NFC mit Android

Seit der Android Version 2.3 (Gingerbread) unterstützt Google neu auch NFC (Near Field Communication) und verleiht damit dem Mobile Payment und Mobile Ticketing neuen Schub. NFC ist eine kontaktlose Schnittstellentechnologie, welche die einfache und schnelle Kommunikation über ca. 2 cm zwischen RFID-Tags und einem speziell ausgerüsteten Mobiltelefon ermöglicht. Mit Release 2.3.3 wird auch das Schreiben von Tags und die Kommunikation zwischen zwei Mobiltelefonen über NFC unterstützt. In diesem Artikel geben wir einen Überblick über NFC und zeigen, wie diese Technologie über die Android APIs verwendet werden kann.

Dominik Gruntz | dominik.gruntz@fhnw.ch

Die Einsatzgebiete von NFC sind vielfältig. Daten, die auf einem NFC-Tag abgespeichert sind, können durch Berührung mit einem Mobiltelefon ausgelesen werden. Bei den ausgelesenen Daten kann es sich um eine URL, eine Telefonnummer, ein Bild, eine Geo-Koordinate, eine Visitenkarte (vCard) oder um applikationsspezifische Daten handeln. Die Daten lassen sich dann direkt auf dem Gerät weiterverarbeiten, ein mühsames Abtippen von URLs oder Fotografieren von QR-Tags entfällt. So wird es beispielsweise möglich sein, alleine durch Berührung mit dem Mobiltelefon die Bedienungsanleitung für den Geschirrspüler abzurufen und auf dem Display anzuzeigen. Tags an den Regalen im Supermarkt informieren über Inhalte von Lebensmitteln, und mit Hilfe der ausgelesenen Produktnummer könnten nützliche Zusatzinformationen aus unabhängigen Quellen angezeigt werden, zum Beispiel eine persönliche Allergiewarnung. Auch das Selfscanning (wie zum Beispiel passabene von Coop [Coop08] oder subito von Migros [Migr11]) könnte damit realisiert werden. NFC-Mobiltelefone wirken damit als Browser für das Internet der Dinge [MF10], d.h., es wird möglich, jedes Objekt, jede Person und jeden Ort mit Online-Dokumenten auf dem Web zu verknüpfen.

Die Anwendung von NFC ist vergleichbar mit dem Einlesen von 2D-Barcodes, ausser dass die Handhabung viel einfacher ist. Es muss nicht speziell eine Applikation wie der Barcode-Scanner gestartet werden, und die Kamera muss auch nicht auf ein Tag ausgerichtet werden. Es genügt, den NFC-Tag mit dem Mobiltelefon zu berühren. Dieses erkennt den Tag automatisch und startet eine Applikation zur Verarbeitung der übertragenen Daten. Mit einem NFC-Mobiltelefon kann im Prinzip auch ein Tag emuliert werden, welches von einem Lesegerät über die NFC-Schnittstelle ausgelesen oder beschrieben werden kann. Häufig handelt es sich dabei um Tags mit erweiterter Funktionalität, sogenannte Smartcards. Damit kann kontaktloses Bezahlen realisiert werden, wie es heute in vielen Ländern bereits in Form kontaktloser Kreditkarten genutzt wird.

All diese Anwendungsfälle haben wir im Rahmen des Projektes touch'n'pay (www.touchnpay.ch) umgesetzt und einen Hofladen so ausgerüstet, dass das Einkaufen mit NFC-fähigen Mobiltelefonen möglich wird. Die Regale haben wir mit NFC-Produktetiketten ausgezeichnet. Wenn der Kunde sein Mobiltelefon an ein solches Produktetikett hält, wird das Produkt automatisch auf seinem elektronischen Kassenzettel auf dem Mobiltelefon eingetragen. Sobald alle gewünschten Produkte in der Einkaufstasche liegen, muss nur noch ein Checkout-Tag berührt oder das Zahlen-Menü auf dem Mobiltelefon gewählt werden, um den Bezahlvorgang auszulösen. Das Mobiltelefon haben wir ausserdem verwendet, um auch ausserhalb der Öffnungszeiten den Zugang zum Hofladen zu ermöglichen. Das Türschloss greift dabei über NFC auf im Telefon abgespeicherte Daten zu [BG10].

