



Sztuka i nauka

Anne-Marie Lesas
Serge Mirende

WI LEV

Sztuka i nauka programowania NFC

Zestaw technologii intelektualnych

koordynowany przez
Jean-Max Noyer i Maryse Carmes

Tom 3

Sztuka i nauka
Programowanie
NFC

Anne-Marie Lesas
Serge Miranda

ISTE

WILEY

Opublikowany po raz pierwszy w 2017 r. w Wielkiej Brytanii i Stanach Zjednoczonych przez ISTE Ltd i John Wiley & Sons, Inc.

Z wyjątkiem uczciwego obrotu do celów badawczych lub prywatnych studiów, krytyki lub recenzji, dozwolonych na mocy ustawy o prawie autorskim, wzorach i patentach z 1988 r., niniejsza publikacja może być powielana, przechowywana lub przekazywana, w jakikolwiek formie lub w jakikolwiek sposób, wyłącznie za uprzednią pisemną zgodą wydawców lub w przypadku reprodukcji reprograficznej zgodnie z warunkami i licencjami wydanymi przez CLA. Zapytania dotyczące powielania poza tymi warunkami należy kierować do wydawców na podany poniżej adres:

ISTE
27-37 St George's
London SW19 4EU
Hoboken
WIELKA BRYTANIA
www.iste.co.uk

Ltd John Wiley & Sons, Inc.
Road 111 River Street
, NJ 07030
USA
www.wiley.com

© ISTE Ltd 2017

Prawa Anne-Marie Lessas i Serge'a Mirandy do bycia identyfikowanymi jako autorzy tej pracy zostały przez nich potwierdzone zgodnie z ustawą o prawach autorskich, wzorach i patentach z 1988 roku.

Numer kontrolny Biblioteki Kongresu: 2016954190

British Library Cataloguing-in-Publication Data
Rekord CIP dla tej książki jest dostępny w British Library ISBN
978-1-78630-057-7

Zawartość

Przedmowa	vii

Przedmowa	xi

	xv
Wprowadzenie	1
.....
Rozdział 1. Stan wiedzy na temat NFC	
.....
1.1. Przyszłe mobilne usługi cyfrowe	2
1.1.1. Era mobilności	3
1.1.2. W kierunku świata bezkontaktowego komunikowanie obiektów	6
1.2. Sprzęt NFC	7
1.2.1. Znacznik NFC	7
1.2.2. Karta inteligentna NFC.	8
1.2.3. NFC smartphone	13
1.2.4. Czytnik/koder: Terminale transakcyjne NFC	14
1.2.5. "Inteligentne miasta" i zrównoważony rozwój.	14
1.2.6. Płatność bezgotówkowa za pomocą NFC	15
1.3. Standardy NFC	16
1.3.1. Sygnał analogowy i transpozycja cyfrowa NFC	18
1.3.2. Trzy standardowe tryby NFC	21
1.3.3. Standardy forum NFC	25

1.3.4. GlobalPlatform (GP)	36
1.3.5. SIMAlliance i otwarte mobilne API	42

Rozdział 2. Rozwój NFC

Aplikacje na Androïda45

2.1. Wprowadzenie do programowania w systemie Android przy użyciu Eclipse	46
2.1.1. Android w pigułce	46
2.1.2. Android w Eclipse IDE.....	49
2.1.3. Intencje i kontekst systemu Android.....	60
2.1.4. Klasa aktywności systemu Android	61
2.1.5. Interfejs graficzny systemu Android: pliki "layout".....	64
2.1.6. Kompilowanie i testowanie aplikacji na Androïda.....	67
2.2. Wdrażanie NFC w systemie Android	70
2.2.1. Deklaracje manifestu systemu Android.....	71
2.2.2. Wdrażanie trybu czytnika/zapisu NFC	71
2.2.3. Wdrażanie trybu NFC P2P w systemie Android	83
2.2.4. Wdrażanie emulacji karty NFC tryb z Androïdem.....	87
2.2.5. Tworzenie usług NFC z Androïd HCE.....	97

Rozdział 3. Przypadki użycia NFC107

3.1. Korzystanie z trybu czytnika/zapisu NFC	107
3.1.1. Przypadek użycia: zarządzanie pożyczkami na sprzęt	108
3.2. Korzystanie z trybu NFC P2P	112
3.2.1. Przypadek użycia: Parowanie NFC	112
3.3. Korzystanie z trybu emulacji karty NFC	114
3.3.1. Przypadek użycia: portfel cyfrowy w SE	115
3.4. Korzystanie z trybu HCE	118
3.4.1. Przypadek użycia: SE w chmurze z HCE.....	119

Wnioski.....121

Bibliografia.....125

Indeks129

Przedmowa

"Główną zasadą jest zadowolenie i dotyk.

Wszystkie inne są tworzone tylko po to, by osiągnąć ten pierwszy".

MOLIERE

Mimo że standard NFC jest młody (opracowany w 2004 r.), od kilku lat jestem proszony o napisanie tej książki ze względu na pionierską rolę, jaką we Francji (i w Europie) odegrał nasz magister informatyki MBDS (www.mbdss-fr.org) na Uniwersytecie w Nicei - Sophia-Antipolis w zakresie prototypowania innowacyjnych usług wykorzystujących ten standard. MBDS prototypował usługi NFC we wszystkich sektorach życia gospodarczego: od turystyki i kultury w Nicei, do płatności społecznych w Indiach, poprzez kampusy na Haiti, muzea, lotniska, hotele, połączone domy w Maroku i samochody elektryczne w Sophia Antipolis. W 2009 r. Nicea była pierwszym miastem w Europie, w którym wdrożono standard NFC, dzięki laboratorium badawczemu MBDS.

Po pół tuzinie opublikowanych książek na temat baz danych, nie byłem zbyt chętny do napisania nowej książki. Moja pierwsza kolekcja książek, po książce Knutha (*The Art of Computer Programming*), która była moją lekturą podczas studiów w Kalifornii, nosiła tytuł "*The Art of Databases*".

To, co zmieniło moje zdanie, to entuzjazm Anne-Marie Lesas, która pracowała nad doktoratem na temat bezpiecznych usług NFC z naszym partnerem przemysłowym Gemalto w ramach konwencji CIFRE¹ z ANRT² oraz kontraktu IFCPAR³ (www.cefipra.org) na temat wirtualnej waluty społecznej NFC w Indiach z TATA Consultancy Services (CS) i Bangalore University, a także eksperzyz naukowej dotyczącej naruszenia patentu NFC w Stanach Zjednoczonych w 2015 roku. Jako błyskotliwa była studentka MBDS po zakończeniu kariery zawodowej, Anne-Marie po raz pierwszy wykazała pasję do nowych technologii mobilnych (samochody NFC i sposoby wykrywania trzęsień ziemi za pomocą czujników smartfonów). Co za rozkosz dla profesora, który jest niczym innym jak dostawcą marzeń, widzieć, jak student przejmuje kontrolę.

Tytuł tej książki sugeruje dwoistość "Sztuki" i "Nauki", które są dwoma podejściami do postrzegania i rozumienia świata; w przedmowach do książek o bazach danych napisałem: "słowo *sztuka* odnosi się do sposobu badania, odtwarzania i interpretacji świata rzeczywistego w opozycji do *nauki*, która niesie abstrakcyjną interpretację, opartą na formalnych koncepcjach, modelach i narzędziach". Kreatywność w aplikacjach NFC jest nieograniczona dzięki mobilnym zastosowaniom, które na nowo odkrywają świat rzeczywisty, tworząc nowe mosty do świata wirtualnego; aplikacje te opierają się na ściśle znormalizowanych koncepcjach, które wyjaśniamy wraz z metodami ich wdrażania. Radzenie sobie z tą dwoistością jest podwójnym celem tej książki.

W ten sposób ta książka jest wynikiem pedagogicznego spotkania profesora i badacza, aby umożliwić innym programistom IT przyczynienie się do zmiany świata poprzez dotknięcie go! Chciałbym zatem podziękować Anne-Marie za jej profesjonalne i ludzkie umiejętności, a za jej pośrednictwem wszystkim studentom, którzy swoim entuzjazmem prowadzą mnie w kierunku niekończącego się, spiralnego procesu innowacji. Kreatywność w zakresie treści i usług to piękna spiralna przygoda, która ożywia

1 Kontrakty przemysłowe na szkolenia poprzez badania.

2 Francuskie stowarzyszenie Association Nationale de la Recherche et de la Technologie.

3 Indyjsko-Francuskie Centrum Promocji Zaawansowanych Badań.

(Specjalne podziękowania dla Franketienne za tę piękną koncepcję *spiralizmu*, którą podzieliliśmy się i omówiliśmy w Port au Prince).

Chciałbym zaprosić czytelników do marzeń o swoim życiu, do wielkich marzeń, pamiętając, że nowe technologie muszą przede wszystkim służyć dobru ludzkości i poprawie wspólnego środowiska oraz życia każdego z nas. "Zawsze stawiaj człowieka w centrum" i nie wahaj się być "*nonkonformistą, a nawet anarchistą innowacji*", jak pokazał Pierre Laffite, założyciel parku naukowego Sophia-Antipolis.

Myszę też o moim przyjacielu i koledze, pionierze wszelkiego rodzaju baz danych (i Big Data), Mike'u Stonebrakerze, ojcu chrzestnym jednej z pierwszych klas MBDS i zdobywcy Nagrody Turinga w 2014 roku, który zawsze podkreślał badania stosowane w systemach informatycznych, z obsesją na punkcie rozwiązywania konkretnych problemów, a nie tylko trzymania się prostych teoretycznych konstrukcji intelektualnych oderwanych od rzeczywistości.

Ta książka ma podwójny cel, który odpowiada jej dwóm głównym częścioom; ma na celu zarówno:

- Wyczerpujące, teoretyczne podejście do standardu NFC, nieuniknionego w przyszłości smartfonów, zarówno w świecie informacyjnym, jak i transakcyjnym;
- pragmatyczne i systematyczne podejście do rozwoju aplikacji NFC, oparte na licznych prototypach innowacyjnych usług stworzonych w ramach naszego programu MBDS Master's od momentu narodzin standardu w 2004 roku.

Ta książka jest przeznaczona dla inżynierów IT (zarówno specjalistów IT, jak i studentów studiów licencjackich lub magisterskich), którzy pasjonują się nowymi technologiami i są ciekawi wykorzystania NFC, szczególnie w aplikacjach mobilnych; naszym celem jest wyjaśnienie specyfiki technicznej i funkcjonalnej standardu NFC poprzez pojęcia niezbędne do zrozumienia ekosystemu, jego mobilnej implementacji (z Androidem) i jego głównych zastosowań.

Dzięki tej książce mamy nadzieję dać czytelnikowi autonomię, aby mógł projektować i rozwijać własne aplikacje NFC. *Innowacja to wynalazek spotykający się z zastosowaniem:* wprowadzaj innowacje za pomocą tego wynalazku, wyobrażając sobie nowe praktyki.

Na zakończenie tej przedmowy zapożyczę dwa cytaty, których często używam na zakończenie moich konferencji:

– "Nigdy nie zapominaj, że mrówka może unieść słonia" - to słowa mojej przyjaciółki, siostry Flory, która od ponad 30 lat zarządza sierocińcem na południu Haiti, w Ile-à-Vaches, zgodną podziwu miłością i kreatywnością. Dziś ktoś ze smartfonem w ręku ma większą moc obliczeniową niż komputer użyty w misji Apollo i dostęp do informacji większy niż kiedykolwiek miał prezydent Kennedy. Smartfon jest "obiektem świata", jak rozumiał to słowo Michel Serres, i nie możemy zapominać, że "nawet myśl mrówki może dotknąć nieba" (japońskie przysłowie);

– "Jeśli nie możesz zmienić świata, spróbuj zmienić SWÓJ świat" (ostatnie zdanie Karola Marks'a), co jest podstawą życia *we wspólnocie* w czasie przyszłym i rzeczywistości, która jest nie tylko mobiquitous.

Ciesz się tą książką i ciesz się programowaniem NFC, życzę Ci, abyś stał się mrówką *komunikacyjną*, otwartą na świat, aby go zmienić.

Profesor Serge MIRANDA
2016

Przedmowa

"Podstawową formą zmysłu jest dotyk, który należy do wszystkich zwierząt. [...]

Zmysł dotyku jest koniecznie tym, którego utrata powoduje śmierć żywych istot".

ARYSTOTELES

Dzięki standardowi *komunikacji bliskiego zasięgu* (NFC) telefon komórkowy obsługujący NFC zyskuje zmysł dotyku Arystotelesa.

NFC to globalny standard komunikacji bezstykowej i w bardzo krótkim polu (zbliżeniowej) (kilka centymetrów) stworzony przez firmy Philips, Sony i Nokia w 2004 roku (trzech głównych graczy i liderów w dziedzinie elektroniki użytkowej). Standard NFC jest jednym z 16 standardów *identyfikacji radiowej*, który zapewnia unikalną identyfikację każdego oznaczonego obiektu znanego od lat czterdziestych XX wieku oraz bezprzewodowy odczyt (za pomocą częstotliwości radiowej). Obecnie NFC, który jest szeroko rozpowszechniony w kartach inteligentnych (dostępu, płatności i transportu), został powszechnie wybrany przez wszystkich producentów smartfonów od 2014 roku, umożliwiając tym samym nowe zastosowania telefonów komórkowych.

Ten standard NFC ma trzy tryby pracy: *czytnik/zapis, emulacja karty i peer-to-peer*; za pomocą prostego dotknięcia urządzenia obsługującego NFC

urządzenie, dotknięcie (stąd paradygmat *tap'n play*) tagu lub innego urządzenia obsługującego technologię NFC:

- zbierać informacje dzięki trybowi czytnika/zapisu NFC;
- połączyć się z innym urządzeniem i zainicjować połączenie (np. Bluetooth®, Wi-Fi, Li-Fi) dzięki trybowi peer-to-peer NFC;
- uwierzytelnianie, otwieranie drzwi lub płacenie, na przykład z powodu trybu emulacji karty.

Telefon komórkowy z obsługą NFC może być zatem postrzegany jako uniwersalne złącze, które zwiększa możliwości sensoryczne telefonu. Po mówieniu/słuchaniu, czytaniu i przeglądaniu (zdjęć, wiadomości tekstowych, postów w sieciach społecznościowych), dzięki NFC, telefony komórkowe pozwolą nam *dотyкаcь* w celu zatwierdzenia dostępu, uzyskania informacji, wymiany treści lub zapłaty. Ten tryb interakcji, który jest nieinwazyjny i intuicyjny, prowadzi do portfolio innowacyjnych usług.

"Smartfon z NFC wygrał bitwę o kieszeń"! Wszystko, co znajdowało się w kieszeniach lub torebce, będzie teraz miało swoją zdematerializowaną wersję w telefonie komórkowym: gotówka, karty debetowe i kredytowe, karty programów lojalnościowych, klucze, aparat fotograficzny, odtwarzacz MP3 itp. Pod koniec 2015 roku połowa naszej planety (3,5 miliarda ludzi) posiadała smartfona, przy stałym tempie wzrostu, tym bardziej, że na początku 2016 roku w Indiach pojawiły się reklamy. Połowa z tych smartfonów obsługuje technologię NFC. Oznacza to również, że 2 miliardy ludzi bez konta bankowego, którzy posiadają smartfony, będą mogli korzystać z usług finansowych: poza bankami pojawią się nowe podmioty zajmujące się płatnościami mobilnymi! *Bankier jest... w Twojej kieszeni!*

W biologii życie definiuje się jako parę: informacja i komunikacja. Dzięki prostemu dotknięciu, telefony z obsługą NFC nawiązują komunikację ze zdalnym serwerem zawierającym historię tego obiektu. Każdy obiekt z tagiem NFC dotknięty przez telefon komórkowy z obsługą NFC staje się w ten sposób, z biologicznego punktu widzenia, żywym obiektem. W przyszłości systemy informatyczne będą musiały uwzględniać ten aspekt obiektów, które stają się żywymi przedmiotami.

Standard NFC pozwala zatem przedmiotom stać się żywymi przedmiotami, a miejsca, w których się znajdują, stają się *inteligentnymi miejscami*: sprawdź wirtualną instrukcję obsługi urządzenia, automatycznie skonfiguruj środowisko i osobiste preferencje, uruchom zaplanowany program prania lub otwórz drzwi domu za pomocą smartfona za pomocą prostego dotknięcia, między innymi za pomocą NFC. Dzięki prostemu gestowi zbliżenia (mniej niż 1 cal), NFC wzmacnia w użytkowniku chęć interakcji i tworzy połaczenie między światem rzeczywistym a wirtualnym, tworząc rzeczywistość *rozszerzoną* lub *pominiejszoną*. Korzystanie z telefonów komórkowych z obsługą NFC jest jednym ze sposobów umożliwiających dokładną lokalizację użytkownika; w ten sposób możemy wyobrazić sobie portfolio geolokalizowanych, spersonalizowanych i kontekstowych usług.

Telefony komórkowe z obsługą NFC są wynikiem cyfrowej rewolucji, która stawia ludzi w centrum interakcji między światem rzeczywistym a światem wirtualnym: właściciel smartfona NFC potencjalnie staje się *Homo mobiquitus, komunikatorem*, tj. dostawcą danych (konsumentem i producentem) do wspólnej przestrzeni w trybie *oddolnym* [MIR 14a, MIR 14b].

Osiągnęliśmy nową erę systemów informacyjnych dzięki konwergencji telefonów komórkowych (które stały się komputerami dzięki smartfonom) i wszechobecności Internetu (który stał się społeczny dzięki łączom szerokopasmowym); podsumowujemy, według Xaviera Dallosa, słowem *mobiility*. Mobilność będzie miała wpływ na cały sektor gospodarczy i trzy poszczególne sfery (publiczną, prywatną i zawodową).

Wraz z NFC pojawiły się następujące nowe multidyscyplinarne koncepcje: *mobiility*, *Homo mobiquitus*, *communactors*, *one-tap marketing*, *mobiquitous tourism* (od dawnego "max min" do przyszłego "mini max"), *mobiquitous currency* (gotówka nie jest już najnowszym łączem płatniczym), *spiralist innovation*, "*Assistants Mobiquitaires InformationnelS (AMIS)*" (Mobiquitous Informational Assistants) lub "mobile cyber café" na Haiti.

NFC to przełomowa innowacja prowadząca do nowych usług i nowych architektur systemów informatycznych, które przynoszą nowe możliwości.

modele biznesowe. U podstaw myślenia o NFC leży *spiralna innowacyjność*. Ponieważ kreatywność w zakresie użytkowania jest nieograniczona, to od nas zależy wprowadzanie innowacji i tworzenie nowych aplikacji za pomocą tego globalnego standardu NFC działającego jako uniwersalny łącznik między światem rzeczywistym a światem wirtualnym w nieinwazyjny sposób: "the sky is the limit!".

Profesor Serge MIRANDA

Październik 2016 r.

Wprowadzenie

Oparte na identyfikacji radiowej przetestowanej podczas II wojny światowej, standardy komunikacji bliskiego zasięgu (NFC) stały się globalne w 2004 roku po kilku latach prototypowania i rozwoju.

Poza faktem, że jest to otwarty standard, NFC ma wiele zalet; jest to technologia ekonomiczna umożliwiająca dynamiczną interakcję wielomodelową, którą można podsumować za pomocą "3S": Bezpieczeństwo, Szybkość i Prostota. Wystarczy włączyć smartfon, aby wejść w interakcję ze światem rzeczywistym i wzbogacić go o wszystkie informacje ze świata wirtualnego.

Przedmioty z tagiem NFC stają się *komunikatywne* i biologicznie żywe: prosty gest stuknięcia pozwala smartfonom z obsługą NFC gromadzić i wymieniać informacje, parować dwa urządzenia, umożliwiając im komunikację, otwieranie drzwi lub płacenie za coś.

Niniejsza praca podzielona jest na trzy rozdziały, ukazujące NFC jako istotne ognisko w łańcuchu przyszłych mobilnych systemów informacyjnych:

– Rozdział teoretyczny omawiający najnowsze osiągnięcia NFC. Przedstawimy jego charakterystykę techniczną, a także tryby działania.

- Rozdział techniczny poświęcony programowaniu mobilnemu NFC w systemie Android, omawiający każdy aspekt implementacji NFC.
- Praktyczny rozdział z konkretnymi scenariuszami zastosowań, w którym przedstawiamy przykłady przypadków użycia oparte na prototypowaniu MBDS z 2004 r., w trzech trybach standardu NFC.

Najnowocześniejsza technologia NFC

Od 2014 roku wszyscy producenci smartfonów oferują łączność bliskiego zasięgu (NFC); standardy NFC wykorzystują właściwości elektromagnetyczne częstotliwości radiowej na bardzo krótkich dystansach, nie większych niż kilka centymetrów.

"NFC" odnosi się do kilku technologii wykorzystujących pola elektromagnetyczne umożliwiające przesyłanie danych między dwoma urządzeniami peryferyjnymi ustawionymi blisko siebie. Znana od czasów II wojny światowej identyfikacja radiowa (RFID) to bezkontaktowy system komunikacji wykorzystujący pola elektromagnetyczne do wysyłania wiadomości w celu identyfikacji i automatycznego śledzenia dzięki znacznikom połączonym z obiektami. Znaczniki zawierają dane przechowywane elektronicznie. Niektóre tagi są zasilane przez indukcję elektromagnetyczną z pól magnetycznych wytwarzanych po zbliżeniu do czytnika/kodera RFID. Tagi pasywne działają jak pasywny transponder, zasilany elektromagnetycznymi falami radiowymi wysyłanymi przez urządzenie peryferyjne inicjujące komunikację (czytnik).

Technologia NFC może oferować *połączenie ogólnego przeznaczenia* z dowolnym innym systemem komunikacji bezprzewodowej (Bluetooth®, Wi-Fi, GPRS, 4G, Li-Fi itp.) i umożliwiać parowanie urządzeń za pomocą prostego dotknięcia (paradygmat "TAP and PLAY"). Niniejszy rozdział koncentruje się na ekosystemie, w którym wdrażane są standardy NFC, jego tle i standardach.

