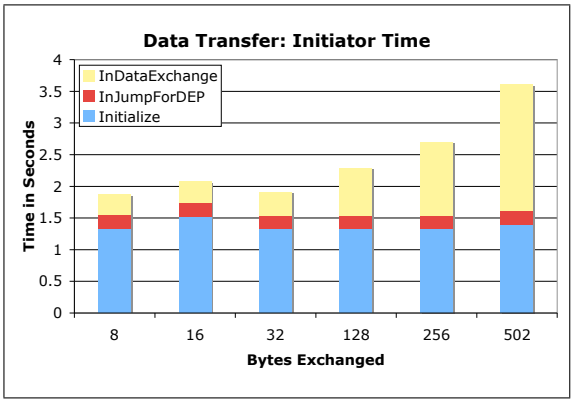
4.3 Evaluation

A number of tests were performed to characterize system behavior and properties when transferring data and scanning tags.

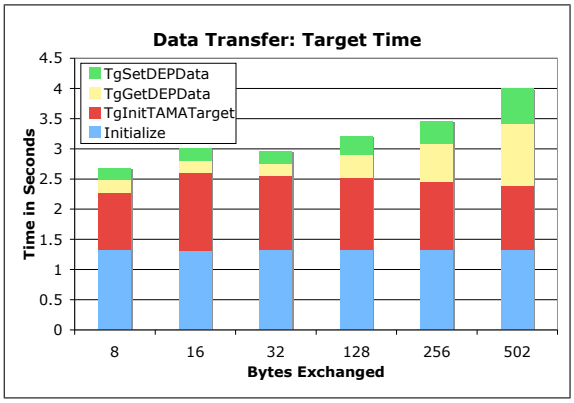
4.3.1 Data Transfer

For the data transfer trials, two ROKR E2 phones were placed so their NFC circuitry touched. One phone was designated as a *initiator* (device requesting the transaction), and the other a *target* (device polling for transactions). The devices implement the NFC Interface and Protocol (NFCIP-1) in active mode at 424kbits/s with a 251 byte maximum packet size. Per the specifications, the initiator executes InJumpForDEP continuously to activate the target. Once the target responds, the initiator executes InDataExchange to read and write data from the target. The target executes TgInitTAMATarget continuously to configure the chip as a target, and once the initiator responds, executes TgGetDEPData to read data and TgSetDEPData to write data. Both initiator and target require an additional Initialize on startup to prepare the PSI board.

Timing for each function starts before the function is called and stops when the function returns. Because the target may issue multiple TgInitTAMATarget commands as it polls, time starts at the TgInitTAMATarget before the first TgGetDEPData. For the initiator, time starts at the InJumpForDEP before the first InDataExchange. Total bytes exchanged are measured as well. For example, in a trial with 8 bytes, 4 bytes are sent by the initiator and read by the target and another 4 bytes are sent by the initiator and read by the target.



(a) Time breakdown of initiator

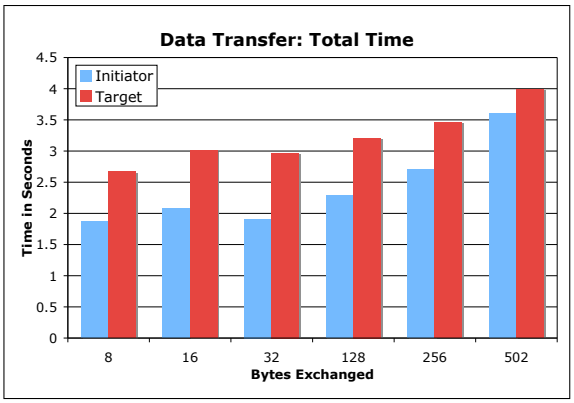


(b) Time breakdown of target

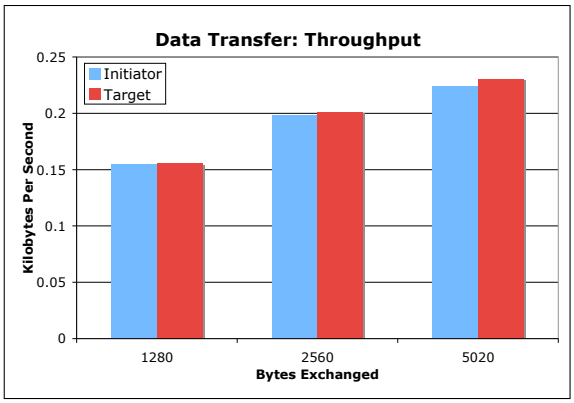
Figure 5: Graphs demonstrating time to completion for initiator and target. Devices were in active mode and set to communicate at 424 kbits/s.

Figure 5a shows time to completion for the initiator as a function of the number of bytes exchanged. The Initialize state as well as the InJumpForDEP are constant regardless of data sent. As expected, InDataExchange increases proportionally to the amount of data sent. The target shows a similar result in Figure 5b where TgGetDEPData and TgSetDEPData are proportional to bytes exchanged. Interestingly, the target's writing of data takes slightly longer than the reading of data, which is further illustrated by the total time to completion graph in Figure 6a. The target takes longer because unlike other commands which write a few bytes, the TgInitTAMATarget command writes over 30 bytes and like TgGetDEPData must wait for a response.