In diesem Artikel geben wir eine kurze Einführung in NFC und zeigen dann, wie mit Android über NFC kommuniziert werden kann. Einen guten Überblick über die NFC Technologie findet man im kürzlich erschienenen Buch von Josef Langer und Michael Roland [LR10].

Betriebsarten

NFC unterscheidet folgende Betriebsarten:

- *Reader/Writer-Modus:* In dieser Betriebsart wird das NFC-Mobiltelefon zum Leser und kann passive NFC-Tags auslesen und mit Daten beschreiben.
- *Card-Emulation-Modus:* In dieser Betriebsart ist das NFC-Mobiltelefon passiv und emuliert ein Tag, typischerweise eine Smartcard. Ein RFID-Leser, z.B. ein Kassensystem oder ein Türschloss, greift auf das im NFC-Mobiltelefon emulierte Smartcard-Tag zu.
- *Peer-to-peer-Modus:* Die peer-to-peer Betriebsart ermöglicht es, Informationen zwischen zwei (aktiven) Geräten auszutauschen. Es

kann sich dabei um einen Gutschein handeln oder um gegenseitige Identifikationen, damit grössere Datenmengen danach einfach über Bluetooth oder über das Internet ausgetauscht werden können.

Android unterstützt ab der Version 2.3.4 den Reader/Writer-Modus und (eingeschränkt) den Peer-to-peer-Modus. Die Peer-to-peer Kommunikation erlaubt nur den Austausch einzelner Meldungen, so wie sie auch auf Tags gespeichert sind. Es ist jedoch nicht möglich, über NFC eine Socketverbindung zwischen zwei Android-Geräten aufzubauen.

Der Card-Emulation-Mode wird von den Android APIs nicht unterstützt. Der Zugriff auf das sogenannte Secure-Element (SE), auf welchem (wie in einer Smartcard) Daten und Programme sicher abgelegt werden können, ist damit aus einer Android-Applikation nicht möglich.

Die Hardware

NFC wird aktuell von den Smartphones *Nexus S, Galaxy SII* und *Galaxy Nexus* von Samsung unterstützt. Im Nexus S ist der PN544 NFC-Controller von NXP eingebaut [PN544]. Dieser Controller unterstützt alle Betriebsarten und enthält ein integriertes SE (SmartMX Security Chip), unterstützt jedoch über das Single Wire Protocol (SWP) auch den Zugriff auf ein auf einer erweiterten SIM Karte (UICC) abgelegtes SE. Das Galaxy SII enthält kein integriertes SE, sondern sieht nur den Zugriff via SWP auf ein SE auf der SIM Karte vor. Im Galaxy Nexus ist der PN65N NFC-Controller von NXP verbaut. Dieser Controller ist von der Funktionalität her identisch mit jenem im Nexus S.

Es ist also nur eine Frage der Zeit, bis der Zugriff auf das SE auch über das Android-API möglich ist. Für diesen Zugriff ist jedoch ein Schlüssel nötig und diesen kennen aktuell nur Google und NXP. Die Frage, wie dieser Schlüssel an die Entwickler verteilt wird, ist hingegen unbeantwortet.

Die Software

Konzentrieren wir uns daher auf das, was mit den heutigen Geräten möglich ist: das Lesen und Schreiben von NFC-Tags, sowie der Austausch von Meldungen zwischen Geräten. Die NFC Data Exchange Format (NDEF) Spezifikation [NDEF] legt die Formate für den Austausch von Informationen zwischen NFC-Geräten und NFC-Tags fest. Anwendungsdaten sind (zusammen mit Metainformationen) in einem oder mehreren NDEF-Records abgelegt, welche zusammen eine NDEF-Meldung bilden. Die Records ermöglichen die Repräsentation von Datenpaketen in verschiedenen Datenformaten. Ein Tag kann in einer NDEF-Meldung z.B. eine URI, ein Text und ein Icon enthalten. Wie Daten in den Records repräsentiert und auf den Geräten interpretiert werden, legt die *NFC Record Type Definition* (RTD) Spezifikation fest.