Sztuka i nauka programowania NFC, wydanie pierwsze.

Anne-Marie Lesas i Serge Miranda.

© ISTE Ltd 2017. Opublikowane przez ISTE Ltd i John Wiley & Sons, Inc.

1.1. Przyszłe mobilne usługi cyfrowe

Mobiuity nie jest zwykłym słowem portmanteau, zasugerowanym przez Xaviera Dalloza na początku lat 90-tych, z dostępem do Internetu w telefonach komórkowych (podczas niepowodzenia WAP, szczególnie z powodu braku treści i usług). Dziś stało się pojęciem łączącym świat rzeczywisty i wirtualny, pełnym nowych treści i usług, tworząc konwergencję między MOBILNOŚCIĄ telefonu komórkowego, który stał się komputerem per se (Smartphone) i wszechobecnością Internetu, obecnie "n.0", lokalnego (*Local Wide Web*) i charakteryzującego się szeroką dystrybucją, będąc dziś w kieszeni każdego. Ta koncepcja mobilności, która odpowiada koncepcji ATAWAD ("Any Time, AnyWhere, Any Device" lub "Any Content"), jest obiecująca pod względem innowacji i multidyscyplinarna w badaniach nad treścią, usługami, architekturami i metodami.

Nowy ekosystem rozwinął się endogenicznie: konsumenci informacji (klienci docelowi) stali się dostawcami informacji - za pośrednictwem stron internetowych i aplikacji mobilnych (dzięki Internetowi) do udostępniania mediów (zdjęcia, filmy w sieciach społecznościowych), alternatywnej prasy (np. Agoravox i Alterinfo), wolnej encyklopedii (Wikipedia), otwartej rzeczywistości rozszerzonej (Wikitude) itp. - W ten sposób profesjonalisci, którzy kiedyś byli dostawcami, konsumentami i menedżerami informacji; powstają nowe modele biznesowe, generujące znaczny ładunek surowych informacji obejmujących nowe przetwarzanie predykcyjne w czasie rzeczywistym (*big data*).

Wkroczyliśmy w nową erę, w której użytkowanie wynika z indywidualnej, stowarzyszeniowej lub partycypacyjnej praktyki (sieci społecznościowe, *komunikacja*), a nie z istniejących wcześniej modeli zarządzanych przez lobbytów. Adaptacyjne podejście "oddolne" ma tendencję do przejmowania tradycyjnego podejścia "odgórnego" do rynku technologii multimedialnych oraz usług cyfrowych i telekomunikacyjnych w każdym środowisku (biznes, turystyka, transport, opieka zdrowotna, edukacja itp.).

NFC to tryb komunikacji bezdotykowej (poprzez "dotyk"), który rozszerza łączność i pozwala na pojawienie się nowych zastosowań przy zachowaniu zasady uprzedniej zgody

(zalecane przez francuską Komisję Informatyki i Wolności (*Commission Nationale de l'Informatique et des Libertés*)¹), ponieważ użytkownik jest proszony o potwierdzenie swojej woli interakcji (bez ingerencji) i musi wykonać wyraźny gest.

Ponieważ smartfon z obsługą NFC może odczytywać tagi NFC, ale także działać jako tag NFC (lub karta inteligentna NFC), informacje mogą być dystrybuowane między tagami, telefonami komórkowymi lub serwerami, co prowadzi do bardzo różnych modeli biznesowych wokół strategicznego pytania: kto będzie kontrolował informacje pochodzące z interakcji użytkownika z jego telefonem komórkowym? Operatorzy sieci komórkowych (MNO), dostawcy Internetu, banki, miasta, dostawcy usług, producenci telefonów komórkowych, menedżerowie tagów?

1.1.1. Era mobilności

W 2015 roku na świecie było tyle samo abonentów telefonii komórkowej, co osób, z 7,3 miliardami podłączonych urządzeń mobilnych, z których około połowa to smartfony² (tylko połowa tej populacji posiada konto bankowe). Prawie każda osoba na świecie ma (lub będzie miała) w kieszeni terminal komunikacyjny z bardzo szybkim przetwarzaniem danych. Oznacza to, że około 3 miliardy ludzi posiada telefon komórkowy, ale nie ma konta bankowego. Połowa naszej planety posiada smartfony, z których 64% będzie w 2018 roku wyposażonych w technologię NFC³, tworząc w ten sposób nowe możliwości usług internetowych z wartością dodaną: *bankier, przewodnik, korepetytor, lekarz, doradca w naszej kieszeni*. Perspektywa ta prowadzi do nieograniczonej kreatywności w zakresie treści, usług i praktyk w sferze prywatnej, publicznej i zawodowej, w której telefon komórkowy się krzyżyuje.

Era mobilnego Internetu dopiero się zaczyna, era Internetu między przedmiotami i osobami, *symbiont* (J. de Rosnay),

1 <http://www.cnil.fr>.

2 Ericsson Mobility, listopad 2015 (<http://www.ericsson.com/res/docs/2015/mobility-report/ericsson-mobility-report-nov-2015.pdf>).

3 Prognozy IHS Technology (<https://technology.ihs.com>).

Thłumienie czasoprzestrzeni (Michel Serres), *twórcza destrukcja* (Schumpeter) i *mobilność*, które będą bardzo produktywne pod względem innowacji i badań we wszystkich sektorach gospodarki: nauczaniu, turystyce, kulturze, handlu mobilnym, płatnościami mobilnymi (miedzynarodowy transfer środków, wirtualne pieniądze itp.).

Trzy kluczowe czynniki sukcesu nowego ekosystemu technologicznego są następujące:

- zastosowania;
- zastosowania;
- zastosowania.

Stworzono prawdziwą inżynierię usług i aplikacji mobilnych, którą ilustruje koncepcja AppStore firmy Apple lub Market/Play Store firmy Google (od 2013 r. na świecie pobiera się ponad 100 miliardów aplikacji mobilnych rocznie, według Gartner Group).⁴

Mobility opiera się na kilku następujących koncepcjach technologicznych:

– Znaczniki świata rzeczywistego odczytywane przez telefon komórkowy użytkownika końcowego; znaczniki te mogą być dwuwymiarowymi kodami kreskowymi (takimi jak standard macierzy danych i jego pochodna open source, kod QR), znacznikami częstotliwości radiowej (RFID; takimi jak znaczniki NFC), znacznikami audio, a nawet niewidocznymi znacznikami z rozpoznawaniem wzorców (takimi jak *Snap'n See* firmy Tokidev lub *Google Google*). W 2020 roku będzie istniało tysiąc miliardów oznakowanych obiektów, które będą odczytywane przez nasze telefony komórkowe, a tym samym połączone i żywe z biologicznego punktu widzenia, ponieważ życie można zdefiniować poprzez połączenie *informacji* (w tym przypadku unikalnego identyfikatora obiektu) i *komunikacji* (za pośrednictwem telefonu komórkowego, który ma dostęp do historii oznakowanego obiektu w bazie danych).

– *Zmieniona rzeczywistość (rozszerzona lub pomniejszona)* z platformami *open source*, takimi jak Wikitude i Layar, umożliwiająca wprowadzanie informacji z bazy danych (na przykład informacji o nieruchomościach lub informacji turystycznych) na wierzchu rzeczywistego świata oglądanego z telefonu komórkowego.

4 Źródło: Mobile World Congress (<http://www.mobileworldcongress.com>).

lub wręcz przeciwnie, do usuwania/zastępowania elementów świata rzeczywistego (np. w celu wyobrażenia sobie nowego wystroju wnętrz, nowej perspektywy urbanistycznej itp.)

– *Transmedia*: Koncepcja stworzona przez Marthę Fisher w 1991 roku i kontynuowana w *Convergence Culture*, książce profesora MIT Henry'ego Jenkinsa z 2003 roku, z treściami dostosowanymi do *pięciu ekranów w naszej historii* (kino, telewizja, komputer, telefon komórkowy i tablet); m u s i m y zauważyc, że w tej historii komunikacji wiodące firmy oferujące treści i usługi dla jednego ekranu nigdy nie były liderami dla następnego ekranu. Następne ekranы będą na ścianach (okna, lustra, ubrania, skóra itp.).

– *Big data*: Według IBM⁵, każdego dnia generowanych jest 2,5 biliona⁶ bajtów danych, a 90% globalnych danych powstało w ciągu ostatnich 2 lat. Dane te pochodzą z czujników, wiadomości publikowanych w sieciach społecznościowych, zdjęć i filmów publikowanych w Internecie, z geolokalizacji (na przykład GPS lub NFC) z urządzeń mobilnych (smartfonów, tabletów, inteligentnych zegarków), dzienników śledzenia płatności online (na przykład z płatności elektronicznych lub płatności mobilnych), słów kluczowych wpisywanych w wyszukiwarkach i, bardziej ogólnie, wszelkich informacji cyfrowych, które można wykorzystać do interpretacji (dane naukowe, wykrywanie oszustw, dane medyczne, dane osobowe, dane behawioralne itp.) Big data można zdefiniować jako "3V": objętość ("tsunami danych": 20 petabajtów⁷ danych jest przetwarzanych każdego dnia przez Google, a dzieje się tak od 2010 r., 1,8⁸ zettabajtów zostało wyprodukowanych w 2011 r., co oznacza wzrost o 50% rocznie), różnorodność (*dane internetowe* z sieci społecznościowych, *powiązane dane* z sieci semantycznej, dane publiczne, *otwarte dane* i dane mobilne pochodzące z oznaczonych obiektów i sieci czujników itp.) i prędkość (przepływ danych z czujników i sieci społecznościowych, w czasie rzeczywistym "w locie"), do których dodajemy czwartą "V": zmienność (ewolucja) i piątą "V": waloryzacja (poprawa i wzbogacenie).

5 IBM jest jednym z dwóch światowych liderów (wraz z Oracle) specjalizujących się w systemach informatycznych i zarządzaniu bazami danych (www.ibm.com).

6 1 bilion = 1 000 miliardów, co oznacza liczbę 10 jednostek.

7 1 petabajt (PB) = 250 bajtów = 1 024 terabajtów (TB) = 1 125 899 906 842 624 bajtów.

8 1 zettabajt (ZB) = 270 bajtów = 1,024 eksabajtów (EB) = 1,024 PB = 1,180,591,620,717 411 303 424 bajtów.



Rysunek 1.1. Wywiad z Pr. S. Miranda dla Docapost (2011)

1.1.2. W kierunku świata przedmiotów komunikujących się bezdotykowych

Z NFC standardy dowolny może stać się "komunikatywny", obiekt (a tym samym żywe). Niezależnie od tego, czy są one dostępne na rynku, czy też dopiero się rozwijają, istnieje już szeroka gama urządzeń obsługujących NFC i urządzeń do noszenia: inteligentne zegarki, bransoletki, okulary, ubrania lub bezpośrednio na naszym ciele. Inteligentne urządzenia coraz częściej stają się częścią naszego codziennego życia; parkometry, skrzynki pocztowe z NFC, lodówki analizujące nawyki użytkowników i przekazujące informacje do smartfona (zużycie energii, temperatura, drzwi), otwarcie itp.); połączone możliwość sparowania z urządzeniem obsługującym NFC piekarnik a smartfon sugerujący użytkownikowi przepisy kulinarne i automatycznie skonfigurować funkcję gotowania. NFC i jego trzy tryby działania (patrz sekcja 1.3.2) w czytelnik/pisarz, emulacja karty i znaczniku (inteligentny) peer-to-peer (P2P) stanowią źródło innowacji dla obiektów, które stały się "inteligentny", ponieważ smartfon łącznością.

"Obiekty" z obsługą NFC można podzielić na dwie kategorie:

- Przedmioty z obsługą NFC (z zasilaczem) zazwyczaj "Aktywny"
- posiadają czytnik NFC: z interfejsem graficznym lub bez, wchodzą w interakcję po zbliżeniu do innego urządzenia peryferyjnego lub urządzenia

Znacznik NFC. W większości przypadków Najnowocześniejsza
aktywnie urządzenie
technologia NFC⁷
peryferyjne jest hostem

aplikacja "terminal", co oznacza, że jego łączność pozwala komunikacja ze zdalnym serwerem sparowan z bazą danych w

y
w celu pobrania informacji z tego ostatniego. W ten sposób użytkownik końcowy jest w stanie wyzwać interaktywne i kontekstowe zdarzenia, niezależnie od tego, czy są to

przetwarzane lokalnie lub zdalnie w celu uzyskania wyniku w świecie cyfrowym (np. wyświetlanie informacji lub bezpieczna płatność) i/lub w świecie rzeczywistym (np. otwarcie drzwi).

- "Pasywne" obiekty NFC są oparte na tagu NFC bez zasilania; aktywują się po zbliżeniu do "aktywnego" obiektu z obsługą NFC; mogą wtedy wykorzystać moc pola magnetycznego indukowaną przez aktywne urządzenie w trybie "czytnika", który następnie inicjuje komunikację.

1.2. Sprzęt NFC

Podstawowe urządzenie NFC składa się co najmniej z anteny połączonej z modulatorem-demodulatorem, który konwertuje sygnały elektromagnetyczne na dane cyfrowe (i odwrotnie); w tagu NFC chip ma bardzo małą pojemność pamięci (~1 kb), podczas gdy w przypadku bardziej złożonego urządzenia peryferyjnego kontroler NFC umożliwia interfejs chipu NFC z systemem operacyjnym urządzenia hosta.

1.2.1. Znacznik NFC

Tagi NFC mogą być dowolnego rodzaju i formy: naklejki, breloki, bransoletki itp. Oferują te same funkcje co kody QR, ale nie wymagają więcej niż prostego "dotknięcia" bez interwencji użytkownika, w przeciwieństwie do kodów QR, które wymagają od użytkownika uruchomienia odpowiedniej aplikacji, aby je odczytać. Tagi NFC są łącznikiem między światem rzeczywistym a wirtualnym; mogą być również używane do aktywacji lub dezaktywacji funkcji (alarmy, urządzenia, przetwarzanie cyfrowe) i wszelkich innych zdarzeń wyzwalanych przez proste "dotknięcie" za pomocą smartfona.



Rysunek 1.2. Naklejka NFC

Rysunek 1.2 przedstawia (pasywny) tag NFC, tutaj przezroczystą naklejkę, która należy do rodziny NTAG produkowanej przez NXP.

Identyfikacja jest cechą charakterystyczną tagów NFC z niemodyfikowalnym uniwersalnym unikalnym identyfikatorem (UID) przypisanym do nich w momencie produkcji. W ten sposób, bez dodatkowych danych, podłączony obiekt może być po prostu jednoznacznie zidentyfikowany ze względu na UID tagu NFC, z którym jest połączony, niezależnie od tego, czy tag jest przyklejony, czy osadzony w tym obiekcie. Aplikacja "terminalowa" podłączona do czytnika NFC lub uruchomiona w urządzeniu mobilnym z obsługą NFC (w trybie czytnika) może zatem odczytać identyfikator UID obiektu w zależności od przypadku użycia aplikacji:

- wyświetlanie informacji o obiekcie (np. identyfikatory/paszporty elektroniczne, informacje turystyczne, przewodniki użytkownika);
- wykonać zadanie (np. dokonać rezerwacji);
- uruchomić zdarzenie w świecie wirtualnym lub nawet rzeczywistym (otwarcie drzwi, aktywacja/dezaktywacja alarmu itp.);
- zapisać zdarzenie, dodając do niego dostępne dane kontekstowe (np. identyfikator UID obiektu, znacznik czasu i daty, lokalizację geograficzną, użytkownika) w celu śledzenia lub wskazywania.

1.2.2. Karta inteligentna NFC

Karta inteligentna jest bardziej zaawansowanym rodzajem tagu z czasami hybrydowym interfejsem komunikacyjnym (*podwójny interfejs*) zapewniającym zarówno tradycyjny interfejs kontaktowy, jak i interfejs NFC umożliwiający komunikację z usługami wbudowanymi w kartę inteligentną.

Bezstykowa karta chipowa NFC jest używana jako karta płatnicza (w 2014 r. na świecie sprzedano 600 milionów kart płatniczych NFC⁹), jako bilet transportowy, a także jako środek identyfikacji i uwierzytelniania posiadacza karty, na przykład do kontroli dostępu. Karta

⁹ Źródło: Mobile Payments Today (<http://www.mobilepaymentstoday.com/news/contactless-payments-us-emv-migration-power-sharp-increase-in-smart-card-growth>).

Karta SIM jest najbardziej rozpowszechnionym typem (podobnym do karty inteligentnej, ale mniejszym): istnieje tyle aktywnych kart SIM, ilu jest mieszkańców na świecie, a wynika to z faktu, że niektórzy użytkownicy mają więcej niż jedną kartę SIM.

Wbudowana w telefon komórkowy z obsługą NFC (smartfon, tablet) karta inteligentna działa jako *bezpieczny element* (SE) w celu bezpiecznego hostowania usług i poufnych danych. Ta konfiguracja pokazuje zalety rozszerzenia usług opartych na kartach inteligentnych na zaawansowane technologie smartfonów (graficzny interfejs dotykowy użytkownika, dźwięk, kamera, wbudowane czujniki, GPS itp.).

1.2.2.1. Karty inteligentne i bezpieczeństwo

Wymuszone bezpieczeństwo to szeroka dziedzina obejmująca wiele aspektów, takich jak ochrona prywatności, kontrola dostępu, ochrona przed wirusami i włamaniami, integralność i niezaprzecjalność danych w świecie cyfrowym.

Uwierzytelnianie służy do weryfikacji tożsamości użytkownika. Może być poświadczane za pomocą różnych, ewentualnie połączonych czynników (uwierzytelnianie wieloskładnikowe). Na przykład, mogą być one oparte na:

- osoby (informacje biometryczne);
- wspólny sekret (hasło, PIN, klucz szyfrowania);
- obiekt (karta inteligentna, smartfon itp.).

Zezwolenie/autoryzacja polega na sprawdzeniu, czy (uwierzytelniony) podmiot, który chce uzyskać dostęp do zasobu, ma na to pozwolenie. Zarządzanie uprawnieniami odbywa się na poziomie indywidualnym lub przez grupę podmiotów (uprawnienia lub ograniczenia mogą być definiowane zgodnie z rolami).

Poufność to gwarancja, że ani dane uwierzytelniające, ani współdzielone zasoby nie mogą zostać przechwycone. Szyfrowanie danych (kryptografia) jest najbardziej rozpowszechnioną techniką zachowania poufności.

Integralność danych polega na upewnieniu się, że zasoby nie ulegają żadnym zmianom podczas przechowywania danych ani podczas dostępu do nich, gdzie muszą być w pełni odzyskane, z taką samą precyzją, ale przede wszystkim poprzez poświadczenie ich autentyczności i ważności.

Standard infrastruktury klucza publicznego (PKI) to metoda zarządzania kluczami kryptograficznymi oparta na zasadzie klucz prywatny/klucz publiczny:

- Klucz prywatny jest generowany w celu szyfrowania wiadomości. Klucze prywatne pozostają tajne i nigdy nie są rozpowszechniane;
- Jeden lub kilka kluczy publicznych jest generowanych i dystrybuowanych do odbiorców zaszyfrowanych wiadomości i są one używane do odszyfrowania wiadomości (i zaszyfrowania odpowiedzi).

W ten sposób system oparty na PKI zabezpiecza pochodzenie wiadomości (tylko zaszyfrowane wiadomości z kluczem prywatnym mogą być odszyfrowane za pomocą klucza publicznego) i poufność (tylko właściciel klucza publicznego może odszyfrować zaszyfrowaną wiadomość za pomocą klucza prywatnego).

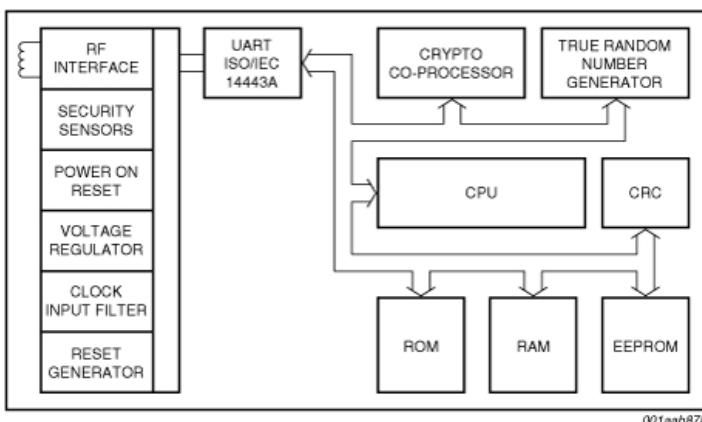
Główne cechy PKI opierają się na bezpieczeństwie oprogramowania i sprzętu:

- Klucz prywatny jest przechowywany w bezpiecznym chipie elektronicznym (SE) i nigdy go nie opuszcza;
- klucz publiczny jest eksportowany;
- Mechanizm szyfrowania jest zakodowany na stałe w chipie;
- Mechanizm uwierzytelniania jest również wbudowany w odporną na manipulacje technologię mikrokontrolera karty intelligentnej.

W ten sposób karty intelligentne idealnie nadają się do celów identyfikacji i uwierzytelniania; między innymi przechowują zaszyfrowany certyfikat cyfrowy w celu uwierzytelnienia usługi. Najczęściej stosowanymi algorytmami kryptograficznymi są potrójny standard szyfrowania danych (3DES) z kluczami prywatnymi i współdzielonymi (w czytniku symetrycznym).

(w trybie asymetrycznym) oraz standard RSA¹⁰ z dystrybucją klucza publicznego (w trybie asymetrycznym). Klucze można załadować lub wygenerować na karcie inteligentnej na etapie dostosowywania.