Figure 6b shows the throughput of the system. As the amount of data exchanged is increased, there is a slight increase in throughput. Instead of the 53 kb/s which NFC promises, the system peaks around .225 kb/s when 5020 bytes are exchanged.



(a) Total time of initiator and target



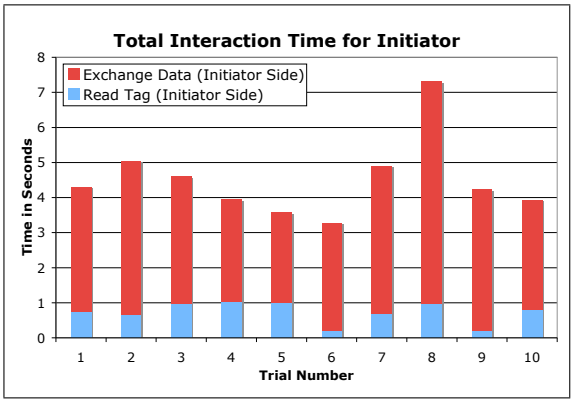
(b) Throughput of data transfer

Figure 6: Graphs demonstrating total time to completion and throughput of bidirectional data transfer.

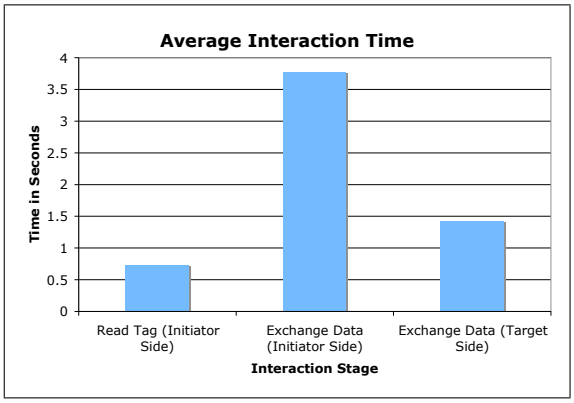
The reasons for the slowdown are independent but compound. First, the serial stream from the NFC chip is buffered in the PSI board before being sent to the phone. This buffering is required because multiple serial streams (e.g., accelerometer data) must also have access to the phone. Secondly, NFC is a 'round trip protocol', so to handle multiple packets of data, the system must receive data, make another call to send data back, and then make the call again to receive data. Finally, due to how the PSI firmware and device driver were implemented, the transfer rates between the PSI and host phone itself are slow. These slowdowns can be fixed, but is outside the scope of this work.

4.3.2 Scan and Transfer

For scan and transfer trails, the initiator first reads a passive MIFARE 4K card. Upon a successful read, the initiator begins the InJumpForDEP and InDataExchange cycle and is moved to the target and held there until a total of 108 bytes are exchanged. This trial is performed 10 times and both the scan and data transfer times are recorded. In these trials, target side time is started from the first valid TgInitTAMATarget and ended at the completion of TgSetDEPData. Initiator side time starts from the first InJumpForDEP and ends at InDataExchange.



(a) Total interaction time



(b) Average interaction time

Figure 7: Graphs demonstrating interaction time for 10 trials.

In Figure 7a, results of tag reads and exchanges of data from the initiator side are shown. Figure 7b shows

that reading the tag on the initiator side takes .75 seconds on average while exchanging 108 bytes takes an average of 3.75 seconds. The target's average time in exchanging this data is only 1.5 seconds. The increase in time in initiator values from the previous trials are due to the physicality of these trials – the phone must read the tag before being moved to the target. This process was done quickly, but has some inherent delay.

5 Conclusion

The contributions offered in this work center on an interaction model that relies on a user's pre-existing knowledge about the items with which they are interacting. While the model provided is natural, there are no claims made about its absolute correctness. Rather, it is a model which keeps the human-computer interaction issues in mind when building NFC systems. The model states that devices and tags should encapsulate the properties of the items they represent. Readers, in turn, should enable the concept of objects/actions so that users are aware of the possibilities of the application. Another contribution is the custom hardware and software which enable NFC researchers to work from protocol layer up to the application layer. The hardware is demonstrated on a commercially available platform and provides a software framework within which developers can build their applications.

Future extensions on this work will evaluate the effectiveness of the model with user experiments as well as explore other interaction issues that remain open. What if a user has no previous knowledge about the item with which they are interacting? How does the problem space change once multiple devices are introduced? Do users want to share their interests to all potential observers through the act of scanning an item? Research into how to build technologies and interfaces that consider these issues could be an interesting future direction.

NFC enabled applications and services offer the chance for improved usability for mobile devices and natural interactions with the world around us. Similar to the way the GUI mediates interactions across desktop applications, this work presents a model for mediating interactions across NFC applications and provides the tools to take technology beyond pairing.