In Android sind die nötigen Datentypen im Paket *andorid.nfc* definiert. Die Klasse *NdefMessage* repräsentiert eine NDEF-Meldung, welche einen oder mehrere NDEF-Records enthält. In diesen Records sind die Nutzdaten abgelegt. Ein NDEF-Record besteht aus einem Header und einem Datenteil. Im Header sind neben Flags Informationen zum Typ der Nutzdaten, die Länge der Nutzdaten sowie optional eine eindeutige Kennung des Records abgelegt. Android bildet einen NDEF-Record in der Klasse *NdefRecord* ab.

Für die Beschreibung des Typs eines Records sind verschiedene Formate vorgesehen. Der Wert des Feldes *Type Name Format* (TNF) gibt an, in welchem Format der Typ codiert ist. Die TNF-Werte sind in der NDEF-Spezifikation (und als Konstante in der Klasse *NdefRecord* definiert. In Tabelle 1 ist die Bedeutung der verschiedenen TNF-Werte erklärt.

Neben den beiden Klassen *NdefMessage* und *NdefRecord* enthält das Paket *android.nfc* auch noch die Klassen *NfcManager* und *NfcAdapter*. Mit dem Manager kann auf den Adapter zugegrif-

TNF Wert		Bedeutung
TNF_EMPTY	0	Dieses Format zeigt an, dass der Record leer ist, d.h., er hat weder Typ, Kennung noch Nutzdaten. Die entsprechenden Getter-Methoden auf dem Record geben leere Arrays zurück.
TNF_WELL_KNOWN	1	Dieses Format zeig an, dass der Typ als NFC Forum Well-known Type codiert ist (entsprechend der Record Type Definition). Beispiel: „T“ steht für einen Text-Record, „U“ für eine URI oder „Sp“ für einen Smart-Poster-Record. Die wichtigsten Typen sind in den Konstanten NdefRecord.RTD_TEXT, RTD_URI und RTD_SMART_POSTER abgelegt.
TNF_MIME_MEDIA	2	Records mit diesem TNF enthalten einen Typ-String im MIME-Format nach RFC 2046, z.B. „image/png“ für ein Bild oder „text/x-vCard“ für eine Visitenkarte.
TNF_ABSOLUTE_URI	3	Wenn diese Konstante gesetzt ist, dann enthält das Typfeld eine absolute URI nach RFC 3986. Dieser URI definiert dann das Format der Nutzdaten.
TNF_EXTERNAL_TYPE	4	Dieses Format gibt an, dass das Typfeld ein NFC Forum External Type ist. Firmen können damit anwendungsspezifische Record-Typen definieren.
TNF_UNKNOWN	5	Wenn dieses Flag gesetzt ist, dann enthält der Record Daten in einem unbekannten Format. Der Typ-String ist in diesem Fall leer.
TNF_UNCHANGED	6	Ein TNF mit Wert 6 gibt an, dass die Nutzdaten auf mehrere Records verteilt sind (analog zu chunked encoding von HTTP-Payloads). Der Typ der Nutzdaten wird auf dem ersten Record spezifiziert, auf den nachfolgenden Records ist TNF=6 gesetzt und das Typfeld ist leer.