Europejskie karty MasterCard Visa (EMV), karty debetowo-kredytowe oparte na chipach, a także terminale płatnicze i bankomaty umożliwiające korzystanie z kart bankowych są szeroko stosowane w celu uwierzytelnienia posiadacza karty. Jest to jeden z powodów, dla których karty inteligentne są głównym celem ataków bezpieczeństwa. Ataki na karty inteligentne obejmują zarówno fizyczną ingerencję w elektronikę (za pomocą sondy jonowej, mikroskopu, ataku chemicznego lub lasera itp.), która prowadzi do zniszczenia karty inteligentnej, jak i ataki półinwazyjne (za pomocą oscyloskopu) i nieinwazyjne (programowo), które wykorzystują słabości sprzętu i oprogramowania karty. Producenci kart inteligentnych i standary bezpieczeństwa rozwijają się równolegle z atakami w celu włączenia środków zaradczych gwarantujących maksymalne bezpieczeństwo i odporność podczas cyklu życia karty; dlatego ważne jest, aby wdrażać tylko najnowsze modele kart inteligentnych, których technologia obejmuje pełne środki zaradcze na znane ataki.



Rysunek 1.3. Architektura bezstykowej karty inteligentnej (źródło NXP)¹¹

10 Algorytm opracowany przez Rivesta, Shamira i Adlemana.

11 <http://www.nxp.com/scale-image/w-800/documents/blockdiagram/001aah878.gif>.

Najbardziej wyrafinowane karty inteligentne są tworzone z materiałów, które są szczególnie dedykowane do kryptografii i algorytmów szyfrowania, takich jak RSA i algorytmy podpisu cyfrowego klucza publicznego oparte na kryptografii krzywej eliptycznej, stosowane w operacjach asymetrycznych do wymiany kluczy w niezabezpieczonym kanale lub do szyfrowania asymetrycznego. Pary kluczy są generowane wewnątrz karty intelligentnej w celu wyeliminowania ryzyka ujawnienia klucza prywatnego, który nie jest dystrybuowany, a zatem pozostaje nieznany.

1.2.2.2. Uwierzytelnianie wieloskładnikowe

Metoda egzekwowania cyfrowej kontroli dostępu, która wymaga więcej niż jednego typowego czynnika uwierzytelniania. Mogą to być dane logowania (login, hasło) i jednorazowy kod generowany losowo przez system, którego ważność jest ograniczona w czasie. Kod jednorazowy jest wysyłany w czasie rzeczywistym do użytkownika za pośrednictwem innego kanału komunikacji, na przykład wysyłanie kodu autoryzacyjnego przez SMS jest szeroko rozpowszechnionym przykładem uwierzytelniania dwuskładnikowego (2FA) stosowanego w bezpiecznych transakcjach płatności online: użytkownik uwierzytelnia się w interfejsie banku w Internecie za pomocą swojego loginu i hasła i zatwierdza zakup, a następnie system bankowy wysyła kod weryfikacyjny przez SMS na telefon komórkowy użytkownika, a użytkownik musi wprowadzić ten otrzymany kod weryfikacyjny w interfejsie internetowym, aby zakończyć transakcję. Płatność jest zatwierdzana dopiero po sprawdzeniu 2FA.

Zazwyczaj uwierzytelnianie wieloskładnikowe opiera się na dwóch uzupełniających się zasadach:

- coś, co użytkownik "zna", na przykład nazwa i hasło lub kod PIN;
- coś, co użytkownik "ma" (i system o tym wie), na przykład token, chip lub karta inteligentna (zawierająca na przykład klucz), telefon komórkowy, klucz sprzętowy itp.

1.2.2.3. Bezpieczne kanały komunikacji

Izolowany kanał komunikacyjny umożliwia ustanowienie prywatnego kanału sieciowego między dwoma programami (np. VPN/specyficzny punkt-do-punktu).

połączenie punkt-gniazdo). Ustanowienie bezpiecznego kanału komunikacji wymaga wcześniejszego uwierzytelnienia i identyfikacji; protokół musi zapewniać prywatność i integralność (szyfrowanie/deszyfrowanie, stan). Izolowany kanał komunikacyjny może być bezpieczny lub nie, a bezpieczna komunikacja może być izolowana lub nie.

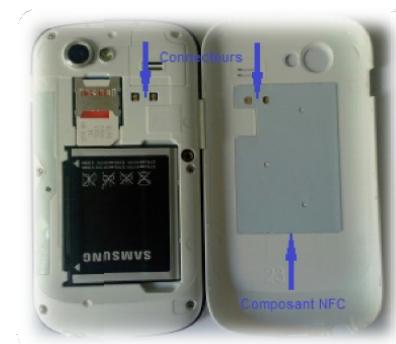
1.2.3. Smartfon z NFC

Smartfony NFC odpowiadają na pięć wymiarów w mobilnym systemie informacyjnym (pięć "W": *kto, gdzie, kiedy, gdzie i co*): tożsamość posiadacza (z jego nawykami, preferencjami), przestrzeń i czas ("tu i teraz", kiedy stuka), cel do osiągnięcia poprzez stuknięcie (informacja, transakcja?) i uzyskany wynik (informacja, kupon, transakcja, spotkanie itp.).

Pierwszy telefon komórkowy z obsługą NFC został wprowadzony na rynek w 2006 roku przez firmę Nokia: Nokia 6131 NFC była dostarczana z wbudowanym SE (lub eSE), do domyślnego korzystania z usług (np. MasterCard, Visa©, SNCF) dostępnych na platformie usługowej.

Obecnie wszyscy główni producenci smartfonów proponują urządzenia mobilne z obsługą NFC: Google po raz pierwszy wprowadził na rynek pierwszy smartfon NFC w 2010 roku (Nexus S) z główną strategią płatności mobilnych uruchomioną w maju 2011 roku w Nowym Jorku. Od tego czasu Samsung, Apple, a następnie LG poszły w ślady Google, oferując własne platformy płatności mobilnych, konkuruje z uniwersalną kartą SIM jako SE kontrolowaną przez MNO.

Z lub bez SE, smartfon NFC może działać jako tag NFC (lub bezstykowa karta inteligentna), ale może również działać jako czytnik NFC w celu odczytu (lub zapisu) zawartości tagów, a nawet działać jako terminal płatniczy, który może również zrewolucjonizować bezpieczne transakcje; do tej pory ta dziedzina była przeznaczona wyłącznie dla zastrzeżonych systemów, będących własnością i kontrolowanych przez profesjonalistów.



Rysunek 1.4. Antena Nexus S NFC wbudowana w obudowę

1.2.4. Czytnik/koder: Terminale transakcyjne NFC

Czytnik NFC może odczytywać i zapisywać zawartość tagów w zależności od formatów określonych przez standard NFC (patrz sekcja 1.3). Z lub

bez klawiatury i ekranu, czytniki NFC są dostępne przez program umożliwiający odczyt/zapis lub komunikację z innym urządzeniem NFC (na przykład tagiem lub smartfonem) w zależności od zgodności czytnika ze standardami.



Rysunek 1.5. Czytnik ASC NFC ACR122U

1.2.5. "Inteligentne miasta" i zrównoważony rozwój

Niezależnie od tego, czy jest to podczerwień, Wi-Fi, Bluetooth® czy NFC, my żyjemy w czasach "maszyny do maszyny" i połączony przedmiot y:

połączona infrastruktura wyposażona w czujniki (zanieczyszczenia, wilgotności, światła, temperatury, ruchu, korków ulicznych itp.), automatyczne wyzwalanie (alarmy, sygnały, oświetlenie miejskie, podlewanie ogrodów publicznych itp.

NFC znajduje praktyczne zastosowanie w życiu eko-obywatela, na przykład w multimodalności, która ma na celu oferowanie wszystkich możliwych rozwiązań transportowych, możliwych połączeń z punktu A do punktu B: aplikacje mobilne pozwalają nam lokalizować, rezerwować i płacić za ekologiczne środki transportu (*vélibs/blue bikes*, pojazdy elektryczne, car sharing lub transport publiczny), jednocześnie obliczając ich dostępność i na podstawie czasu podróży, rozkładów jazdy itp.

NFC to uniwersalne złącze, dzięki któremu każda przestrzeń może stać się inteligentnym miejscem: przechowywanie danych medycznych w czasie rzeczywistym na platformie opieki zdrowotnej, która powie nam, kiedy udać się do lekarza, zbieranie statystyk dotyczących zużycia energii bezpośrednio na naszym telefonie komórkowym, korzystanie ze wskazówek dotyczących użytkowania i możliwość zdalnej interakcji, unikanie marnowania czasu przy kasie, aby za coś zapłacić.

Wszystkie te koncepcje są w modzie, istnieją już programy pilotażowe, a ich pojawienie się idzie w parze z nastawieniem na zrównoważony rozwój. W tym kontekście, ze względu na niskie zużycie energii i bezpieczne podejście do komunikacji, NFC będzie odgrywać kluczową rolę.

1.2.6. Płatności bezgotówkowe z NFC

Handel elektroniczny narodził się wraz z pojawieniem się Internetu i handlu online; ten nowy wzorzec konsumpcji rozkwitł wraz z pojawieniem się smartfonów i tabletów, otwierając drogę do m-handlu (geolokalizowanego lub nie), a płatności mobilne stały się nawykiem. Dziś dzięki technologii możemy wybrać produkt, gdziekolwiek jesteśmy (w domu, w biurze lub poza nim, sami lub z innymi), kiedykolwiek, czy to w dzień, czy w nocy.

Płatności mobilne otworzyły drogę do dematerializacji tradycyjnych środków płatniczych (gotówka, czeki, karty bankowe): już przetestowane na całym świecie

wykorzystanie standardów NFC wydaje się być najbardziej odpowiednią technologią do konkurowania z kartami bankowymi (jako logiczne uzupełnienie transakcji blockchain i kryptowalut).

Stawka jest wysoka, ponieważ NFC może zakłócić ugruntowany ekosystem i zainicjować przejście od monopolu bankowego do nowych podmiotów (MNO, Google, Apple, Samsung) oraz otwarcie na osoby prywatne z zarządzaniem bankowością online (transfer pieniędzy z prywatnych kont bankowych na konta osób trzecich), a następnie z naszych telefonów komórkowych (podejście "oddolne", w którym użytkownik końcowy przejmuje kontrolę w porównaniu z podejściem "odgórny", w którym dostawca usług bankowych zarządza i kontroluje wszystko).

Koncepcja wirtualnej waluty może mieć również zastosowanie do populacji nieubankowionych: według badania Banku Światowego z 2012 r. przeprowadzonego na temat włączenia finansowego, 2,5 miliarda dorosłych na świecie (prawie połowa światowej populacji) nie ma konta bankowego (wśród nich 70% ma telefon komórkowy). Jest to ogromne pole do potencjalnej dynamizacji, z którego wyłoniły się mobilne projekty dotyczące finansowania społecznościovego lub mikrofinansowania, których waluta wymiany niekoniecznie jest finansowa (na przykład wymiana zwierząt gospodarskich i handel nasionami).

1.3. Standardy NFC

Poza standardami, nie ma ratunku dla deweloperów!

Standaryzacja zaspokaja powszechną potrzebę w ekosystemie obejmującym wiele podmiotów w celu ułatwienia interoperacyjności, trwałego rozwoju aplikacji i promowania wdrażania nowych technologii i ich zastosowań.

Standardy NFC są nieodłącznie związane z telekomunikacją i smartfonami, z jednej strony ze względu na fakt, że ich przypadki użycia są idealnie dostosowane do użytku mobilnego, a z drugiej strony z powodu najczęściej używanej karty inteligentnej, globalnie wdrażanej we wszystkich telefonach komórkowych: karty SIM.

W 2002 r. firmy¹² , Philips (w 2006 r. pod nazwą NXP) i Sony opracowały technologię i standard NFC w oparciu o swoje know-how w zakresie chipów bezprzewodowych (FeliCa dla Sony i MIFARE). W tym czasie Nokia dołączyła do Philipsa i Sony, tworząc w 2004 roku NFC Forum (www.nfc forum.org).

NFC jest jednym z 16 standardów RFID ISO, sklasyfikowanych według zakresu częstotliwości: niska częstotliwość (<135 kHz), wysoka częstotliwość (w tym NFC przy 13,56 MHz), ultra wysoka częstotliwość (433 MHz) i mikrofale (2,45 i 5,8 GHz).

Standardy NFC mogą być wykorzystywane zarówno w kartach inteligentnych, jak i urządzeniach mobilnych. Spotkanie najpopularniejszego urządzenia sprzętowego (telefonu komórkowego) z najbardziej rozpowszechnionymi platformami komunikacyjnymi (Internetem) i najnowszym bezprzewodowym standardem krótkiego zasięgu (NFC) dało początek nowemu paradygmatowi usług w transporcie publicznym, płatnościach mobilnych, marketingu, geolokalizacji, wyszukiwaniu informacji kontekstowych, sieciach społecznościowych, nauczaniu, dostarczaniu i udostępnianiu treści cyfrowych, przy jednoczesnym ponownym przemyśleniu interakcji w świecie fizycznym. Przypadki użycia aplikacji i usług zbliżeniowych z NFC są nieograniczone.

W ten sposób NFC jest pół- lub pełnoduplexową bezprzewodową technologią komunikacyjną krótkiego zasięgu (<10 cm) opartą na RFID i nielicencjonowanych usługach.

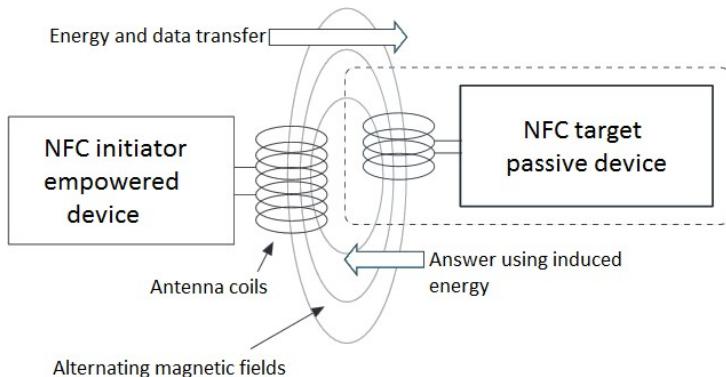
Częstotliwość radiowa 13,56 MHz. Standardy NFC są uznawane przez ISO/IEC i ETSI i obejmują kilka norm, takich jak ISO/IEC 18092, ISO/IEC 14443 i ECMA (ECMA-340).

NFC umożliwia dwa tryby komunikacji oparte na sprzężeniu indukcyjnym RFID:

– *Tryb komunikacji pasywnej NFC*: w tym trybie tylko inicjator komunikacji NFC jest uprawniony; urządzenie docelowe komunikacji odpowiada na modulację obciążenia za pomocą sprzężenia indukcyjnego w pobliżu urządzenia inicjującego, jak pokazano na rysunku 1.6.

12 Źródło: TechPats (<http://www.techpats.com/evolution-near-field-communication-nfc>).

– *Aktywny tryb komunikacji NFC*: w tym trybie komunikacji oba urządzenia (inicjator i urządzenie docelowe) generują własne fale radiowe w celu przesyłania danych.



Rysunek 1.6. Pasywny tryb komunikacji NFC

1.3.1. Sygnał analogowy i transpozycja cyfrowa NFC

Komunikacja bezstykowa polega na modulowaniu i demodulowaniu danych binarnych (bitów o wartości 0 lub 1) w sygnale (sygnale analogowym) poprzez wysyłanie fali (elektromagnetycznej) zwanej "falą nośną": poprzez tworzenie zmian amplitudy, fazy i częstotliwości, sygnał jest transponowany cyfrowo po odebraniu.

W początkowej specyfikacji standard NFC obejmuje trzy rodzaje kodowania:

– NFC-A wykorzystuje kodowanie Miller (nadawca) i Manchester (odbiorca) z modulacją amplitudy przy 100% przesunięciu amplitudy (ASK) (zero sygnału podczas przerw) przy 106 kbps¹³ ;

– NFC-B wykorzystuje kodowanie non-return-to-zero (NRZ) z modulacją amplitudy na poziomie 10% ASK (nadal brak sygnału podczas przerw) przy wysyłaniu i modulację poprzez binarne przesunięcie fazowe (BPSK) przy odbiorze z prędkością 106 kbps;

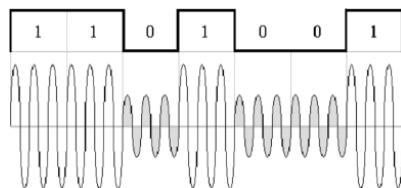
13 Szybkość zegara w kilobajtach na sekundę (kb/s).

– NFC-F odpowiada technologii FeliCa, która jest powszechnie rozwijana w Japonii i wykorzystuje kodowanie Manchester z modulacją amplitudy ASK przy prędkości 212 lub 424 kb/s.

UWAGA: - Forum NFC bada nową specyfikację, dzięki której standard (Tag Type 5) byłby zgodny z sygnalizacją kart zbliżeniowych (standard ISO/IEC 15693) w krótkim zasięgu wymaganym przez NFC.

1.3.1.1. *Modulacja ASK*

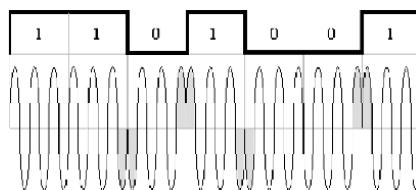
Przejście (zmiana z 1 na 0 bitów) następuje z powodu zmiany amplitudy sygnału: amplituda jest obniżona lub zerowa.



Rysunek 1.7. *Modulacja ASK*

1.3.1.2. *Modulacja BPSK*

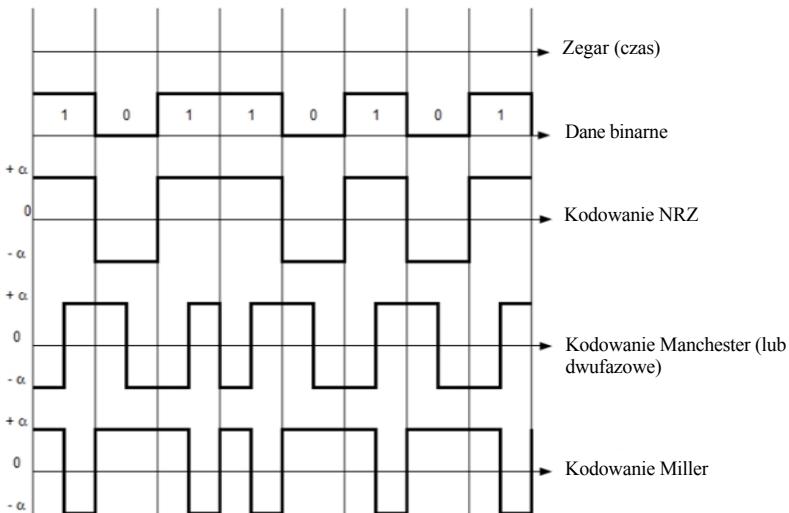
Przy każdym przejściu amplituda sygnału jest odwracana, tworząc skok fazy.



Rysunek 1.8. *Modulacja BPSK*

1.3.1.3. *Demodulacja*

Demodulacja to proces ekstrakcji danych, który polega na konwersji źródła analogowego na kodowanie binarne.



Rysunek 1.9. Kodowanie cyfrowej modulacji sygnałów analogowych

1.3.1.4. Kodowanie NRZ

NRZ jest najprostszym kodowaniem. Dodatnie napięcie reprezentuje bit danych o wartości 1, podczas gdy ujemne napięcie reprezentuje bit o wartości 0.

1.3.1.5. Kodowanie Manchester (lub faza binarna)

Jest to kodowanie przejścia (nie kodowanie poziomu): przejście zbocza narastającego reprezentuje bit o wartości 0. Przejście zbocza opadającego odpowiada bitowi o wartości 1.

1.3.1.6. Kodowanie Miller

Schemat kodowania Millera opiera się na schemacie kodowania Manchester, z którego każde inne przejście jest odrzucane: dla bitu o wartości 1 przejście jest umieszczane w połowie przedziału bitowego, chyba że następuje po nim bit o wartości 0, w którym to przypadku przejście jest umieszczane na końcu przedziału bitowego.

1.3.2. Trzy standardowe tryby NFC

Urządzenia obsługujące technologię NFC z o s t a ł y zilustrowane na rysunku 1.10:

- czytnik/zapis (tj. czytnik/zapis tagów);
- P2P;
- emulacja karty (tj. chip).



Rysunek 1.10. Trzy tryby pracy urządzeń NFC¹⁴

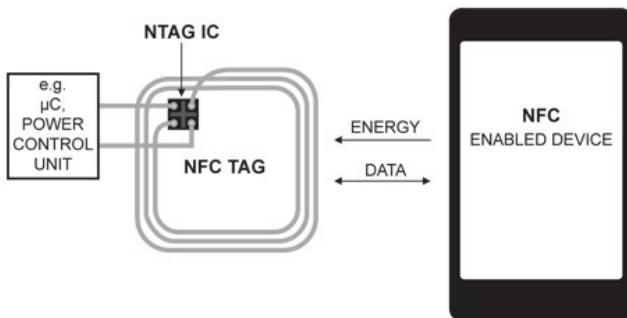
UWAGA: - Te trzy uzupełniające się tryby działania standardu NFC wykluczają się wzajemnie.

Tryby NFC są oparte na standardach ISO/IEC 18092 NFC IP-1, JIS X 6319-4 i ISO/IEC 14443 bezstykowych kart inteligentnych (określanych jako NFC-A, NFC-B i NFC-F w specyfikacjach forum NFC).

1.3.2.1. Tryb odczytu/zapisu NFC

Jak pokazano na rysunku 1.11, tryb odczytu/zapisu NFC umożliwia urządzeniom NFC odczytywanie i zapisywanie informacji przechowywanych na pasywnych urządzeniach NFC, składające się z anteny i układu scalonego. Zasilanie jest dostarczany przez sprężenie indukcyjne z urządzenia NFC inicjującego komunikację, gdy znajdzie się ono w pobliżu pasywnego tagu NFC.

14 Źródło: Forum NFC (<http://nfc-forum.org/what-is-nfc/what-it-does>).