6 Acknowledgments

I would like to thank Gaetano Borriello, Rohit Chaudri, James Fogarty, Hélène Martin, Trevor Pering, Roy Want, and Pei Zhang for their invaluable contributions to this work.

References

[1] Y. Anokwa, G. Borriello, T. Pering, and R. Want. A user interaction model for NFC enabled applications. In *Fifth Annual IEEE International Conference on Pervasive Computing and Communications Workshops*, pages 357–361, 2007.

[2] BlueZ. BlueZ - Official Linux Bluetooth protocol stack. *http://www.bluez.org/*, 2007.

[3] G. Broll, S. Siorpaes, E. Rukzio, M. Paolucci, J. Hamard, M. Wagner, and A. Schmidt. Supporting mobile service usage through physical mobile interaction. In *Fifth Annual IEEE International Conference on Pervasive Computing and Communications Workshops*, pages 262–271, 2007.

[4] H. Keranen, L. Pohjanheimo, and H. Ailisto. Tag Manager: a mobile phone platform for physical selection services. *International Conference of Pervasive Services (ICPS)*, pages 405–412, 2005.

[5] T. Kindberg, J. Barton, J. Morgan, G. Becker, D. Caswell, P. Debaty, G. Gopal, M. Frid, V. Krishnan, H. Morris, J. Schettino, B. Serra, and M. Spasojevic. People, places, things: web presence for the real world. *Mobile Networks and Applications*, 7(5):365–376, 2002.

[6] V. Kostakos and E. O'Neill. NFC on mobile phones: Issues, lessons and future research. *PerCom*, 0:367–370, 2007.

[7] K. Makela, S. Belt, D. Greenblatt, and J. Hakkila. Mobile interaction with visual and RFID tags: a field study on user perceptions. In *CHI '07: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 991–994, New York, NY, USA, 2007. ACM Press.

[8] NFC-Forum. NFC-Forum homepage. *http://nfc-forum.org*, 2006.

[9] Nokia. NFC shell for Nokia 3220. *http://www.nokia.com/link?cid=EDITORIAL_4795*, 2006.

[10] Nokia. Nokia 6131 NFC. *http://europe.nokia.com/A4307094*, 2006.

[11] S. N. Patel, J. A. Kientz, G. R. Hayes, S. Bhat, and G. D. Abowd. Farther than you may think: An empirical investigation of the proximity of users to their mobile phones. In *Proceedings of the Eighth International Conference on Ubiquitous Computing (Ubicomp 2006)*, pages 123–140, 2006.

[12] T. Pering, Y. Anokwa, and R. Want. Gesture Connect: Facilitating tangible interaction with a flick of the wrist. In *TEI '07: Proceedings of the 1st international conference on Tangible and Embedded Interaction*, pages 259–262, New York, NY, USA, 2007. ACM Press.

[13] T. Pering, P. Zhang, R. Chaudhri, Y. Anokwa, and R. Want. The PSI board: Realizing a phone-centric body sensor network. In *4th International Workshop on Wearable and Implantable Body Sensor Networks (BSN 2007), Aachen, Germany, 26-28 March 2007. IFMBE Proceedings Vol 13*, March 2007.

[14] L. Pohjanheimo, H. Keranen, and H. Ailisto. Implementing touchme paradigm with a mobile phone. In *sOc-EUSAI '05: Proceedings of the 2005 joint conference on Smart objects and ambient intelligence*, pages 87–92, New York, NY, USA, 2005. ACM Press.

[15] A. Research. Near field communications (NFC): simplifying and expanding contactless commerce, connectivity and content. *http://www.abiresearch.com/products/market_research/Near-Field_Communications_(NFC)*, 2007.

[16] J. Riekki, T. Salminen, and I. Alakarppa. Requesting pervasive services by touching RFID tags. *IEEE Pervasive Computing*, 05(1):40–46, 2006.

[17] E. Rukzio, K. Leichtenstern, V. Callaghan, P. Holleis, A. Schmidt, and J. S.-Y. Chin. An experimental comparison of physical mobile interaction techniques: Touching, pointing and scanning. In *Proceedings of the Eighth International Conference on Ubiquitous Computing (Ubicomp 2006)*, pages 87–104, 2006.

[18] D. Scott, R. Sharp, A. Madhavapeddy, and E. Upton. Using visual tags to bypass bluetooth device discovery. *SIGMOBILE Mob. Comput. Commun. Rev.*, 9(1):41–53, 2005.

[19] P. Valkkynen, I. Korhonen, J. Plomp, T. Tuomisto, L. Cluitmans, H. Ailisto, and H. Seppa. A user interaction paradigm for physical browsing and near-object control based on tags. In *Physical Interaction Workshop on Real-world User Interfaces*, 2003.

[20] R. Want, K. P. Fishkin, A. Gujar, and B. L. Harrison. Bridging physical and virtual worlds with electronic tags. In *CHI '99: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 370–377, New York, NY, USA, 1999. ACM Press.

[21] G. Wearden. Nokia: 2 billion cell phone users by 2006. *http://news.com.com/2102-1039_3-5485543.html*, 2006.