Tabelle 1: Bedeutung der TNF-Werte gemäss NDEF-Spezifikation

```
   <intent-filter>
      <action android:name="android.nfc.action.NDEF_DISCOVERED"/>
      <category android:name="android.intent.category.DEFAULT"/>
      <data android:mimeType="application/vnd.fhnw.coupon" />
   </intent-filter>
```

Listing 1: Intent-Filter für das Lesen von NDEF-Meldungen

```
byte[] getCodeFromNdefMessages(Intent intent) {
   byte[] payload = null;
   if (NfcAdapter.ACTION_NDEF_DISCOVERED.equals(intent.getAction())) {
      Parcelable[] rawMsgs = intent.getParcelableArrayExtra(
                                 NfcAdapter.EXTRA_NDEF_MESSAGES );
      if (rawMsgs != null && rawMsgs.length > 0) {
         NdefMessage msg = (NdefMessage)rawMsgs[0];
         NdefRecord[] recs = msg.getRecords();
         if(recs.length > 0) payload = recs[0].getPayload();
      }
   }
   return payload;
}
```

Listing 2: Auslesen der Daten aus einem Intent

fen werden, und der Adapter enthält Methoden für den Peer-to-peer-Modus sowie eine Methode *isEnabled* mit welcher geprüft werden kann, ob NFC in den Einstellungen für Drahtlosnetzwerke aktiviert ist.

Lesen von NFC-Tags

Sobald Android ein NFC-Tag erkennt, löst es einen entsprechenden Intent aus, der die eingelesenen Daten enthält. Falls es sich um einen Tag mit einer NDEF-Meldung handelt, dann wird ein Intent mit der Aktion *NDEF_DISCOVERED* verschickt. Falls der Tag keine Meldung im NDEF-Format enthält (oder falls der entsprechende Intent von keiner Activity behandelt worden ist), so wird ein Intent mit der Aktion *TECH_DISCOVERED* verschickt. Über das im Intent enthaltene Tag-Objekt kann geprüft werden, welche Protokolle dieser Tag unterstützt. Alle Android NFC-Geräte müssend die folgenden Technologien unterstützen:

- NfcA (ISO/IEC 14443-3 Typ A)
- NfcB (ISO/IEC 14443-3 Typ B)
- NfcF (JIS 6319-4, auch bekannt unter dem Namen FeliCa)
- NfcV (ISO 15693, auch bekannt unter dem Namen Vincinity)
- IsoDep (ISO 14443-4, APDU basierte Smart-Cards)
- Ndef (Tags, welche einem der vom NFC Forum definierten Tag-Typen entsprechen)

Das Nexus-S unterstützt daneben noch spezielle Protokolle für Mifare Karten. Falls der Tag keines dieser Protokolle unterstützt (oder falls der entsprechende Intent von keiner Activity behandelt worden ist), so wird ein Intent mit der Aktion *TAG_DISCOVERED* verschickt. Falls mehrere Aktivitäten auf einen Intent reagieren können, so muss der Benutzer wählen, welche Aktivität er verwenden will.

Falls eine Activity auf einen dieser Intents reagieren will, so muss ein passender Intent-Filter definiert werden. In Listing 1 ist angegeben, welchen Filter eine Activity deklarieren muss, um NFC-Tags lesen zu können, welche eine NDEF-Meldung mit einem speziellen MIME-Typ enthalten (hier *application/vnd.fhnw.coupon*).

Auf dem Intent, den die Activity erhält, können Zusatzinformationen ausgelesen werden:

- EXTRA_ID: Identifikationsnummer des Tags
- EXTRA_TAG: ein Objekt, welches den Tag repräsentiert (und zusätzliche Operationen anbietet)
- EXTRA_NDEF_MESSAGES: NDEF-Meldungen (falls es sich um einen NDEF-Tag handelt)

In Listing 2 wird der Inhalt (Payload) des ersten NDEF-Records der ersten NDEF-Meldung eines Intents ausgelesen und im Binärformat zurückgegeben.

```
   <intent-filter>
      <action android:name="android.nfc.action.TECH_DISCOVERED" />
   </intent-filter>

   <meta-data android:name="android.nfc.action.TECH_DISCOVERED"
      android:resource="@xml/filter_nfc"
   />
```