Rysunek 1.11. Tryb odczytu/zapisu NFC¹⁵

Informacje przechowywane na tagu NFC powinny być zgodne ze standardem definicji typu rekordu (RTD) określonym przez forum NFC lub być formatem zastrzeżonym (patrz sekcja 1.3.3.3.2).

Różne typy tagów NFC są opisane przez standard (patrz sekcja 1.3.3.3.1): w tagu NFC dane są przechowywane w komunikatach formatu wymiany danych NFC (NDEF) (patrz sekcja 1.3.3.2), które obejmują typ, format typu, identyfikator i tablicę bajtów. Na przykład możliwe jest zapisanie linków do pobrania/URL w tagu NFC, który może przechowywać kilka rekordów. Samo odniesienie może również umożliwić urządzeniu pobieranie powiązanych danych ze zdalnego serwera, na przykład za pomocą dedykowanej usługi internetowej.

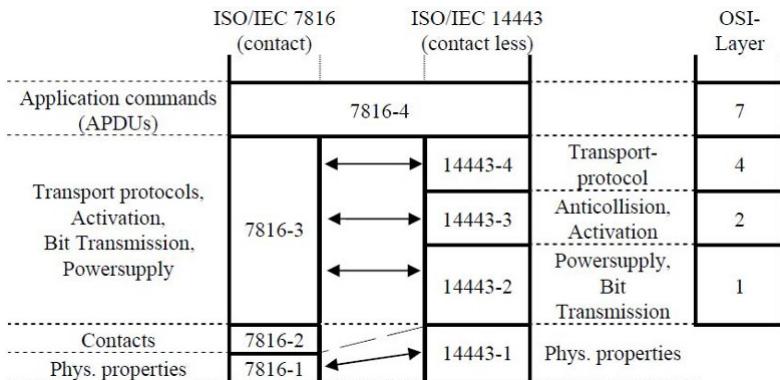
Urządzenie obsługujące technologię NFC może, po uprzednim zaakceptowaniu przez użytkownika końcowego lub automatycznie, uruchomić odpowiednią aplikację w celu uzyskania dostępu do połączonej treści (na przykład tekst, obrazu multimedialnego/audio/video, strony internetowej) lub wywołać działanie elektryczne (na przykład włączenie światła), mechaniczne lub zapachowe.

1.3.2.2. Tryb emulacji karty NFC z SE

Tryb emulacji karty NFC umożliwia urządzeniom NFC działanie jak karty inteligentne, pozwalając użytkownikom na wykonywanie transakcji, takich jak płatności mobilne,

15 Źródło: NFC World (<http://www.nfcworld.com/2013/08/14/325492/nxp-begins-shipping-nfc-tags-that-can-wake-up-a-host-device>).

biletów i kontroli dostępu do tranzytu za pomocą jednego dotknięcia. W tym trybie warstwa bezstykowa i pola RF działają jako transport dla standardu kart inteligentnych (ISO/IEC 7816), jak pokazano na rysunku 1.12.



Rysunek 1.12. Protokoły trybu emulacji karty NFC

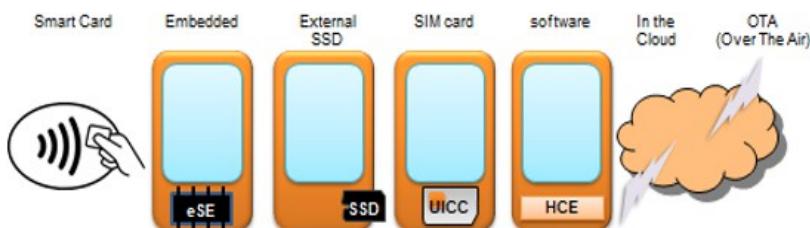
W trybie emulacji karty urządzenia NFC komunikują się z zewnętrznym czytnikiem zbliżeniowym: tryb emulacji karty NFC umożliwia obsługę już istniejących protokołów na stosach NFC bez zmiany infrastruktury (patrz rysunek 1.12). W przypadku aplikacji płatniczych NFC ten tryb emulacji karty NFC jest oparty na standardzie EMV i specyfikacjach kart PIN.

Tryb emulacji karty NFC obejmuje odporny na manipulacje komponent sprzętowy zwany "bezpiecznym elementem". SE to wielousługowy układ scalony z własnym mikroprocesorem (CPU) i procesorem kryptograficznym, własnym systemem operacyjnym (JavaCardTM), a także pamięcią lotną i nielotną, w tym programowalną pamięcią tylko do odczytu (EPROM) o pojemności do 6 Gb, interfejsem wejścia/wyjścia do odbierania wiadomości i wysyłania odpowiedzi oraz jednym lub kilkoma (na przykład hybrydową kartą stykową/RFID) interfejsami zasilania. Może być samodzielnią (stykową lub bezstykową) kartą inteligentną lub komponentem karty hybrydowej.

urządzenie hosta (np. smartfon lub tablet) z trzema możliwymi konfiguracjami SE, jak pokazano na rysunku 1.13:

- na karcie SIM lub uniwersalnej karcie zintegrowanej, obsługiwanej przez MNO;
- wbudowany w urządzenie (eSE) i obsługiwany przez producenta urządzenia (na przykład Apple, Samsung, LG);
- zewnętrzne/wymienne bezpieczne karty pamięci, takie jak micro Secure Digital (microSD).

SE może być dostępny zdalnie w chmurze i/lub emulowany przez usługę działającą w tle, która zachowuje się jak (sprzętowy) SE. Ten tryb, który nie jest oparty na sprzęcie, nazywany jest trybem *emulacji karty opartej na hoście* (HCE).



Rysunek 1.13. Kilka konfiguracji SE

SE jest głównie przeznaczony do bezpiecznego przechowywania poufnych danych (np. danych konta, identyfikacji, danych uwierzytelniających użytkownika, kluczy szyfrowania/deszyfrowania). Procesy działające w SE (poza głównym systemem operacyjnym urządzenia hosta) są powszechnie nazywane *apletami* (dziedziczenie po JavaCardTM), *kardletami* lub *serwletami*; aplikacja, która pozwala użytkownikowi zarządzać zestawem wirtualnych kart i usług zdigitalizowanych w SE, nazywana jest *portfelem*.

Standardy związane z SE są scentralizowane przez GlobalPlatform (www.globalplatform.org), międzynarodową międzybranżową organizację normalizacyjną przyjętą przez MNO jako globalny punkt odniesienia.

z GSM Alliance (GSMA) i Europejskim Instytutem Norm Telekomunikacyjnych, a także instytucjami bankowymi (np. EMV) i innymi organami normalizacyjnymi, w tym forum NFC.

1.3.2.3. Tryb NFC P2P

Tryb NFC P2P umożliwia dwóm urządzeniom obsługującym technologię NFC wymianę informacji i udostępnianie treści. Tryb NFC P2P można wykorzystać do sparowania innego urządzenia NFC (np. komputera, głośnika, słuchawek, telewizora) i nawiązania dodatkowego szybkiego połączenia, takiego jak Bluetooth® lub Wi-Fi (np. poprzez wymianę parametrów konfiguracji). W tym przypadku NFC służy do negocjowania drugiego kanału komunikacyjnego i przesyłania danych uwierzytelniających dla drugiego protokołu. Plik lub dane (np. plik multimedialny: zdjęcia, filmy lub pliki audio) są następnie przesyłane przez protokół o wysokiej przepustowości. Istnieją dwa standardowe tryby pracy (rola pasywna lub aktywna) (patrz Tabela 1.1):

- NFC IP-1 (inicjator i docelowe urządzenia NFC);
- *Protokół kontroli łącza logicznego NFC (LLCP)* (kolejno tryb aktywny/tryb pasywny - tryb pasywny/tryb aktywny).

Standardy te zostały przedstawione w sekcji 1.3.3.

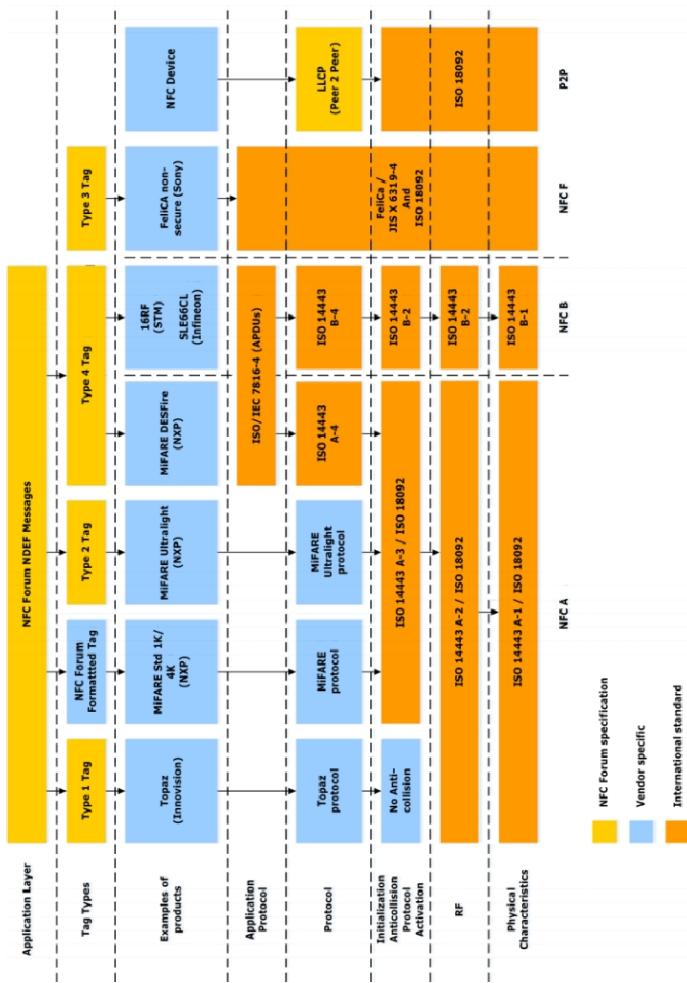
NFC tryb	Aktywny	Pasywny
Aktywny	P2P	Emulacja karty
Pasywny	R/W	-

Tabela 1.1. Trzy standardowe tryby NFC

1.3.3. Standardy forum NFC

NFC Forum zostało założone w 2004 roku przez Philips Semiconductors (ówczes NXP) i Nokia, wynalazców NFC, a następnie dołączyła do nich firma Sony. Jest to stowarzyszenie branżowe non-profit, którego celem jest

rozwija i promuje technologię NFC. Specyfikacje forum NFC opierają się na standardzie ISO/IEC 14443, który nieznacznie różni się od standardu ISO/IEC 18092 pod względem warstw RF i trybu NFC P2P, który został uwzględniony przez forum NFC w 2011 roku (patrz rysunek 1.14).



Rysunek 1.14. Standardy forum NFC¹⁶



Rysunek 1.15. Znak N forum NFC

UWAGA: - Użycie oznaczenia "N" (patrz rysunek 1.15.) wskazuje na zgodność z przepisami.

z forum NFC specyfikacje. licencja umowa dla
Podpisywanie

używanie marki jest wymagane w celu uzyskania znaku N.

Dokumenty aplikacyjne forum NFC zawierają wskazówki dotyczące tego, jak skorzystać z bezdotyko Technologia NFC w określonym celu wy

scenariusze. Na przykład w dokumentach aplikacyjnych (tj. dla NFC) dostarczonych przez forum NFC możemy znaleźć wskazówki dotyczące korzystania z NFC do bezpiecznego parowania Bluetooth®: ten dokument aplikacyjny opisuje interakcje między technologiami Bluetooth® i NFC. Zawiera przykłady zarówno implementacji protokołu, jak i transferu danych dla najbardziej odpowiednich przypadków użycia tych dwóch technologii. Deweloperzy znajdują w nim przydatne wskazówki dotyczące ich własnej pracy. Dokument został rozszerzony (w czerwcu 2014 r.) o technologię *Bluetooth Low Energy*, wersję technologii Bluetooth, która oferuje zmniejszone zużycie energii (NFC

forum, "Bluetooth Bezpieczne i proste Korzystanie z aplikacji NFC parowanie Document", 2011).

W ten sposób, gdy są używane ze specyfikacjami forum NFC, narzędzia te może pomóc w osiągnięciu pełnej interoperacyjności między technologiami: technicznymi

Dokumentacja i narzędzia sugerowane przez forum NFC oferują wskazówki dotyczące najlepszych praktyk i rozwiązań wdrożeniowych NFC (bezpłatne dla członków forum NFC, płatne dla osób niebędących członkami), wymienione poniżej:

- 1) Protokół interfejsu komunikacyjnego NFCIP-1 (ISO/CEI 18092) znormalizowany przez normy ISO/IEC 18092 i ECMA-340 (odpowiedni dla urządzeń parowanych w trybie P2P). Stos protokołów

w NFCIP-1

jest umieszczony protokołu NFC-A
na górze

zdefiniowane w normie ISO/IEC 14443 i tagi typu 3 forum NFC (patrz sekcja
1.3.3.3.1). NFCIP-1 obejmuje zarówno komunikację pasywną, jak i aktywną

Najnowocześniejsza

technologia NFC

29

warstwa sekcja 1.3.1).

(zob.

które umożliwiają urządzeniu NFC komunikację z innymi urządzeniami NFC w trybie P2P:

– *Aktywny* tryb komunikacji oznacza, że zarówno inicjator, jak i cel używają własnego, samodzielnie generowanego i modulowanego pola RF do przesyłania danych;

– Tryb komunikacji *pasywnej* oznacza, że urządzenie docelowe NFC odpowiada na polecenie inicjatora w schemacie modulacji obciążenia (wykorzystując energię indukowaną przez pole RF generowane przez inicjatora).

UWAGA - NFCIP-2 (ISO/IEC 21484) to specyfikacja mechanizmu wyboru między różnymi technologiami zbliżeniowymi działającymi na tej samej częstotliwości 13,56 MHz. Obsługuje komunikacje zgodnie z ISO/IEC 18092, ISO/IEC 14443. Specyfikacja ta jest również kompatybilna z innymi technologiami zbliżeniowymi, takimi jak karty Vicinity (ISO/IEC 15693), znane również jako NFC-V, obecnie przeglądane przez forum NFC (tj. w celu zaoferowania specyfikacji obejmującej bliską zgodność ze standardem NFC).

2) Norma ISO/IEC 14443 opisuje protokoły interfejsów komunikacyjnych bezstykowych układów scalonych podzielonych na cztery części:

- 14443-1: właściwości fizyczne;
- 14443-2: Zasilanie RF i interfejs sygnału;
- 14443-3: inicjalizacja i mechanizmy antykolizyjne;
- 14443-4: protokół transmisji.

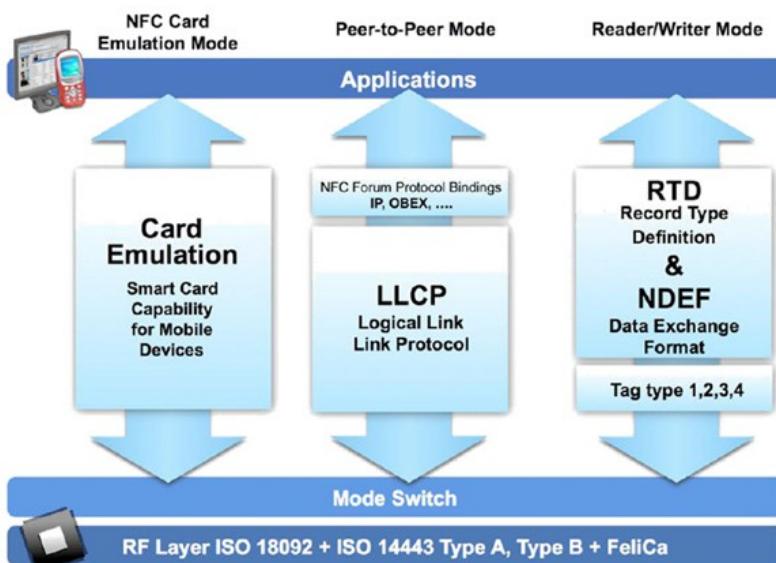
3) Logiczny format wymiany danych NDEF (patrz sekcja 1.3.3.2).

1.3.3.1. Specyfikacje protokołów forum NFC

Standardowy dokument specyfikacji wyraźnie opisuje zestaw wymagań dotyczących kryteriów technicznych, metod i procesów. W inżynierii systemów i oprogramowania ma on na celu ustanowienie wytycznych dotyczących rozwoju i wdrażania systemu (oprogramowania lub sprzętu), opisując na przykład, co musi / może / powinno lub nie powinno być zrobione

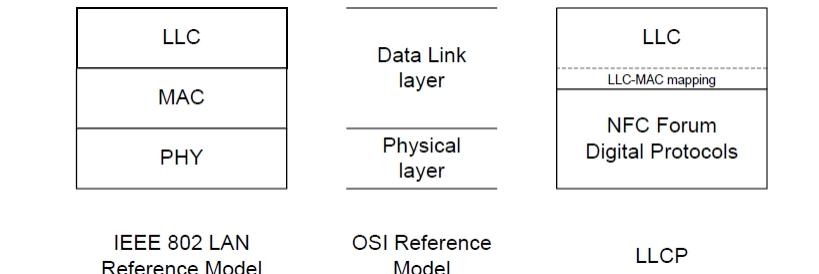
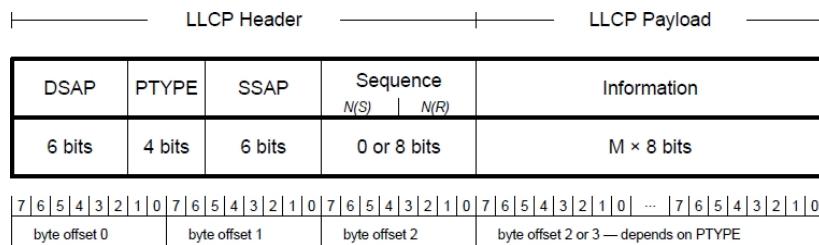
i w jaki sposób. Każdy dostawca może opracować własną implementację specyfikacji.

UWAGA: - Forum NFC proponuje program certyfikacji dla dostawców, którzy chcą certyfikować swoje produkty i zapewnić ich zgodność ze specyfikacjami forum NFC. Specyfikacje forum NFC są związane z protokołami komunikacyjnymi i warstwami aplikacji (przesyłania wiadomości) modelu OSI, w tym z typami tagów.



Rysunek 1.16. Stosy standardów NFC (źródło: forum NFC)

Protokół NFC LLCP oparty na kodzie uwierzytelniania wiadomości warstwy 2 standardu IEEE 802.2 Open Systems Interconnection (OSI) (patrz rysunek 1.17), ISO/IEC 18092 i ISO/IEC 14443 obsługujący P2P między dwoma urządzeniami obsługującymi NFC do komunikacji dwukierunkowej. Jest on używany w górnej części NFCIP-1.

**Rysunek 1.17.** Protokół NFC LLCP i model OSI¹⁷**Rysunek 1.18.** Format PDU LLCP¹⁸

Specyfikacja protokołu LLCP opisuje format struktur wymiany jednostek danych protokołu (PDU) składających się z nagłówka i ładunku (patrz rysunek 1.18):

- Format PDU LLCP jest następujący:
 - docelowy punkt dostępu do usług (DSAP);
 - typ ładunku (PTYPE);
 - punkt dostępu do usługi źródłowej (SSAP);
 - Sekwencja PDU;

¹⁷ Źródło: Forum NFC, Rysunek 1, s. 4 w *Logical Link Control Protocol Technical Specification 1.0*, grudzień 2009.

¹⁸ Źródło: Forum NFC, rysunek 3, s. 14 w *Logical Link Control Protocol Technical Specification 1.0*, grudzień 2009.

- Ładunek.

– Cyfrowy protokół NFC oparty na standardach ISO/IEC 18092 i ISO/IEC 14443 oraz JIS X6319-4 gromadzi wspólne cechy i interfejs cyfrowy oraz protokół transmisji półdupleksowej urządzenia obsługującego NFC w czterech rolach: inicjatora, celu, czytnika/zapisu i emulatora karty.

– Aktywność NFC zapewnia konfigurację komunikacji między urządzeniami NFC: specyficzne dla profili parametry konfiguracyjne, odpisywanie dla tagu NFC lub urządzenia NFC w połączeniu, dane NDEF itp.

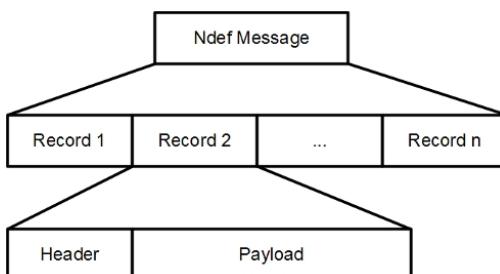
– Prosty protokół wymiany NFC NDEF dla urządzeń obsługujących NFC przesyłających wiadomości przez LLCP.

– Analog NFC: Interfejs RF (postać sygnału, charakterystyka czasowa/częstotliwościowa/modulacyjna) i zapotrzebowanie na moc urządzenia obsługującego NFC w jego czterech rolach (inicjator trybu P2P, cel trybu P2P, tryb czytnika/zapisu i tryb emulacji karty) dla wszystkich trzech technologii: NFC-A, NFC-B i NFC-C (w zależności od metod modulacji/demodulacji sygnału) oraz dla wszystkich różnych przepływności (106, 212 i 424 kb/s).

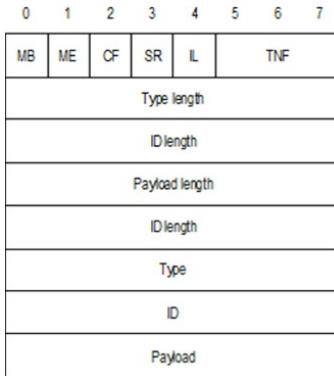
– Interfejs kontrolera NFC pomiędzy kontrolerem NFC a głównym procesorem aplikacji urządzenia.

1.3.3.2. Format wymiany danych NDEF

NDEF definiuje strukturę formatu danych komunikatu NFC, który ma być odczytany lub zakodowany w tagu. Komunikaty NDEF mają nagłówek i jeden lub kilka rekordów NDEF z nagłówkiem dla każdego z nich.