Listing 3: Intent-Filter für die Kommunikation mit NDEF-Tags

```
<resources xmlns:xliff="urn:oasis:names:tc:xliff:document:1.2">
   <tech-list>
      <tech>android.nfc.tech.Ndef</tech>
   </tech-list>

   <tech-list>
      <tech>android.nfc.tech.NdefFormatable</tech>
   </tech-list>
</resources>
```

Listing 4: Definition einer Tag-Technologie Liste

Schreiben von NFC-Tags

Um einen Tag mit dem Handy zu beschreiben, muss eine Verbindung zu diesem aufgebaut werden. Typischerweise wird dazu ein Intent-Filter vom Typ *TECH_DISCOVERED* definiert (Listing 3). Die Technologien, die vom Programm verarbeitet werden können, müssen in einer separaten Datei spezifiziert werden, auf die mit einem *meta-data* Tag verwiesen wird. Die Liste in Listing 4 spezifiziert, dass sie Tags vom Typ *Ndef* oder *NdefFormatable* beschreiben kann.

Listing 5 zeigt, wie ein Tag, der mit dem Intent-Filter in Listing 3 und 4 erkannt wurde, beschrieben wird. Dazu wird die Tag-Instanz aus dem Intent in den Typ *Ndef* oder *NdefFormatable* konvertiert, damit mit der Methode *writeNdefMessage* bzw. *format* eine NDEF-Meldung auf den Tag geschrieben werden kann. Die Methode *createRtdUriRecord* haben wir bereitgestellt. Diese Hilfsfunktion erzeugt einen NDEF-Record mit TNF=1 (WELL-KNOWN) und Typ=„U" (URI) und codiert den Prefix der URL gemäss Spezifikation.

In der Applikation weist man den Benutzer darauf hin, dass der Tag, der als nächstes berührt wird, beschrieben wird. Damit nun nicht andere Applikationen ebenfalls auf den Intent reagieren, der beim Berühren dieses Tags verschickt wird, kann man mit der Methode *enableForegroundDispatch* auf dem NFC-Adapter sicherstellen, dass nur die eigene Activity den Intent erhält.

Peer-To-Peer

Seit der Version 2.3.3 unterstützt Android auch die Peer-to-peer-Kommunikation zwischen zwei Geräten. Damit kann eine NDEF-Meldungen von Gerät zu Gerät übertragen werden. Ab der Version *Ice Cream Sandwich* von Android ist auch das Live-Pushing zwischen zwei Geräten möglich (Android Beam). Wenn man z.B. auf einem Gerät mit YouTube einen Film betrachtet und dann ein anderes NFC-Gerät berührt, so läuft der Film auf dem anderen Gerät an derselben Stelle weiter. Was jedoch auch mit *Ice Cream Sandwich* noch nicht unterstützt wird, ist die direkte Kommunikation über LLCP (NFC Logical Link Control Protocol).

Mit der Methode *enableForegroundNdefPush* kann eine Activity eine NDEF-Meldung via Peer-to-peer publizieren und mit *disableForegroundNdefPush* die Publikation wieder deaktivieren. Listing 6 zeigt, wie in den Methoden *onResume* und *onPause* die Meldung *pushMessage* publiziert wird. In einem zweiten Handy wird diese NDEF-Meldung über normale Intents publiziert.

```
void writeUrlToTag(Intent intent, String url) throws IOException,
      FormatException {
   String action = intent.getAction();
   if (NfcAdapter.ACTION_TECH_DISCOVERED.equals(action)) {
      NdefRecord rec = NdefRecordRtdUri.createRtdUriRecord(url);
      NdefMessage msg = new NdefMessage(new NdefRecord[] { rec });

      Tag tag = intent.getParcelableExtra(NfcAdapter.EXTRA_TAG);
      Ndef ndef = Ndef.get(tag);
      if (ndef != null) {
         ndef.connect();
         ndef.writeNdefMessage(msg);
         ndef.close();
      } else {
         NdefFormatable ndeff = NdefFormatable.get(tag);
         if (ndeff != null) {
            ndeff.connect();
            ndeff.format(msg);
            ndeff.close();
         }
      }
   }
}
```