Rysunek 1.19. Struktura komunikatów NDEF

**Rysunek 1.20.** Struktura rekordu NDEF

NDEF Record Type	Description	Full URI Reference	Specification Reference
Sp	Smart Poster	urn:nfc:wkt:Sp	NFC Forum Smart Poster RTD
T	Text	urn:nfc:wkt:T	NFC Forum Text RTD
U	URI	urn:nfc:wkt:U	NFC Forum URI RTD
Gc	Generic Control	urn:nfc:wkt:Gc	NFC Forum Generic Control RTD ¹⁹
Hr	Handover Request	urn:nfc:wkt:Hr	NFC Forum Connection Handover Specification
Hs	Handover Select	urn:nfc:wkt:Hs	NFC Forum Connection Handover Specification
Hc	Handover Carrier	urn:nfc:wkt:Hc	NFC Forum Connection Handover Specification
Sg	Signature	urn:nfc:wkt:Sg	NFC Forum Signature RTD

Tabela 1.2. Dobrze znane typy rekordów NDEF¹⁹

19 Źródło: Forum NFC (<http://www.nfc-forum.org/specs/nfcforumassignednumbersregister>).

Jak zilustrowano na rysunku 1.20, pierwsze bajty są używane przez nagłówek: - MB: wiadomość rozpoczyna się od 1 bitu, przyjmując wartość 1, jeśli wiadomość uruchamia się, w przeciwnym razie 0.

- ME: zakończenie wiadomości na 1 bicie przyjmującym wartość 1, jeśli wiadomość się kończy, w przeciwnym razie 0.
- CF: bit wskazuje na podzielony ładunek.
- SR: bit wskazuje krótki zapis na 7 bajtach zamiast 10.
- IL: bit wskazuje, czy należy odczytać długość bajtów ID i ID.
- TNF: format nazwy typu na 3 bitach.
- 0x00: pusty rekord.
- 0x01: dobrze znany typ zdefiniowany przez forum NFC (patrz Tabela 1.2).
- 0x02: Typ MIME (tekst, media, obraz itp.).
- 0x03: URI.
- 0x04: zewnętrzny.
- 0x05: nieznany typ.
- 0x06: bez zmian (dla rekordów zbiorczych).
- 0x07: zarezerwowane do wykorzystania w przyszłości.
- Długość typu: długość pola PTYPE wynosząca 8 bajtów.
- Długość ID: rozmiar pola ID ładunku na 8 bajtów.
- Długość ładunku: określa długość pola ładunku, którego rozmiar jest określony przez pole SR.
- Typ: typ rekordu (patrz tabela 1.2) w systemie szesnastkowym (np. "U" dla URI, wartość 0x55).
- ID: identyfikator typu lub prefiks szesnastkowy (np. 0x01 to "http://www").
- Prefiks: zgodnie z TNF, na przykład dla typu URI:
 - 0x00: brak prefiksu;
 - 0x01: http://www;

- 0x02: https://www.
- 0x03: http://
- 0x04: https://
- 0x05: tel:
- 0x06: mailto:
- 0x1D: file://
- 0x24...0xFF: zarezerwowane do wykorzystania w przyszłości.
 - Payload: zawartość, której rozmiar jest określony przez pole długości payloadu.

1.3.3.3. Typy tagów forum NFC

Zawartość i pojemność tagów NFC różni się w zależności od ich charakterystyki, od tagów blokujących i odblokowujących możliwość zapisu, przechowujących jedną lub kilka wiadomości powiązanych z treściami internetowymi, po bardziej złożone tagi, takie jak karty inteligentne, zdolne do osadzania danych, a także aplikacji.

1.3.3.3.1. Typy tagów NFC

Cztery typy tagów NFC (patrz Tabela 1.3) są zdefiniowane zgodnie z różnymi standardami i protokołami.

NFC Forum type	Tags type 1	Tags Type 2	Tags type 3	Tags type 4
Memory	96 Bytes - 2KB	96 Bytes - 2KB	Up to 1MBytes (service)	4 - 32KB
Rate	106Kbits/s	106Kbits/s	212 or 424 Kbits/s	106, 212 or 424 Kbits/s
Data access	R/W or Read only	R/W or Read only	R/W or Read only	R/W or Read only
Collision handling	No	Yes	Yes	Yes
Compliance	Broadcom Topaze	NXP Mifare Ultralight, NXP Mifare Ultralight C, NXP NTAG203	Sony FeliCa	NXP DESFire / NXP SmartMX-JCOP
Standard	ISO14443A	ISO14443A	Japan Industrial Standards (JIS) X 6319-4	ISO14443A et ISO14443B
Price	Low cost	Low cost	Expensive	Medium to expensive

Tabela 1.3. Cztery rodzaje tagów NFC

W tabeli 1.4 poziom protokołu w fazie aktywacji komunikacji i wymiany danych rozróżnia typy tagów.

		Standard			
Collisions management	Protocol NFC-DEP (P2P) based on ISO 18092	ISO 18092 NFCIP - 1 (ECMA 340)		ISO 21481 NFCIP – 2 (ECMA 352)	ISO 18092 NFCIP – 1 (ECMA 340)
		NFC-A		NFC-B	NFC-F
Activation	Protocol NFC-DEP (P2P) based on ISO 18092	Platforme			
		ISO 14443A		ISO 14443B (Proximity) & 15693 (Vicinity)	JIS X 6319-4
Data exchange, Deactivation	Protocol Half-Duplex cmd RW Topaz Innovation (Broadcom)	Type 1	Type 2	Type 4A	Type 3
		Protocol Half-Duplex cmd RW Mifare Ultra Light (NXP)	Protocol Half-Duplex cmd RW Mifare Ultra Light (NXP)	Protocol ISO-DEP based on ISO 14443 (4) & EMV_CLESS	Protocol Half- Duplex cmd RW Felica (Sony)

Tabela 1.4. Standardy komunikacji NFC

Wszystkie tagi typu 1-4 są oparte na istniejących produktach bezstykowych:

– Tag typu 1 forum NFC oparty na standardzie ISO/IEC 14443 jest zdolny do przepisania 96 bajtów (z możliwością rozszerzenia do 2 kb). Innovision Topaz Tag (firma Broadcom) jest najczęściej używanym tagiem typu 1.

– Znacznik typu 2 forum NFC oparty na standardzie ISO/IEC 14443A ma możliwość przepisania ponad 48 bajtów (z możliwością rozszerzenia do 2 kb). Tagi Mifare Ultralight i NTAG (NXP) są najczęściej używanymi tagami typu 2.

– Tag NFC forum typu 3 jest wstępnie skonfigurowany podczas produkcji do odczytu i wielokrotnego zapisu lub tylko do odczytu, teoretyczny limit pamięci wynosi 1 MB na usługę. Tagi typu 3 są oparte na japońskim standardzie przemysłowym (JIS X 6319-4); najczęściej używane są tagi Felica (Sony).

– Tag NFC forum typu 4 (zdefiniowany w listopadzie 2010 r.) jest w pełni zgodny z serią standardów ISO/IEC 14443. Tagi są wstępnie konfigurowane podczas produkcji do odczytu i wielokrotnego zapisu lub tylko do odczytu, zmenna pamięci do 32 kb na usługę; interfejs komunikacyjny jest zgodny z typem A lub typem B. Te typy znaczników można łączyć na poziomie aplikacji zgodnie z normą ISO/IEC 7816-4 Application Protocol Data Unit (APDU). Te typy

4 tagi są zwykle używane do płatności zbliżeniowych i sprzedaży biletów, takich jak Mifare DESfire (NXP).

1.3.3.3.2. Definicja typu rekordu

Typy rekordów używane przez aplikację forum NFC i strony trzecie są oparte na formacie danych NDEF opisany w sekcji 1.3.3.2. Zdefiniowano pięć określonych RTD (patrz tabela 1.2): Text, URI, Smart Poster, Generic Control i Signature.

1.3.3.3.3. Aplikacje referencyjne

Przekazywanie połączenia forum NFC zostało określone w 2010 roku, aby umożliwić statyczną i dynamiczną konfigurację NFC za jednym dotknięciem z szybkimi technologiami komunikacyjnymi, takimi jak konfiguracja Wi-Fi lub parowanie Bluetooth®.

Forum NFC posiada również standard komunikacji z urządzeniami peryferyjnymi dedykowany technologiom opieki zdrowotnej (NFC Forum Personal Health Device Communication) i oparty na standardzie ISO/IEC/IEEE 11073- 20601 (Health informatics).

1.3.4. GlobalPlatform (GP)

W oparciu o standary "Visa© OpenPlatform" w rękach konsorcjum OpenPlatform, przekształconego w GlobalPlatform (GP) w 1999 r., GP został stworzony w celu zaspokojenia potrzeb interoperacyjności wynikających z globalizacji transakcji płatniczych za pomocą kart inteligentnych; jego głównymi celami było promowanie bezpieczeństwa, platform programistycznych i zarządzania wdrażaniem kart inteligentnych oraz standaryzacji SE.

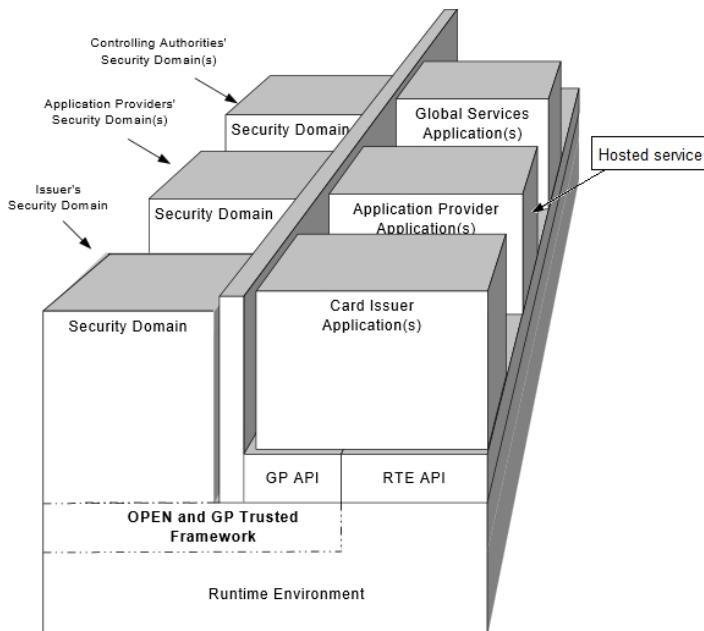
GP działa na rzecz interoperacyjności podmiotów ekosystemu NFC w trybie emulacji karty, a zwłaszcza na rzecz zarządzania SE. Wydaje specyfikacje i instrukcje dotyczące ekosystemu, w szczególności zasad bezpieczeństwa, protokołów, architektur i interfejsów kart (specyfikacje kart²⁰), urządzeń (specyfikacje urządzeń²¹) i systemów.

20 <http://www.globalplatform.org/specificationscard.asp>.

21 <https://www.globalplatform.org/specificationsdevice.asp>.

(specyfikacje systemów²²); GP definiuje scenariusze przypadków użycia, wzorce projektowe, ramy i interfejsy API w celu ułatwienia wdrażania usług SE (niektóre zasoby mają ograniczony dostęp: bezpłatny dla członków i płatny dla osób niebędących członkami).

1.3.4.1. Specyfikacje kart GP



Rysunek 1.21. Architektura GP karty²³

Specyfikacje w tej sekcji dotyczą architektury, interakcji, interfejsów i bezpieczeństwa kart inteligentnych, niezależnie od podstawowych technologii (tj. niezależnie od producenta lub systemu operacyjnego karty). Zasoby w sekcji kart bardziej szczegółowo odnoszą się do wydawców kart inteligentnych (mających na celu hostowanie aplikacji innych firm).

22 <http://www.globalplatform.org/specificationssystems.asp>.

23 Źródło: Global Platform, Rysunek 3-1, str. 38 w GP Card Specification V2.2, marzec 2006.

Rysunek 1.21 przedstawia usługi hostowane w dedykowanym bezpiecznym środowisku wykonawczym, domenie bezpieczeństwa (SD), w tym warstwę API zgodną ze standardem GP do zarządzania kluczami, szyfrowania / deszyfrowania, podpisu i kontroli dostępu, obsługę przenoszenia aplikacji między kartami zgodnymi z GP. Możemy zauważać SD przeznaczoną dla dostawcy kart, podobnie jak inną dedykowaną dla organu kontrolującego (do zarządzania kluczami i certyfikatami). "OPEN" API zapewnia standaryzowane funkcje do łączenia i zarządzania cyklem życia usług zgodnych z GP, w tym ramy bezpiecznej komunikacji między aplikacjami.

OPEN obsługuje następujące funkcje zarządzania:

- wykonywanie poleceń;
- aplikacja lub wybór SD;
- logiczny kanał zarządzanie (tj. pomijanie kanałów fizycznych);
- wysyłanie poleceń;
- zarządzanie zawartością karty;
- kontrola zawartości;
- konfiguracja zawartości;
- usuwanie treści;
- reguły kontroli dostępu do zarządzania treścią;
- zarządzanie bezpieczeństwem;
- blokowanie/odblokowywanie;
- terminal kart;
- zarządzanie uprawnieniami;
- identyfikowalność i dziennik zdarzeń.

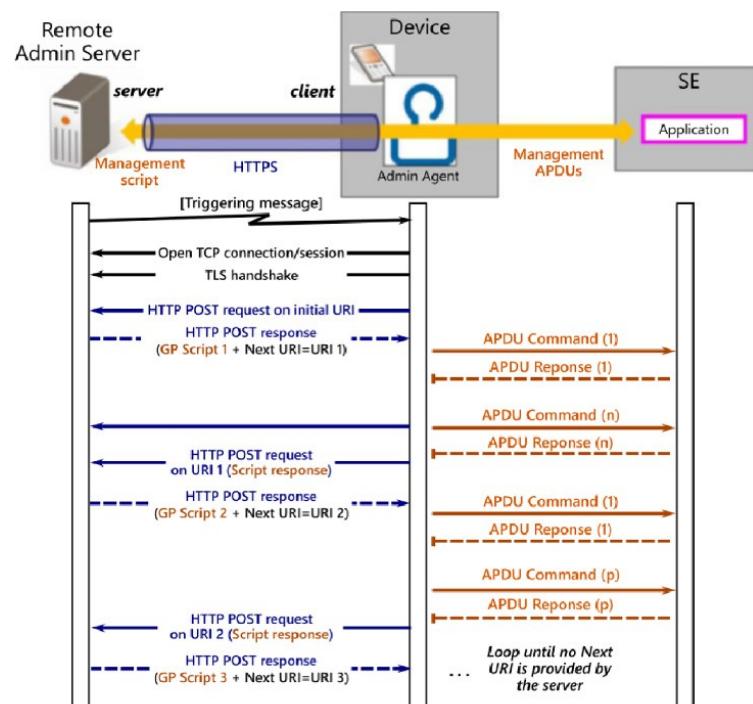
Specyfikacje dokładnie opisują przejścia stanów w cyklu życia, instrukcje APDU do zarządzania zawartością karty i zalecenia dotyczące konfiguracji bezpiecznego kanału komunikacji (PKI, tryby uwierzytelniania, szyfrowanie i deszyfrowanie, konfiguracja sesji itp.)

1.3.4.2. Dane techniczne urządzenia GP

Sekcja zasobów urządzeń jest poświęcona oryginalnym urządzeniom producentów, a w szczególności dla środowisk mobilnych. Specyfikacje są również przeznaczone dla programistów, którzy łączą się z urządzeniami peryferyjnymi wymienionymi w tej sekcji.

Ta sekcja dotyczy dwóch środowisk:

- SE z własnym systemem operacyjnym;
- zaufane środowisko wykonawcze (TEE) działające w pamięci urządzenia hosta.



Rysunek 1.22. Przykład zdalnej administracji SE²⁴

24 Źródło: Global Platform, Rysunek 6-1, str. 31 w *Secure Element Remote Application Management*, wersja 1.0.1, listopad 2015.

TEE obejmuje bezpieczne przechowywanie, w którym funkcje są wykonywane w odizolowanym obszarze przestrzeni pamięci od systemu operacyjnego (oddzielone zaporami ogniwymi); komunikacja z urządzeniami (na przykład ekranem, kartą SIM lub SE) wykorzystuje bezpieczne kanały, zabezpieczając w ten sposób prywatność instrukcji kodowania i tak zwane "zaufane" dane aplikacji.

Aplikacje i narzędzia zalecane przez GP są związane z architekturą, kontrolą dostępu i interfejsami komunikacyjnymi lokalnych (tj. z urządzenia hosta) lub zdalnych (tj. z zewnętrznego serwera) aplikacji klienckich. Na przykład rysunek 1.22 pokazuje, w jaki sposób żądanie sesji administracyjnej jest wyzwalane na żądanie zdalnego serwera administracyjnego w kierunku aplikacji *Admin Agent* w urządzeniu mobilnym; następnie telefon komórkowy ustanawia bezpieczne połączenie HTTPS (wymagające uwierzytelnienia) z serwerem w celu zebrania instrukcji APDU do przesłania do SE.

1.3.4.3. Specyfikacje systemów GP

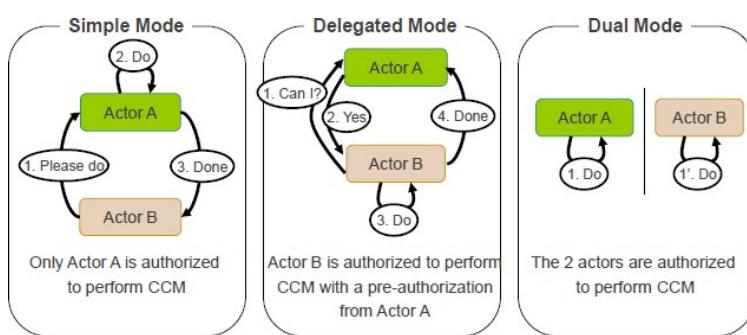
Ta sekcja jest przeznaczona dla każdego interesariusza, który projektuje i wdraża systemy administracyjne, a także systemy zarządzania kartami inteligentnymi i ich zawartością: specyfikacje te określają rolę i obowiązki każdego uczestnika ekosystemu w bezpiecznej infrastrukturze środowiska wielu kart aplikacji oraz opisują format wymiany wiadomości, protokoły i interfejsy.

Specyfikacje obejmują platformę mediatora zwaną zaufanym menedżerem usług (TSM). TSM zapewnia znaleziony interfejs API, który umożliwia dostawcom usług NFC hostowanych w SE łączenie się z systemami zarządzania emitentów SE (SEI) lub MNO, które obsługują subskrypcje użytkowników końcowych. GP opisuje bezpieczne usługi internetowe SOAP w plikach XML (WSDL i XSD) przeznaczonych do zarządzania cyklem życia (i audytu) zdalnych aplikacji, takich jak (między innymi):

- wdrożenie usługi;
- zawieszenie/blokada/odblokowanie;
- aktualizacja;
- usunięcie usługi.

Rysunek 1.23 ilustruje zarządzanie treścią SE uruchamiane przez "Aktora B" (na przykład w imieniu dostawcy usług, tj. TSM) autoryzowanego przez "Aktora A" kontrolującego SE (SEI lub MNO) zgodnie z trzema trybami delegowania:

- *W trybie prostym tylko SEI jest upoważniony do zarządzania zawartością karty. TSM może weryfikować ładowanie za pomocą wzorca uwierzytelniania danych (podpisu);*
- *w trybie delegowanym TSM jest upoważniony do zarządzania zawartością karty poprzez uzyskanie uprzedniej autoryzacji od SEI z zarządzaniem tokenami;*
- *w trybie autoryzowanym (podwójnym) TSM ma pełny dostęp do dedykowanego SD po uzyskaniu uprzedniej autoryzacji od SEI.*

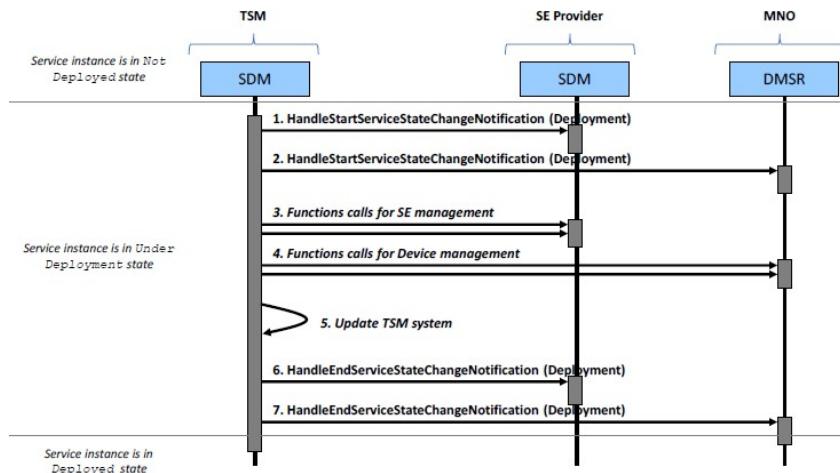


Rysunek 1.23. Trzy tryby zarządzania treścią SE²⁵

System komunikatów GP (GPM) jest standardowym interfejsem do zdalnego zarządzania cyklem życia usługi przez TSM, jak pokazano na rysunku 1.24. Rysunek ilustruje przykład powiadomienia o wdrożeniu (i wywołania funkcji) wyzwanego przez TSM do dostawcy SE, a także do MNO; następnie każdy z nich może zaktualizować stan usługi i podjąć odpowiednie działania. Dla każdego uczestnika tego ekosystemu, GPM dokładnie opisuje kompleksowe przypadki użycia w dokumencie

25 Źródło: Global Platform, rysunek 1-1, s. 26 w *Messaging Specification for Management of Mobile-NFC Services*, wersja 1.2, listopad 2015.

trzy tryby zarządzania usługami SE, dla każdego kroku i zdarzenia cyklu życia.



Rysunek 1.24. Powiadomienie o wdrożeniu przez TSM²⁶

1.3.5. SIMAlliance i otwarte mobilne API

SIMAlliance jest organizacją non-profit zajmującą się technologiami urządzeń mobilnych, a w szczególności kartami SIM. SIMAlliance ma na celu ułatwienie rozwoju i zarządzania bezpiecznymi usługami mobilnymi oraz uproszczenie sprzętowych zabezpieczeń urządzeń. Członkami organizacji są podmioty zajmujące się bezpieczeństwem cyfrowym, kartami inteligentnymi i usługami mobilnymi, takie jak Gemalto, Oberthus, Giesecke & Devrient (G&D), Incard i Morpho (Safran).

SIMAlliance podaje specyfikacje swojego standardu Open Mobile API (OMAPI)²⁷ dla aplikacji mobilnych: OMAPI definiuje interfejs podobny do czytnika (tj. czytnik kart inteligentnych) z SE niezależnie od konfiguracji (tj. SIM-SE, eSE lub MicroSD); OMAPI był

26 Źródło: Global Platform, rysunek 3-5, s. 131 w *Messaging Specification for Management of Mobile-NFC Services*, wersja 1.2, listopad 2015.

27 Zob. <http://simalliance.org/se/se-technical-releases>.

został przyjęty jako standard przez GP, a API zostało dostosowane w celu zapewnienia zgodności ze specyfikacjami ETSI i GSMA. EMVco zleciło również SIMAlliance dostosowanie API do płatności zbliżeniowych (NFC Tagify, luty 2016).²⁸

28 Zob. <https://nfctagify.com/simalliance-updates-omapi-secure-element-specification>.

Tworzenie aplikacji NFC w systemie Android

Aplikacje NFC mogą być tworzone w trzech trybach pracy NFC (patrz sekcja 1.3.2):

- W trybie *czytnika/zapisu* smartfon NFC działa jako aktywny terminal inicjujący (generujący pola RF jako źródło zasilania) w celu odczytania i zakodowania pasywnego docelowego tagu NFC;
- Tryb *P2P* umożliwia komunikację między dwoma urządzeniami obsługującymi NFC: mogą one wymieniać dane, będąc zarówno inicjatorem, jak i celem (w trybie aktywnym/aktywnym lub aktywnym/pasywnym) dzięki warstwie LLCP (patrz sekcja 1.3.4.1) i jej zarządzaniu kolizjami;
- w trybie *emulacji karty* pasywne urządzenie docelowe z obsługą NFC zachowuje się jak karta inteligentna; w tym konkretnym trybie smartfon może hostować jedną lub kilka aplikacji terminalowych wykonywanych w środowisku systemu operacyjnego hosta, które mogą działać jako interfejs użytkownika i/lub jako aplikacja pośrednicząca (proxy) do łączenia się z systemem zdalnym, oraz jedną lub kilka usług wykonywanych w bezpiecznym środowisku SE (patrz sekcja 1.3.2.2).

Smartfon lub tablet NFC może działać jako czytnik/koder tagów NFC w trybie czytnika/zapisu. W trybie P2P smartfon może również wymieniać dane z innym urządzeniem NFC. W trybie emulacji karty

W trybie tym smartfon może działać jako bezstykowa karta inteligentna NFC (z usługą SE lub HCE), która może być "odczytywana" przez czytnik NFC.

W tym rozdziale przedstawiono programowanie aplikacji mobilnych na platformie Android w trzech trybach NFC, w tym w trybie HCE.

UWAGA: - W tym rozdziale przedstawiamy krótkie wprowadzenie do tworzenia aplikacji na Androida za pomocą Eclipse. Wymaga to podstawowej wiedzy w zakresie programowania, a zwłaszcza dobrego zrozumienia języka Java.

2.1. Wprowadzenie do programowania w systemie Android przy użyciu Eclipse

Kilka zintegrowanych środowisk programistycznych (IDE) umożliwia tworzenie aplikacji na Androida (w trybie natywnym). Wśród najczęściej używanego wolnego i otwartego oprogramowania, przykładami są Eclipse^{TM1}, którego będziemy używać z wtyczką *Android Development Toolkit*² (ADT), IntelliJ IDEA firmy JetBrains używaną z wtyczką *idea-Android*³, Netbeans używaną z podstawową wtyczką *NBAndroid*⁴ itp.

Jednak oficjalnym i darmowym IDE Google, które pojawiło się w 2013 roku, jest *Android Studio*⁵: to IDE jest oparte na technologii IntelliJ IDEA w wersji open source. Programista specjalizujący się w kodowaniu aplikacji na Androida będzie raczej korzystał z Android Studio.

2.1.1. Android w pigułce

Android to system operacyjny typu open source (oparty na jądrze Linux) opublikowany przez Google. Środowisko uruchomieniowe aplikacji Android jest zoptymalizowaną wielozadaniową maszyną wirtualną dla urządzeń mobilnych.

1 Pobieranie Eclipse: <http://www.eclipse.org/downloads>.

2 Adres URL wtyczki Android dla Eclipse: <https://dl-ssl.google.com/android/eclipse>.

3 Pobierz wtyczkę Android dla IntelliJ IDEA: <http://plugins.jetbrains.com/plugin/1792?pr=>.

4 Pobierz wtyczkę Android dla Netbeans: <http://plugins.netbeans.org/plugin/19545>.

5 Pobierz Android Studio: <https://android.googlesource.com/platform/tools/adt/idea>.

urządzeń (korzystających z małej ilości pamięci) o nazwie *Dalvik*. Język programowania dla aplikacji Android jest oparty na Java: po skompilowaniu klas Java w kodzie bajtowym, są one konwertowane do pliku "classes.dex", który jest interpretowany i przetwarzany przez Dalvik. Plik wykonywalny jest skompresowanym plikiem typu archiwum (jar⁶) z rozszerzeniem "APK" (Android PacKage).

UWAGA - Każda aplikacja Android działa w dedykowanym obszarze pamięci i otrzymuje unikalny identyfikator użytkownika (identyfikator użytkownika Linux). Co więcej, pliki danej aplikacji Android mogą być widziane (domyślnie) tylko przez tę aplikację, co zwiększa bezpieczeństwo.

Plik deklaracji *AndroidManifest.xml* (w formacie XML) jest wymagany do uruchomienia aplikacji Android: przechowuje nazwę pakietu aplikacji, jego wersję, zgodność wersji Androïda, uprawnienia, które mają zostać przyznane przez użytkownika w celu korzystania z funkcji wymaganych przez aplikację, deklaracje i klucze dostępu Google API (pobierane z konsoli programisty Google dla Map, Google Play, Street view, Google Cloud Messaging, Prediction, YouTube itp.), wszystkie działania i filtry działań (na przykład w celu zadeklarowania głównego działania, które zostanie wykonane podczas uruchamiania lub działania wykonywanego po wykryciu urządzenia z obsługą NFC).

UWAGA - Interfejsy użytkownika i menu są również opisane w plikach XML.

2.1.1.1. Wersje Androïda

Wersje Androïda są zarządzane na dwóch poziomach:

- Wersja systemu Android (system operacyjny zainstalowany na smartfonach);
- numer zestawu SDK (Software Development Kit), za pomocą którego aplikacja została skompilowana.

Główne wersje Androïda są oznaczone nazwą kodową, podczas gdy wersje wdrożone i SDK są ponumerowane (patrz Tabela 2.1).

⁶ Archiwum Java.

Rok	Wersje	Nazwa	SDK	Jądro Linux
2007	1.0	Szarlotka	1	2.6
2008	1.1	Bananowy split	2	
2009	1.5	Babeczka	3	2.6.27
	1.6	Pączek	4	2.6.29
2009-2010	2.0, 2.0.1, 2.1	Eclair	5, 6, 7	
2010-2011	2.2, 2.2.1, 2.3	Froyo	8	2.6.32
2010-2011	2.3, 2.3.3, 2.3.4, 2.3.5, 2.3.6, 2.3.7	Piernik	9, 10	2.6.35
2011-2012	3.0, 3.1, 3.2, 3.2.1, 3.2.2, 3.2.4, 3.2.6	Plaster miodu	11, 12, 13	2.6.36
	4.0.1, 4.0.2, 4.0.3, 4.0.4	Ice Cream Sandwich	14, 15	3.0.1
2012-2013	4.1, 4.1.1, 4.1.2, 4.2, 4.2.1, 4.2.2, 4.3	Jelly Bean	16, 17, 18	3.0.31, 3.0.4
2013-2014	4.4, 4.4.2, 4.4W.2	KitKat	19, 20	3.0.4
2014-2015	5.0, 5.0.1, 5.1.1	Lizak	21, 22	3.10
2015-2016	6.0, 6.0.1	Marshmallow	23	
2016	Oczekiwane 7,0	Nugat	24	4.4

Tabela 2.1. Wersje systemu Android⁷

2.1.1.2. Publikowanie aplikacji na Androida w Google Play

Konsola dla programistów Google Play⁸ umożliwia programistom/dostawcom aplikacji na Androida publikowanie ich aplikacji (rozszerzenie pliku "apk" o nazwie

⁷ Źródło: Wikipedia, dostępne pod adresem http://fr.wikipedia.org/wiki/Historique_des_versions_d'Android#cite_note-gingerbread-dev-blog-35.

⁸ Konsola dewelopera Android Google Play: patrz <https://play.google.com>.

maksymalny rozmiar 50 MB) w sklepie Google Play. W tym celu należy posiadać konto Google i uścić jednorazową opłatę rejestracyjną (25 USD).

Orientacyjnie, dozwolony zakres cen dla aplikacji na Androïda wynosi od 0,99 USD do 200 USD za dystrybucję na rynku amerykańskim zgodnie z tabelą cen lokalizacji Google⁹. Opłaty transakcyjne pobierane przez Google wynoszą 30% ceny detalicznej aplikacji.

2.1.2. *Android w Eclipse IDE*

Eclipse to szeroko rozpowszechniony, wielojęzyczny i działający na wielu systemach operacyjnych framework programistyczny typu open source (Windows, Linux i Mac OSX w wersjach 32- i 64-bitowych). Na platformie Eclipse można zainstalować dużą liczbę wtyczek; pozwala to na rozwój nie tylko wielu frameworków opartych na Javie (w tym J2EE, J2ME itp.), C / C ++, PHP, Python itp., ale także wielu narzędzi, na przykład do wspólnego zarządzania pracą i synchronizacji z repozytoriami Git¹⁰ lub Apache Subversion¹¹, a także narzędzia do testowania i wydajności kodowania (na przykład JUnit dla Java¹²) lub narzędzia do projektowania / modelowania i wiele innych. Pierwszym krokiem jest pobranie i zainstalowanie standardowej wersji Eclipse dostępnej pod adresem URL: <https://www.eclipse.org/downloads>.

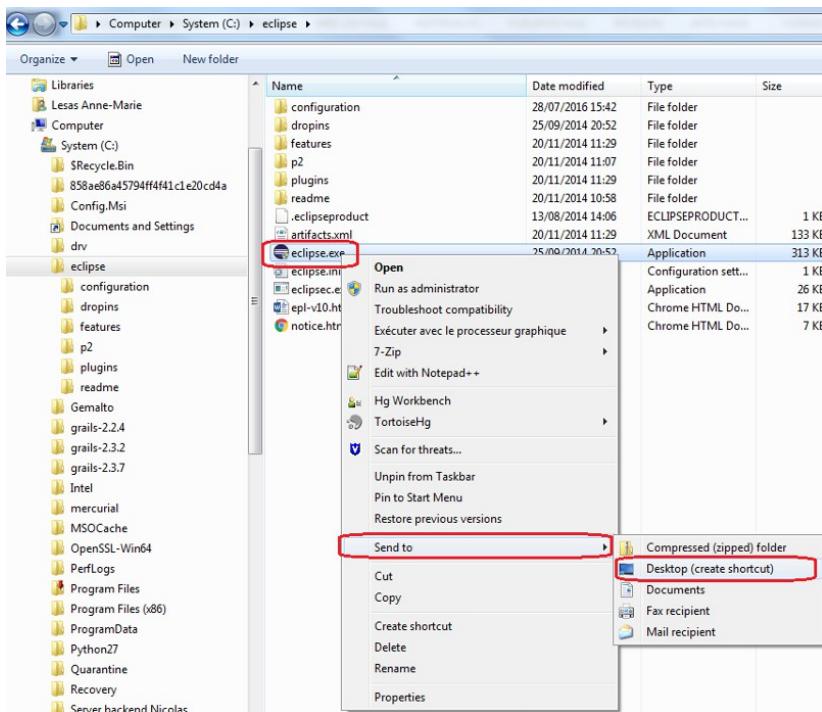
Eclipse jest uruchamiany poprzez uruchomienie pliku "eclipse.exe" (poprzez dwukrotne kliknięcie ikony lub z menu start) w folderze "Eclipse"; w systemie Windows można utworzyć skrót na pulpicie za pomocą menu skrótów (wyświetlanego po kliknięciu prawym przyciskiem myszy ikony pliku), jak pokazano na rysunku 2.1.

9 Zob. <https://support.google.com/googleplay/android-developer/table/3541286>.

10 Zob. <https://fr.wikipedia.org/wiki/Git>.

11 Zob. https://fr.wikipedia.org/wiki/Apache_Subversion.

12 Zob. <https://wiki.eclipse.org/Eclipse/Testing>.



Rysunek 2.1. Eclipse: utwórz skrót na pulpicie

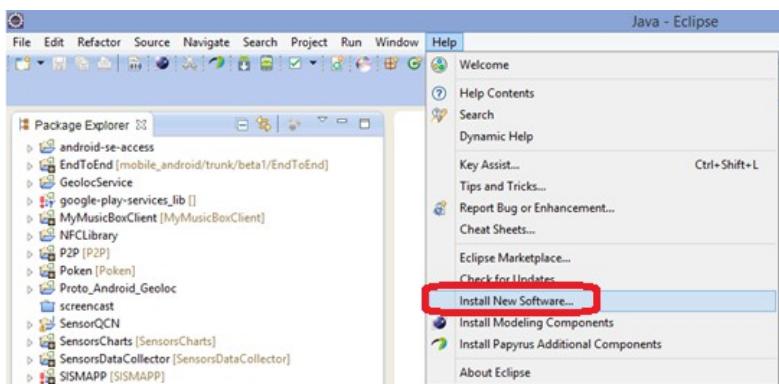
2.1.2.1. *Android Development Toolkit*

Android Development Toolkit (ADT) SDK można pobrać z adresu URL w celu samodzielnej instalacji: <http://developer.android.com/sdk/index.html>.

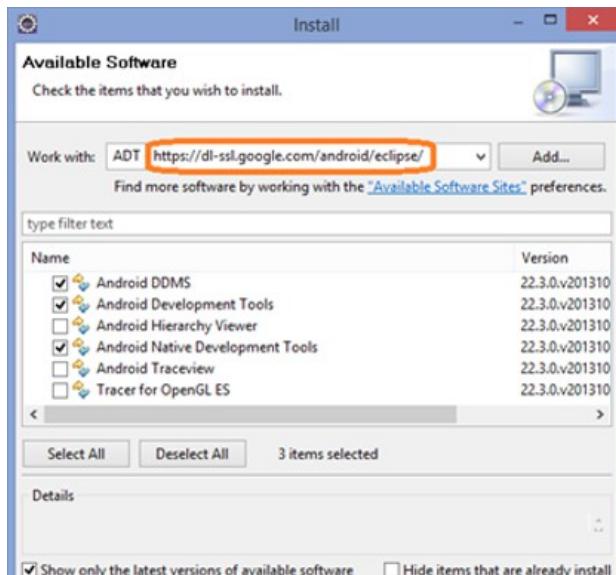
UWAGA - Google udostępnia dwa pakiety SDK dla jednej wersji, wersję Android o otwartym kodzie źródłowym, w pełni otwartą, oraz wersję dającą dostęp do interfejsów API Google, która nie jest otwarta i wymaga umowy licencyjnej (płatnej lub bezpłatnej).

Wtyczkę ADT dla Eclipse należy zainstalować za pomocą instalatora oprogramowania w menu *Help/Install new software* (patrz rysunek 2.2). Wprowadź adres URL <https://dl-ssl.google.com/android/eclipse>, aby zainstalować wtyczki Androida (patrz Rysunek 2.3) i wybierz komponenty przed kliknięciem przycisku

Przycisk *Next...* na pasku narzędzi. Może to zająć trochę czasu, ale po wyrażeniu zgody na wszystkie instalacje (wybierz zaufanie do oprogramowania w przypadku dialogu systemowego) i ponownym uruchomieniu Eclipse, nowe menu i narzędzia są dostępne w perspektywie Java (patrz rysunek 2.4).

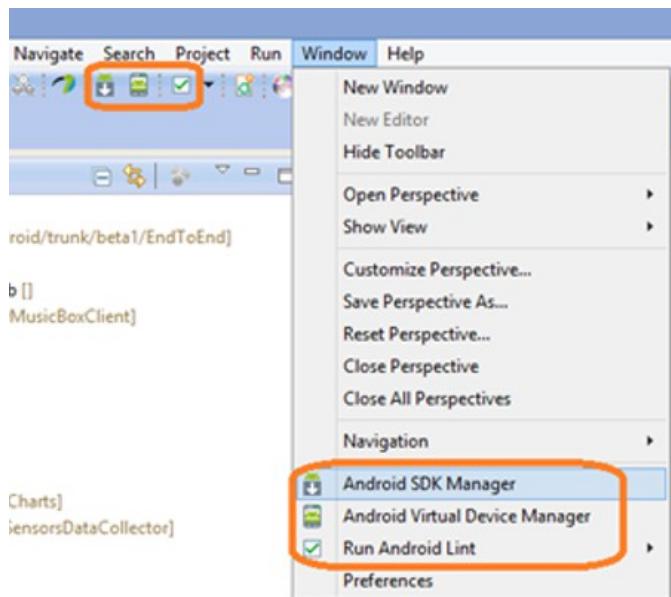


Rysunek 2.2. Eclipse: instalacja nowego oprogramowania



Rysunek 2.3. Eclipse: instalacja wtyczki ADT

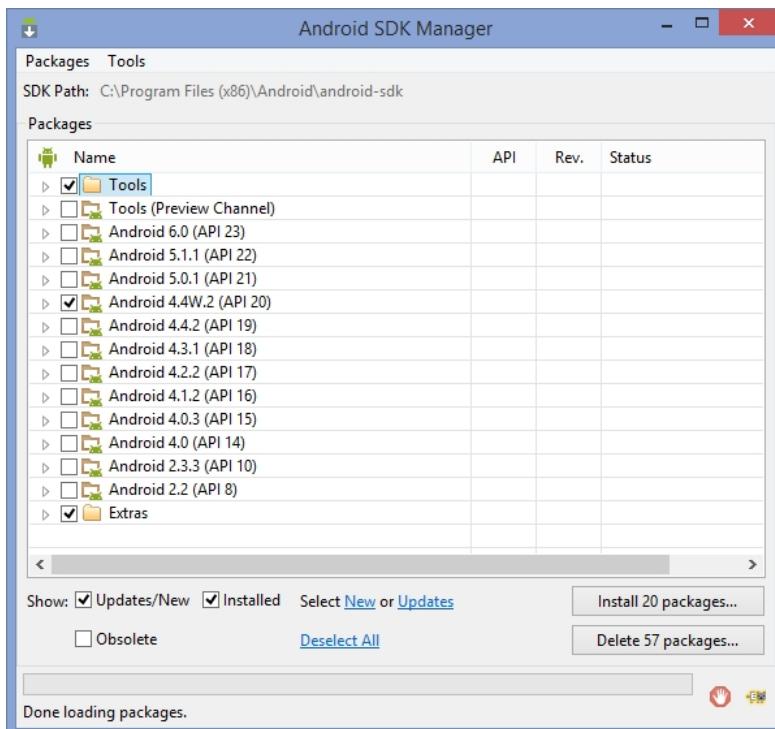
Menedżer Android SDK umożliwia instalację (lub odinstalowanie) dostępnych wersji dla projektów Android (patrz rysunek 2.5): możesz zainstalować co najmniej minimalną wersję, z którą chcesz, aby Twoje projekty były kompatybilne.



Rysunek 2.4. Eclipse: Narzędzia i menu ADT

Istnieje również możliwość instalacji narzędzi i dodatków (API i sterowniki Google). Dokumentacja, kod źródłowy i przykłady projektów znajdują się w folderze instalacyjnym Android SDK (np. dla systemu Windows, zwykle w folderze *Program files*: C:\Program Files (x86)\Android\android-sdk).

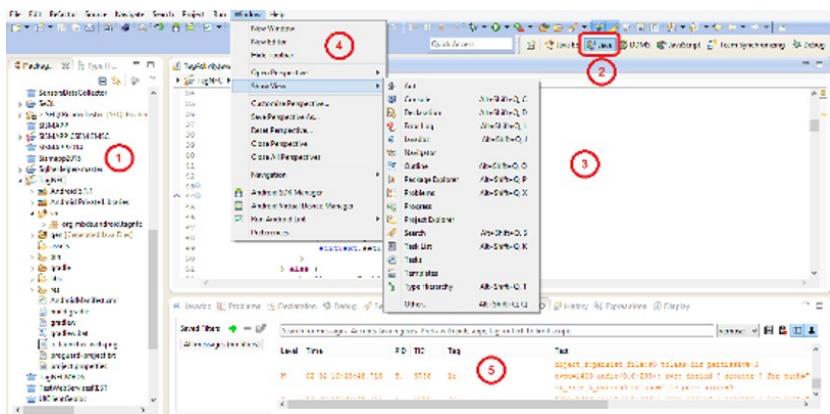
UWAGA - Aktualizacje są regularnie sugerowane; aby wyświetlać tylko aktualizacje lub nowe wersje, wystarczy zaznaczyć pole wyboru *aktualizacji/nowości*, a następnie odznaczyć *Zainstalowane* i *przestarzałe* (domyślne odznaczone). Instalowanie pakietów może zajść dużo czasu i przepustowości, wymaga otwarcia niektórych portów połączenia, które nie zawsze są dozwolone w niektórych sieciach lokalnych. Z tego powodu należy upewnić się, że konfiguracja sieci jest dostosowana do instalacji.