Listing 5: Methode welche eine URL auf einen Tag schreibt

```
public void onResume() {
   super.onResume();
   if (nfcAdapter != null)
      nfcAdapter.enableForegroundNdefPush(this, pushMessage);
}
public void onPause() {
   super.onPause();
   if (nfcAdapter != null)
      nfcAdapter.disableForegroundNdefPush(this);
}
```

Listing 6: Aktivierung und Deaktivierung von NDEF-Push

Anwendungen

Die auf dem Nexus S vorinstallierte *Tags*-Applikation unterstützt die Verwaltung von Tags. Wird z.B. ein mit dem Programm in Listing 5 beschriebener Tag berührt und vom Gerät erkannt (vorausgesetzt, NFC ist in den Einstellungen für Drahtlosnetzwerke aktiviert worden), so wird der darauf gespeicherte Link angezeigt (siehe Abb. 1) und kann mit einem Fingerklick weiterverarbeitet werden. Auf der Seite *Mein Tag* kann eingestellt werden, welcher Inhalt mit NDEF-Push publiziert werden soll (siehe Abb. 2).

Im Android Markt findet man weitere Anwendungen, welche NFC verwenden. Mit der Applikation *NFC Tag Info* können detaillierte Informationen zu Tags abgefragt werden. Abbildung 3 zeigt, wie diese Applikation die mit Listing 5 geschriebene NDEF-Meldung darstellt. Eine Applikation, welche das Beschreiben von Tags unterstützt, ist der *NXP Tag Writer*.

Mit der Applikation *WiFiTap* können WIFI-Konfigurationen auf einen Tag gespeichert und geladen werden (mit Netzname, Typ und Passwort). Die Applikation *NFC TaskLauncher* erlaubt es, Aktionen auf einem Tag abzuspeichern, die dann bei Berührung des Tags ausgeführt werden. Als Aktionen können z.B. Konfigurationseinstellungen (Lautstärken und Klingeltöne) vorgenommen werden, Alarme gestellt werden, Applikationen gestartet werden, etc. Ein solcher Tag könnte z.B. im Auto, am Arbeitsplatz oder im Sitzungszimmer deponiert werden, um entsprechende Einstellungen vornehmen zu können.

Taglet ist eine Applikation die es erlaubt, im Internet Informationen zu einem Tag zu hinterlegen. Diese Information wird dann angezeigt, wenn der Tag mit einem anderen Gerät ausgelesen wird. Auf diese Weise lassen sich Tags annotieren. Die Applikation *Enable Table* erlaubt es, Besuchern eines Restaurants Rabatt-Gutscheine abzugeben. Der NFC-Tag ist dabei im Rechnungsetui eingebaut. Aus der Beschreibung im Android Markt und im Internet geht jedoch nicht eindeutig hervor, wie diese Gutscheine dann eingelöst werden können.

Damit der Android Markt eine Applikation nur dann anzeigt, wenn das Gerät NFC unterstützt, muss dem Manifest das folgende uses-feature hinzugefügt werden: *<uses-feature android:name=„android.hardware.nfc" android: required=„true">*.