Rysunek 2.5. Eclipse: Menedżer Android SDK

2.1.2.2. Okno środowiska roboczego Eclipse

Podobnie jak większość IDE, główne okno w Eclipse jest kontenerem dla wewnętrznych okien roboczych z paskiem menu i jednym lub kilkoma paskami narzędzi. Wewnętrzne okna mogą być wyświetlane, przenoszone, zmniejszane/powiększane lub ukrywane, ale skupimy się na domyślnym układzie, a dokładniej na perspektywie Java; w rzeczywistości dostępnych jest kilka perspektyw, w oparciu o zainstalowane wtyczki, na przykład J2EE, JavaScript, zarządzanie wersjami i synchronizacja oraz debugowanie. Rysunek 2.6 przedstawia perspektywę Java (2) z eksploratorem pakietów i eksploratorem projektów po lewej stronie (1), oknem kodu źródłowego w środku (3), wyświetlaniem lub ukrywaniem widoków można zarządzać z menu *Windows* (4) i oknem *LogCat* (5) (patrz sekcja 2.1.2.2.1).



Rysunek 2.6. Eclipse: Perspektywa Java

2.1.2.2.1. LogCat

Logcat to konsola wyświetlająca logi telefonu podłączonego do portu USB komputera (wymaga zainstalowania pilota *Android Composite Android Device Bootloader* (ADB), dostarczanego przez Google na urządzeniach Google¹³). Jest to bardzo przydatne źródło informacji wyświetlające logi, które zostały dobrowolnie dodane w programach (z trzema możliwymi trybami: debugowania, informacji lub błędu). Nieobsługiwane wyjątki również pojawiają się w Logcat, co wskazuje na ich pochodzenie w przypadku nieprawidłowego przetwarzania.

UWAGA - Jeśli okno Logcat nie jest wyświetlane domyślnie, można je wyświetlić z menu "Windows|show view|other..." (patrz Rysunek 2.6, (4)).

2.1.2.3. Projekt Android

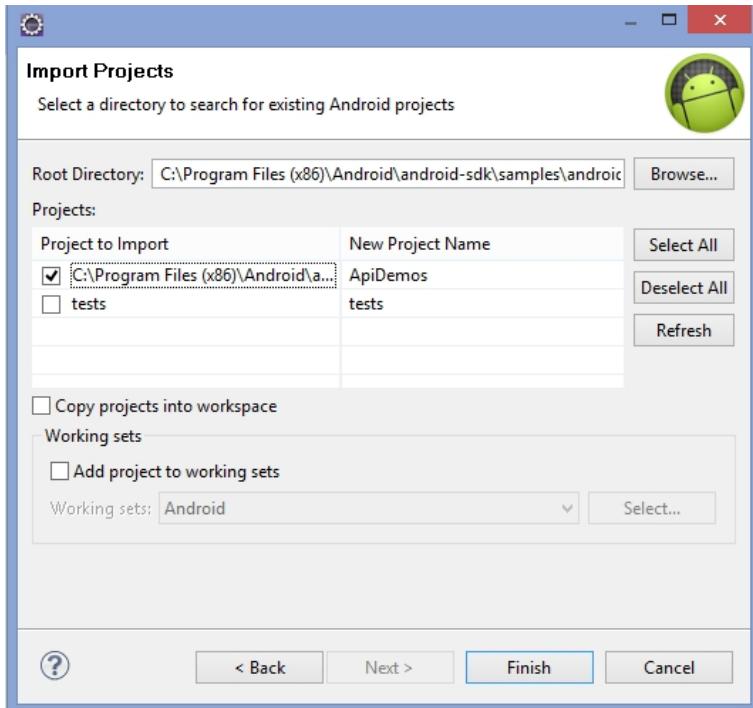
Ta sekcja ma na celu zapoznanie początkujących z projektem Android na przykładzie istniejącego projektu.

2.1.2.3.1. Importowanie istniejącego projektu Android

Z menu *File|New|Project...|Android Project z istniejącego kodu*, w przykładowych projektach dostępnych w Android SDK, my

13 Zob. <http://developer.android.com/sdk/win-usb.html>.

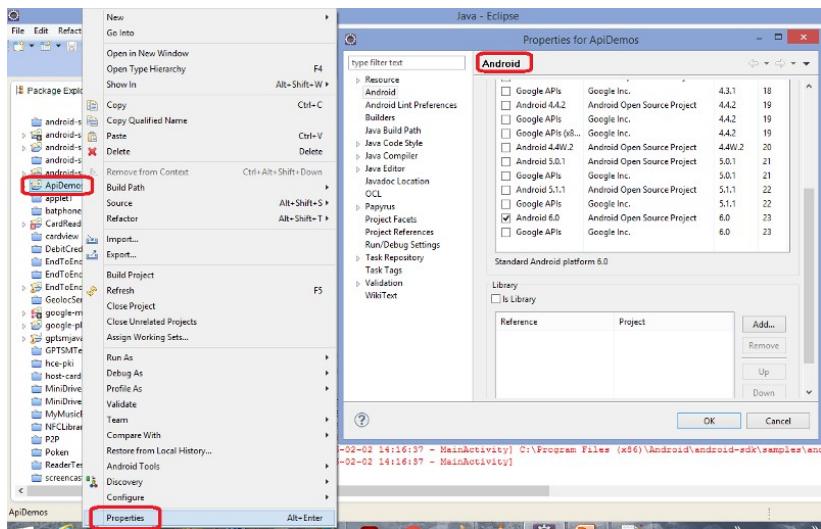
można wybrać (na przykład) projekt *ApiDemos* w C:\Program Files (x86)\Android\android-sdk\samples\android-23\legacy\ApiDemos (tylko dla środowiska Windows) w oknie dialogowym importowania projektu (patrz rysunek 2.7). Opcja *Copy projects into workspace* umożliwia skopiowanie projektu do obszaru roboczego użytkownika (oryginalny projekt źródłowy nie jest modyfikowany).



Rysunek 2.7. Eclipse: importowanie istniejącego projektu Android

2.1.2.3.2. Właściwości projektu Android

Dostęp do właściwości można uzyskać z menu *Project|Properties* lub z menu skrótów (kliknięcie prawym przyciskiem myszy na projekcie): sekcja właściwości *Androïda* (patrz rysunek 2.8) umożliwia wybór SDK, z którym chcemy skompilować aplikację.



Rysunek 2.8. Eclipse: Właściwości projektu Android

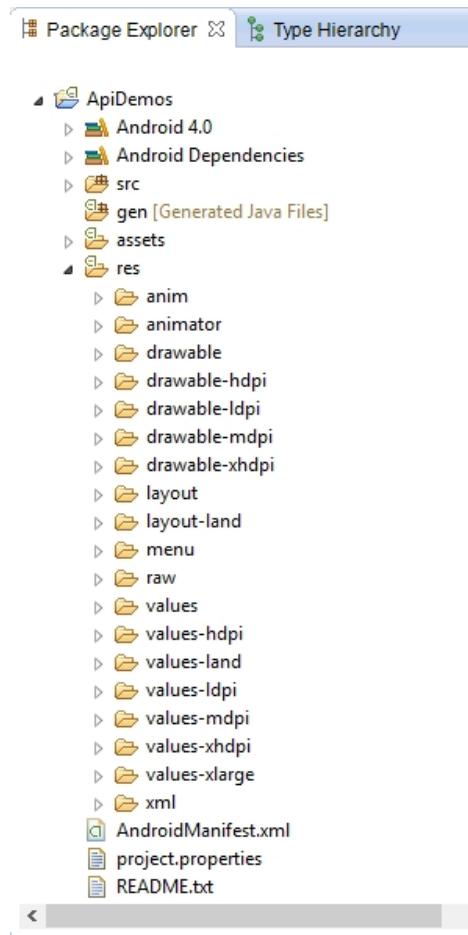
UWAGA - Android SDK powinien być zgodny z deklaracjami złożonymi w pliku Manifest (patrz `AndroidManifest.xml`) i musimy upewnić się, że kodowanie i używane interfejsy API są kompatybilne, na przykład, aby korzystać z interfejsów API Google (Mapy, Google Cloud Messaging itp.). Musimy skompilować z odpowiednią wersją biblioteki Google API.

2.1.2.3.3. Struktura projektu Android

Widok drzewa projektu Android (patrz rysunek 2.9) w Eclipse składa się zazwyczaj z następujących folderów (pod katalogiem głównym projektu):

- pliki archiwum (biblioteki kompatybilności i zależności Androida oraz zewnętrzne biblioteki dodane przez dewelopera);
- folder SRC pakietów i źródeł projektu (rozszerzenie `.java`);
- folder **GEN** z plikami java wygenerowanymi przez ADT podczas komplikacji projektu, na przykład plik `R.java` ("R" dla "Resources"), który zawiera stałe referencyjne wszystkich zasobów projektu (widoki, menu, kontrolki ekranu i przycisków, ciągi znaków, pliki obrazów),

) oraz plik *BuildConfig.java* do zarządzania wyświetlaniem dziennika w trybie debugowania.



Rysunek 2.9. Eclipse: widok drzewa projektu Android

– Folder **ASSETS**: w tym folderze znajdują się wszystkie zewnętrzne zasoby dodane przez dewelopera do wykorzystania przez aplikację (na przykład pliki konfiguracyjne). Po wdrożeniu aplikacji projektu na urządzeniu pliki te są przechowywane w tym samym niezmienionym stanie, co podczas komplikacji (tylko do odczytu), ale mogą być kopiowane przez aplikację.

w telefonie w obszarze pamięci (a następnie w razie potrzeby zmodyfikować).

– Folder **BIN** plików binarnych wygenerowanych podczas komplikacji. Zawiera on archiwum APK aplikacji, która zostanie wdrożona na urządzeniach z systemem Android.

– Folder **LIBS** zewnętrznych bibliotek (dodanych i zarządzanych przez dewelopera).

– Folder **RES** z plikami zasobów: opisy interfejsów użytkownika (w podfolderze **layout**), podfolder **values** zawiera style, ciągi znaków (w tym tłumaczenia), **menu** można znaleźć w podfolderze menus itp. Wszystkie te zasoby są opisane w plikach *XML*. Dodatkowe opisy znajdują się w podfolderze **xml**, a pliki graficzne (np. ikony aplikacji) w podfolderze **drawable-xxx**, natomiast pliki multimedialne w podfolderze **raw**. Podfoldery projektu Android można podzielić zgodnie z docelową rozdzielczością i formatem układu lub zgodnie z wersją Androida (wskazaną przez rozszerzenie w nazwie podfolderu).

2.1.2.3.4. Plik *AndroidManifest.xml*

W projekcie Android plik deklaratywny aplikacji (manifest) znajduje się w katalogu głównym projektu (patrz rysunek 2.9): jest to plik *AndroidManifest.xml*, którego zawartość w formacie XML została zilustrowana na rysunku 2.10. Informacje o manifestie można również przeglądać według tematu w interfejsie Eclipse (patrz rysunek 2.11), a także w zakładkach *manifest*, *aplikacja*, *uprawnienia* i *oprzyrządowanie*. W sekcji manifestu zostaną wprowadzone ogólne atrybuty, takie jak nazwa pakietu aplikacji, wersja i nazwa tej wersji, zgodność numeru Android SDK i funkcje, do których aplikacja musi uzyskać dostęp (oprogramowanie lub sprzęt), sekcja z **uprawnieniami**, które użytkownik musi wyrazić zgodę na przyznanie aplikacji podczas instalacji; sekcja **instrumentacji** umożliwia łączenie klas rozszerzających nadklasę *android.app.Instrumentation* używaną na przykład do testów automatycznych, podczas gdy sekcja **aplikacji** zawiera atrybuty aplikacji i deklaracje działań (w tym filtry i metadane, na przykład w celu zadeklarowania głównego działania programu uruchamiającego aplikację lub działań filtrowanych po wykryciu technologii NFC),

a także deklarację niektórych interfejsów API z kluczami aktywacyjnymi lub bez, na przykład interfejsów API Google (monitorowanych z konsoli: Mapy, Google Play, Street view, Google Cloud Messaging, Prediction, YouTube itp.).

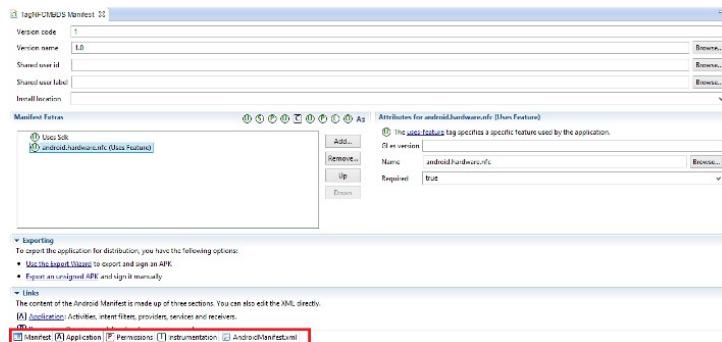
```

1 <?xml version="1.0" encoding="utf-8"?>
2<manifest xmlns:android="http://schemas.android.com/apk/res/android"
3   package="org.mbdz.android.tagnfc"
4   android:versionCode="1"
5   android:versionName="1.0" >
6   <uses-sdk android:minSdkVersion="14" android:targetSdkVersion="18" android:targetSdkVersion="18"/>
7   <uses-feature android:name="android.hardware.nfc" android:required="true" />
8   <uses-permission android:name="android.permission.NFC" />
9</application>
10  <application android:allowBackup="true">
11    android:icon="@drawable/ic_launcher"
12    android:label="@string/app_name"
13    android:theme="@style/AppTheme" >
14      <activity android:name="org.mbdz.android.tagnfc.TagActivity"
15        android:label="@string/app_name" >
16        <intent-filter>
17          <action android:name="android.intent.action.MAIN" />
18          <category android:name="android.intent.category.LAUNCHER" />
19        </intent-filter>
20      </activity>
21      <activity android:name="org.mbdz.android.tagnfc.NFCReader"
22        android:launchMode="singleTop"
23        android:label="@string/app_name" >
24        <intent-filter>
25          <action android:name="android.nfc.action.NDEF_DISCOVERED" />
26          <category android:name="android.intent.category.DEFAULT" />
27          <data android:host="www.mbdz-fr.org" android:scheme="http" />
28        </intent-filter>
29      </activity>
30    </application>

```

Rysunek 2.10. Eclipse: Plik AndroidManifest.xml

UWAGA: - Aby dowiedzieć się więcej na temat korzystania z pliku AndroidManifest, odwiedź witrynę dla programistów Androïda pod adresem URL: <http://developer.android.com/guide/topics/manifest/manifest-intro.html>.



Rysunek 2.11. Eclipse: perspektywy pliku AndroidManifest.xml

2.1.2.3.5. Filtruje zamiary dotyczące działań i usług

Filtry zdefiniowane w sekcji tagów *aktywności* w pliku Manifest (podkategoria *akcji* tagów) mają na celu zapewnienie zamierzonego wykorzystania danej aktywności.

Istnieją trzy rodzaje filtrów intencji:

- filtr **akcji** wskazuje typ akcji wykonywanej przez aktywność, na przykład dla głównej aktywności i akcji *android.intent.action.MAIN* lub filtr wykrywania technologii NFC (patrz sekcja 2.2.1);
- filtr **kategorii** wskazuje kategorię aktywności, na przykład aplikacja zostanie uruchomiona z aktywnością zadeklarowaną w manifeście z filtrem *android.intent.category.LAUNCHER*;
- filtr **danych** wskazuje typ danych, których oczekuje działanie (tj. działanie zostanie wybudzone tylko po wykryciu tego formatu): nazwany format, adres URL, domena itp. Na przykład filtr danych jest używany do kierowania określonego typu zawartości tagów NFC; domena, adres URL itp.

2.1.3. Intencje i kontekst Android

W systemach Android aplikacje działają w podzielonych środowiskach, dzięki czemu nie wpływają na zachowanie innych wykonań. Intencje są częścią mechanizmu wymiany między zasobami aplikacji w celu poinformowania systemu o zamiarze uzyskania dostępu do zasobu. Klasa Intent opisuje akcję lub zdarzenie, w tym kontekst, komponent, dodatki i flagi opcji. Intencje umożliwiają również wysyłanie *Bundle* (w którym parametry lub obiekty są spakowane parami klucz-wartość) pomiędzy instancjami aktywności i ich pochodnymi, na przykład usługą działającą w tle (tj. procesem demona) lub *Broadcast Receiver* nasłuchującym wiadomości wysyłanych przez system i do których subskrybuje.

Obiekt *Context systemu* Android umożliwia dostęp do informacji i stanu aplikacji, ale umożliwia także dostęp do zasobów dostępnych w kontekście aplikacji, ponieważ aplikacja jest kontekstem samym w sobie, który implementuje interfejs *Context systemu* Android.

2.1.4. Klasa aktywności Android

Klasy dziedziczące z nadklasy Activity są klasami Androïda przeznaczonymi do zarządzania interakcjami użytkownika. Aktywność umożliwia zarządzanie interfejsem graficznym (ekranem): aktywność jest zwykle powiązana z widokiem (opisanym w XML w plikach *układu* znajdujących się w folderze *res*) w celu zarządzania działaniami użytkownika na opcjach menu, przyciskach, wpisach na klawiaturze, zdarzeniach dotykowych na ekranie lub nawet głosie (*Google Talk*).

UWAGA - Aplikacja Android z interfejsem użytkownika (ekranem) składa się z co najmniej jednej aktywności; w tym przypadku możemy założyć, że jest to główna aktywność (filtr akcji "MAIN") uruchamiana automatycznie po dotknięciu ikony aplikacji przez użytkownika.

2.1.4.1. Cykl życia aktywności w systemie Android

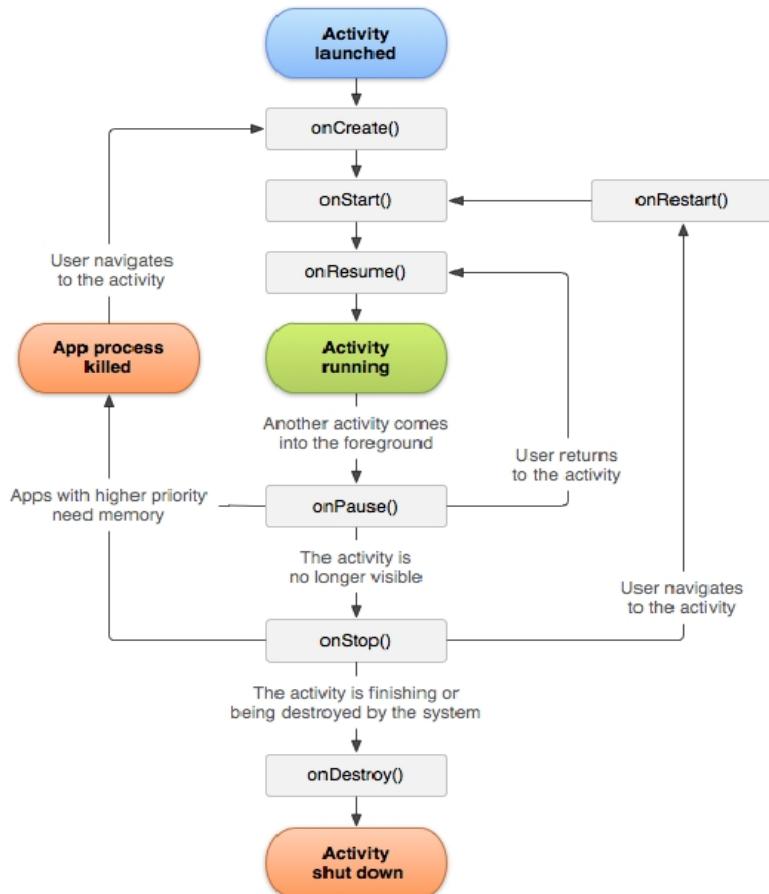
Jednym z atutów rozwoju Androïda jest zwolnienie dewelopera z zarządzania zasobami pamięci, zwłaszcza poprzez implementację klasy *Activity*.

Wzorzec cyklu życia jest przedstawiony na rysunku 2.12. pokazującym metody odziedziczone po nadklasie, które są wywoływanie w każdym stanie cyklu życia aktywności: metody te mogą być nadpisywane (uwaga *@override*) w celu wykonania kodu zaimplementowanego dla tego przejścia stanu.

UWAGA - Gdy aktywność nie jest już na pierwszym planie (użytkownik uruchamia inną aktywność), jej stan (wstrzymanie lub zatrzymanie) może się różnić w zależności od flag przekazanych jako parametry podczas uruchamiania aktywności (patrz sekcja 2.1.4.2). Należy to uwzględnić w sposobie kodowania inicjalizacji w metodach wywołania *zwrotnego*¹⁴ cyklu życia aktywności, aby nie uruchamiać tej samej aktywności kilka razy, gdy jedna (lub kilka) jest już wstrzymana. W ten sam sposób, gdy aktywność jest uruchamiana (lub wznowiana), niekoniecznie przechodzi przez wywołania zwrotne *onCreate* lub *onStart*, ale zawsze przechodzi przez metodę *onResume*:

¹⁴ Funkcja wywołania zwrotnego po wystąpieniu zdarzenia.

należy uważać, aby prawidłowo umieścić kod inicjalizacji i zwalniania obiektów we właściwej metodzie.



Rysunek 2.12. Android: cykl życia aktywności

2.1.4.2. Zarządzanie działalnością

Jak pokazuje kod na rysunku 2.13, metoda `startActivity` uruchamia i umieszcza aktywność na pierwszym planie, przyjmuje intencję jako parametr (patrz sekcja 2.1.3), a także klasę aktywności do aktywacji.

```
Intent intent = new Intent(this, TestActivity.class);
startActivity(intent);
```

Rysunek 2.13. Android: uruchamianie aktywności

UWAGA - W zależności od *intencji* i przekazanych parametrów, metoda *startActivity* odziedziczona z kontekstu Androïd (patrz sekcja 2.1.3) jest również używana do uruchamiania działań, takich jak wysyłanie wiadomości e-mail, wyświetlanie adresu URL w przeglądarce.