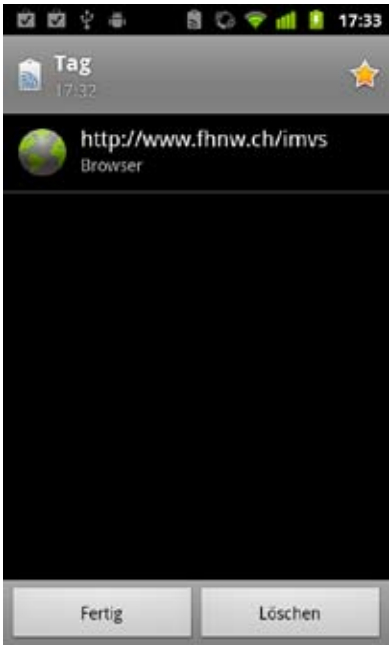

Abbildung 1: Nexus S Tags Applikation

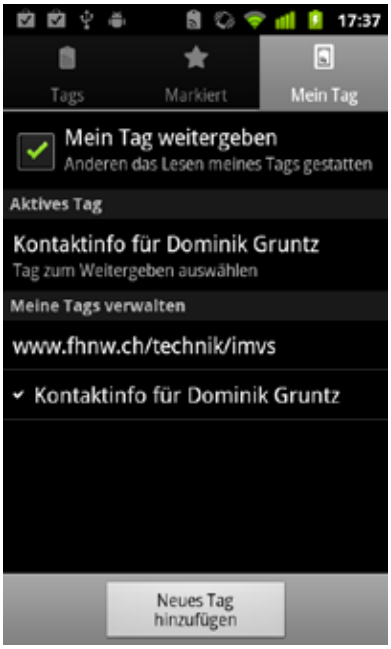

Abbildung 2: Weitergeben von Tags

Abbildung 3: NFC Tag Info App

Zusammenfassung

Die NFC-APIs und das Nexus S sind ein erster Schritt in Richtung Mobile Payment und Mobile Ticketing. Mit den vorliegenden Android-Versionen lässt sich diese Vision jedoch noch nicht realisieren, da dazu wichtige Funktionen noch fehlen. Der im Nexus S und Galaxy Nexus eingebaute NFC-Controller unterstützt jedoch die volle Funktionalität, daher ist es nur eine Frage der Zeit, bis Google auch die Bibliotheken auffrischt. Unklar ist auch, wie auf das SE zugegriffen werden kann, d.h., wie man die dazu nötigen kryptographischen Schlüssel erhält. Google Wallet zeigt, dass dies grundsätzlich möglich ist.

Die im Paket *android.nfc* definierten Klassen sind sehr spartanisch gehalten. Es wäre wünschenswert, wenn für die in den NDEF- und RTD-Spezifikationen definierten Typen eigene Klassen definiert wären. Es ist nicht einsehbar, warum jeder Programmierer die Tags erneut parsen soll. Aber auch in diesem Punkt darf damit gerechnet werden, dass hier in einer zukünftigen Version von Android nachgebessert wird.

Eines steht jedoch fest: NFC ist eine wichtige Technologie, deren Verbreitung und Akzeptanz durch die Initiative von Google nun rasch zunehmen wird. Ob 2011 als NFC Jahr in die Geschichte eingehen wird? Man hat in der Vergangenheit schon oft ein NFC Jahr ausgerufen, ohne dass Taten gefolgt sind. Google hat inzwischen erreicht, dass NFC in aller Munde ist, aber leider noch nicht, dass es auch in allen Mobiltelefonen verfügbar ist. Wenn Apple dereinst doch noch ein iPhone 5 mit NFC angekünden wird, dann ist NFC nicht mehr aufzuhalten.

Referenzen

[BG10] Ingo Bauersachs, Dominik Gruntz; Touch'n pay: ein NFC-Feldversuch, IMVS Fokus-Report, p. 37-42, 2010.

[Coop08] Coop Supermarkt, passabene für das praktische Einkaufen, 2008, http://tinyurl.com/passabene.

[LR10] Josef Langer, Michael Roland, Anwendungen und Technik von Near Field Communication (NFC), Springer Verlag, Sept 2010.

[MF10] Friedemann Mattern, Christian Flörkemeier, Vom Internet der Computer zum Internet der Dinge, Informatik Spektrum, Vol. 33, No. 2, S. 107-121, 2010.

[Migr11] Subito – einfach und schnell einkaufen, 2011, http://www.migros.ch/subito.

[NDEF] NFC Forum, NFC Data Exchange Format (NDEF), Rev. 1.0. Technical Specification, Jul. 2006, http://www.nfc-forum.org/specs/spec_list/

[PN544] NXP NFC Controller PN544 http://www.nxp.com/acrobat_download2/literature/9397/75016890.pdf