Gdy aktywność jest wywoływana z oczekiwany wynikiem w zamian, użycie metody *startActivityForResult* spowoduje *wywołanie zwrotne* metody *onActivityResult* aktywności wywołującej. Następnie pochodzenie żądania można zidentyfikować za pomocą kodu zwróconego przez wywołaną aktywność połączoną z intencją danych (wartość null, jeśli jest bezużyteczna). Wystarczy nadpisać metodę wywołania zwrotnego, aby sprawdzić pochodzenie żądania i zwrócony wynik w celu przeprowadzenia właściwego przetwarzania. Na przykład kod przedstawiony na rysunku 2.14 pokazuje, jak przekonwertować tekst na wiadomość głosową i jak uruchomić działanie rozpoznawania głosu za pomocą *TextToSpeech*, a następnie zebrać tekst w tablicy ciągów za pomocą metody *onActivityResult*.

```
// Init language (locale)
if (tts.isLanguageAvailable(locale) == TextToSpeech.LANG_COUNTRY_AVAILABLE) {
    tts.setLanguage(locale);
}
// Convert text message into voice
boolean res = tts.speak(message, TextToSpeech.QUEUE_FLUSH, null)==0;

}

// When click on the speech icon
public void startVoiceRecognition() {
    Intent intent = new Intent(RecognizerIntent.ACTION_RECOGNIZE_SPEECH);
    intent.putExtra(RecognizerIntent.EXTRA_LANGUAGE_MODEL, RecognizerIntent.LANGUAGE_MODEL_FREE_FORM);
    startActivityForResult(intent, REQUEST_CODE);
}

// The result is sent to the activity
@Override
protected void onActivityResult(int requestCode,
        int resultCode, Intent data) {
    if (requestCode == REQUEST_CODE && resultCode == RESULT_OK) {
        // Get the speech to text result
        ArrayList<String> speech = data.getStringArrayListExtra(RecognizerIntent.EXTRA_RESULTS);
        // Do something with speech
    }
    super.onActivityResult(requestCode, resultCode, data);
}
```

Rysunek 2.14. Android: rozpoznawanie głosu (*TextToSpeech*)

Aby zakończyć aktywność, która na przykład została wywołana przez akcję użytkownika, wystarczy wywołać metodę *finish()*. W tym

Jeśli inną aktywność znajduje się na szczycie stosu wywołań aplikacji, zastąpi ona zakończoną aktywność na pierwszym planie.

2.1.5. Interfejs graficzny systemu Android: "layout" pliki

Interfejsy użytkownika (widoki) są opisane w plikach XML, które definiują pozycjonowanie elementów sterujących (przycisków, obrazów, pól tekstowych itp.) oraz sposób ich rozmieszczenia (poniżej, po prawej, po lewej itp.) w kontenerze liniowym, w pozycji bezwzględnej, w siatce, w poziomie, w pionie itp.

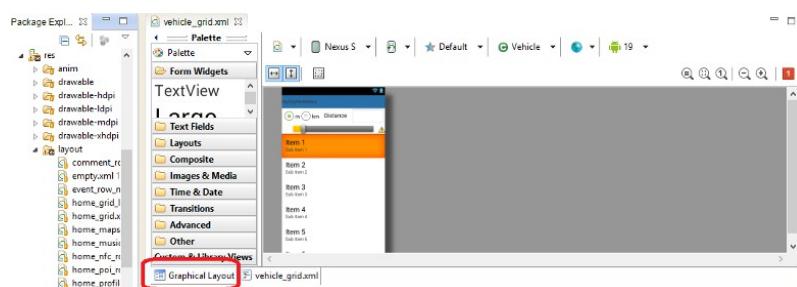
UWAGA - Jak pokazano na rysunku 2.15, orientacją ekranu (zgodnie z pozycją wykrytą przez czujniki inercyjne: pionową lub poziomą) można zarządzać na poziomie deklaracji aktywności w pliku manifestu.

```
<activity
    android:name=".android.activity.Musics"
    android:label="Musics"
    android:clearTaskOnLaunch="true"
    android:screenOrientation="fullSensor"
    android:theme="@style/Theme.Default" >
</activity>
```

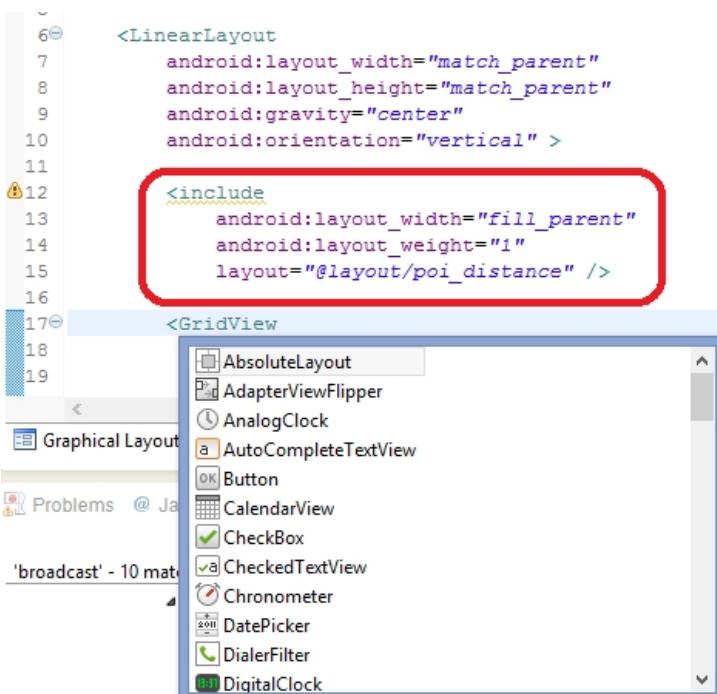
Rysunek 2.15. Android: orientacja aktywności na ekranie

2.1.5.1. Projektowanie układu

W Eclipse dostępne są dwa tryby projektowania: projektant wizualny (zakładka "Układ graficzny"), który umożliwia przeciąganie i upuszczanie komponentów oraz graficzne umieszczanie ich w widoku (patrz rysunek 2.16), oraz widok XML (patrz rysunek 2.17).



Rysunek 2.16. Android: graficzne tworzenie układu



Rysunek 2.17. Android: włączenie układu i kreator kodowania

UWAGA - Układ może odwoływać się do innego układu zdefiniowanego w innym pliku XML; pozwala to na ponowne wykorzystanie atomowych fragmentów widoków. Aby to zrobić, używamy znacznika *include* odwołującego się do pliku *układu*, który ma zostać dołączony (patrz rysunek 2.17).

Rysunek 2.17 pokazuje również menu kreatora kodowania w Eclipse, które jest aktywowane poprzez umieszczenie kurSORA w odpowiednim miejscu i naciśnięcie kombinacji klawiszy "Ctrl "+"spacja"; następnie wystarczy wybrać element, który chcesz wstawić z listy dostępnych kodów, przewijając (za pomocą klawiszy strzałek) i nacisnąć "enter", aby potwierdzić.

2.1.5.2. Powiązanie układu z aktywnością i obsługa kontrolek

Layout może być "ustawiony" (metoda "setContentView") do jednej lub kilku aktywności (zazwyczaj w metodzie "onCreate"). Z tego u k ł a d u ,

elementy sterujące mogą być wyświetlane w różny sposób, aby dostosować docelowe urządzenie niezależnie od typu wyświetlacza i niezależnie od opublikowanej wersji aplikacji na Androide; na przykład może być zainstalowany na smartfonie, tablecie, inteligentnym zegarku, a także na inteligentnym piekarniku i dowolnym innym inteligentnym obiekcie działającym pod kontrolą systemu operacyjnego Android z wyświetlaczem.

UWAGA - *Fragmenty*¹⁵, wprowadzone od wersji 3.0 Honeycomb i Android API 11, ułatwiają modułowe zarządzanie widokami i wyświetlanie ich na ekranach o różnych rozmiarach i kształtach. W ten sposób fragment jest kawałkiem kodu odpowiedzialnym za kontrolowanie widoku. Ma on swój własny cykl życia i może być dynamicznie dodawany lub usuwany z aktywności na podstawie zdarzeń lub typu wyświetlacza wykrytego podczas tworzenia aktywności (metoda *onCreate*).

2.1.5.3. Obsługa działań użytkownika

Działania użytkownika są obsługiwane w aktywności i/lub w układzie i/lub na poziomie kontrolki za pomocą dziedziczonych metod wywołania zwrotnego lub poprzez implementację interfejsu lub obiektu wywołania zwrotnego. Na przykład *onClickListener*, *onTouchListener*, *onScrollListener* itp. Kod źródłowy przedstawiony na rysunku 2.18 przedstawia przykład, w jaki sposób zdarzenia dotykowe są wyzwalane i przetwarzane po przesunięciu palcem po ekranie poprzez zarządzanie współrzędnymi *x* i *y* o d p o w i a d a j ą c y m i lewemu górnemu i prawemu dolnemu rogowi wirtualnego prostokąta ukształtowanego przez położenie palców w stosunku do punktu orientacyjnego ekranu na początku ruchu (*ACTION_DOWN*), podczas ruchu (*ACTION_POINTER_DOWN*) i po zakończeniu ruchu, gdy użytkownik podnosi palce z ekranu (*ACTION_MOVE*).

W ten sam sposób, gdy użytkownik naciśnie klawisze "powrotu" i "domu" systemu, zdarzenie jest wysyłane do wywołania zwrotnego *onKeyDown*, które można zastąpić w aktywności, jak pokazano w kodzie na rysunku 2.19.

15 Zob. <http://www.tutorialspoint.com/android/android.fragments.htm>.

```
1 // Screen touch is handled by
2 // the onTouch listener
3 myView.setOnTouchListener(this);
4 // An activity that implement onTouchListener
5 // should implement the override onTouchEvent method
6 @Override
7 public boolean onTouchEvent(MotionEvent event) {
8     // x, y are the current position while moving
9     float y = event.getY();
10    float x = event.getX();
11    switch (event.getAction() & MotionEvent.ACTION_MASK) {
12        case MotionEvent.ACTION_DOWN:
13            // Do something: Record initial position?
14            break;
15        case MotionEvent.ACTION_POINTER_DOWN:
16            // Do something: Estimate the spacing from initial position?
17            break;
18        case MotionEvent.ACTION_MOVE:
19            // do something: Translate according the final spacing?
20            break;
21    }
22    return super.onTouchEvent(event);
23 }
```

Rysunek 2.18. Android: zarządzanie ekranem dotykowym

```
1 @Override
2 public boolean onKeyDown(int keyCode, KeyEvent event) {
3     if (keyCode == KeyEvent.KEYCODE_BACK) {
4         // Do something
5     } else if (keyCode == KeyEvent.KEYCODE_HOME) {
6         // Do something else...
7     }
8     // etc..
9
10 }
```

Rysunek 2.19. Android: orientacja aktywności na ekranie

Akcje w menu mogą być również przechwytywane (lub przetwarzane), na przykład poprzez nadpisanie metody *onMenuItemSelected*.

2.1.6. Kompilowanie i testowanie aplikacji Android

Domyślnie w Eclipse IDE komplikacje projektów Android wykonywane są automatycznie. Czasami może to być źródłem kłopotów podczas kodowania (komunikaty o błędach, powolność z powodu komplikacji).

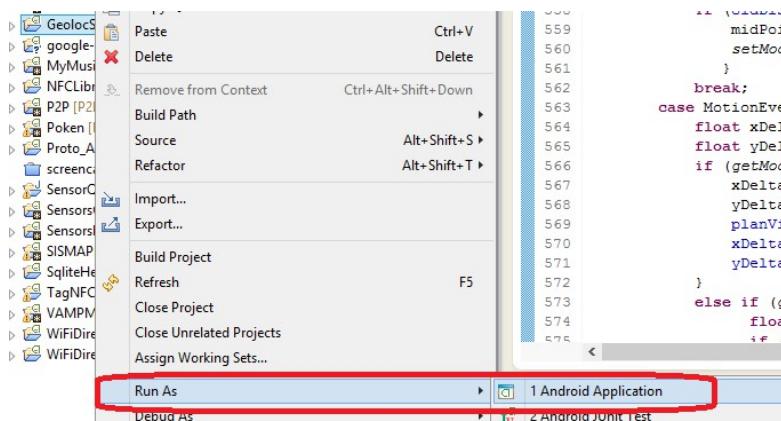
w tle). Opcję tę można wyłączyć w menu *Project*, odznaczając opcję automatycznej komplikacji "Build automatically".

Opcja "Build project" jest wtedy dostępna w menu skrótów projektu (kliknij prawym przyciskiem myszy na katalogu głównym projektu w eksploratorze pakietów).

Czasami projekt może zawierać błędy, których możesz nie rozumieć: może to być spowodowane wcześniej wygenerowanymi obiekttami binarnymi, które nie są już kompatybilne z ostatnią wersją kodu źródłowego (np. usunięcie układów). W takim przypadku należy "wyczyścić" wygenerowane obiekty binarne z menu *Project|Clean* (pamiętaj, by robić to za każdym razem, gdy wystąpi tego typu problem!).

2.1.6.1. Uruchamianie aplikacji

Gdy aplikacja jest gotowa do testowania (brak błędów komplikacji), można ją uruchomić z menu *Run|Run.as|Android Application*, które jest również dostępne w menu skrótów projektu (patrz rysunek 2.20).



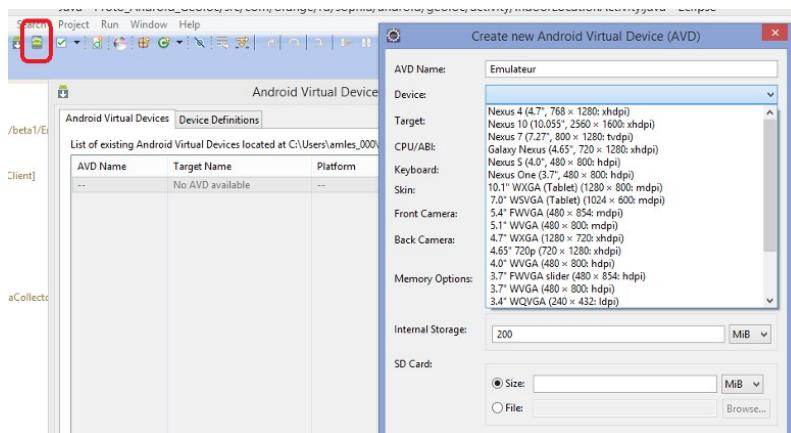
Rysunek 2.20. *Android: uruchamianie aplikacji z menu kontekstowego*

Z poziomu projektu Eclipse dostępne są dwie opcje uruchamiania aplikacji Android: ADT udostępnia konfigurowalne emulatory do stworzenia urządzenia z Androidem, ale jeśli mamy urządzenie z Androidem, lepiej z niego skorzystać

aby uruchomić nasz projekt aplikacji bezpośrednio na naszym urządzeniu z Androidem, ponieważ emulator wymaga przydzielonych mu zasobów, a to może zwiększyć obciążenie naszego systemu i spowodować bardzo, a nawet bardzo powolne wyświetlanie.

2.1.6.2. Korzystanie z emulatora urządzenia Android

Menu *Window|Android Virtual Device (AVD) manager* oraz powiązane narzędzie przedstawione na rysunku 2.21 umożliwiają zarządzanie emulatorem urządzenia Android: opcja *New* wyświetla ekran konfiguracji nowego emulatora.



Rysunek 2.21. *Android: Android Virtual Device (AVD)*

UWAGA: - Możemy zauważyć, że aplikacje NFC nie są łatwe do przetestowania przy użyciu urządzenia wirtualnego: w rzeczywistości konieczne będzie również zainstalowanie emulatora czytnika NFC, a także emulatora tagu NFC lub karty zbliżeniowej¹⁶, co jest bardzo kłopotliwe w konfiguracji, ale nie zapewni to takiego samego zachowania jak na urządzeniu z Androidem obsługującym NFC. Dlatego zdecydowanie zalecamy testowanie aplikacji na WSZYSTKICH typach urządzeń, na których chcesz wdrożyć swoją aplikację.

16 Zobacz narzędzia "Android Edition" dostarczone przez Open NFC™: <http://open-nfc.org/wp/ editions/android>.

2.1.6.3. Korzystanie z urządzenia z systemem Android podłączonego do portu USB

Komponent ADT ADB wykrywa urządzenie z systemem Android podłączone do portu USB i umożliwia uruchomienie aplikacji bezpośrednio na podłączonym urządzeniu.

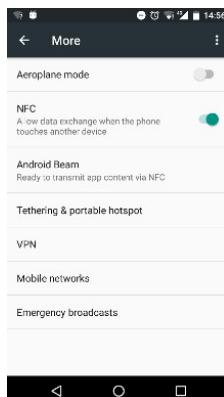
UWAGA - Będziemy musieli upewnić się, że zainstalowana wersja systemu operacyjnego Android i funkcje urządzenia, na którym uruchamiamy aplikację, są zgodne z tym, co zostało zadeklarowane w pliku *AndroidManifest*.

Co więcej, aby urządzenie z Androidem zostało wykryte przez ADB, potrzebujemy odpowiedniego sterownika ADB w zależności od modelu urządzenia z Androidem, które podłączamy do systemu.

UWAGA - Sterowniki Google są zgodne z szeroką gamą urządzeń i można je zainstalować ręcznie z folderu *extras* znajdującego się w katalogu Android SDK (na przykład w zwykłych środowiskach Windows można go znaleźć pod adresem C:\Program Files (x86)\Android\android-sdk\extras\google\usb_driver).

2.2. Wdrażanie NFC za pomocą Android

Pierwszym warunkiem uzyskania dostępu do funkcji NFC jest włączenie opcji NFC w parametrach łączności systemu Android.



Rysunek 2.22. Android: włączenie funkcji NFC w parametrach systemu Android

2.2.1. *Android manifest deklaracje*

Należy zadeklarować użycie funkcji sprzętowej NFC. Atrybut *required* definiuje wymagania warunkowe. Gdy jest ustawiony na *true*, umożliwia sugerowanie aplikacji tylko urządzeniom mobilnym obsługującym NFC w Google Play (co oznacza, że NFC jest obowiązkowe, aby móc zainstalować i używać aplikacji).

```
<uses-feature android:name="android.hardware.nfc" android:required="true"/>
```

Box 2.1. *Android: deklarowanie funkcji NFC*

Poza tym, dostępność funkcji NFC może być również testowana w kodzie.

```
PackageManager manager = this.getPackageManager();
// Sprawdź, czy NFC jest dostępne na urządzeniu
if (!manager.hasSystemFeature(PackageManager.FEATURE_NFC))
{
    //informuje użytkownika, że funkcja NFC jest niedostępna
}
```

Box 2.2. *Android: testowanie dostępności NFC*

Korzystanie z NFC wymaga również deklaracji zgody na korzystanie, o której użytkownik zostanie powiadomiony po zainstalowaniu aplikacji; jeśli użytkownik odmówi udzielenia zgody na korzystanie z NFC, aplikacja nie zostanie zainstalowana.

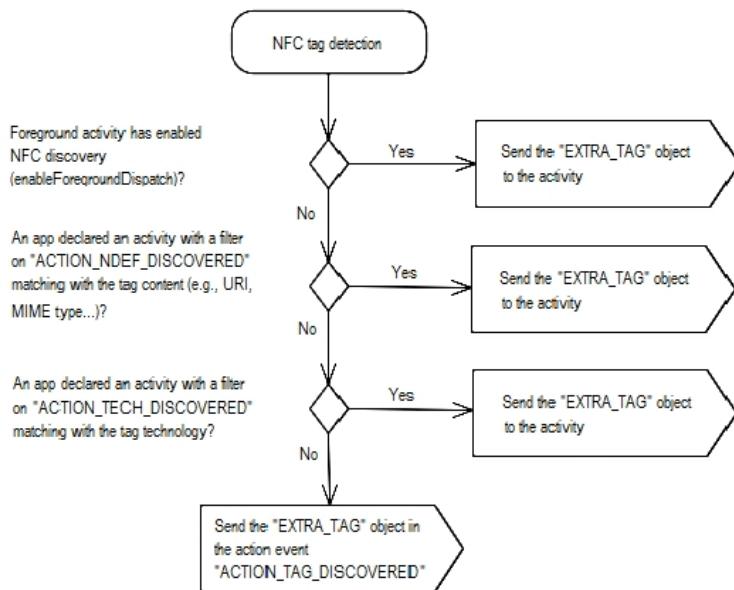
```
<uses-permission android:name="android.permission.NFC"/>
```

Box 2.3. *Android: Deklaracja uprawnień NFC*

2.2.2. *Wdrożenie trybu czytnika/zapisu NFC*

Ta sekcja wprowadza czytnik do odczytu i zapisu pasywnego tagu NFC.

UWAGA: - Technologia Mifare (firmy NXP) i bezstykowe karty inteligentne to specyficzne typy tagów, które nie są oparte na NDEF, ale na protokole opartym na APDU wspomnianym w sekcji dotyczącej trybu emulacji karty NFC (patrz sekcja 2.2.4).



Rysunek 2.23. *Android: wykrywanie zarządzania urządzeniem NFC*

W manifeście, intencja¹⁷ filtrowania zdarzeń NFC musi być zadeklarowana na poziomie aktywności (w sekcji aplikacji) implementacji.

przetwarzanie NFC wykrywanie urządzeń (na tag);
przykład

Rysunek prioryt wykrywanie
2.23 z którym et

Technologie NFC są obsługiwane wydarzeń jest do
przed rozpoczęciem aktywności. ie uruchamian
y

Po pierwsze, system Android sprawdza, czy bieżąca aktywność (na pierwszym planie) zasubskrybowała filtr wykrywania NFC (dynamicznie zadeklarowany w aktywności lub statycznie w manifeście) dla pasującego

¹⁷ Słowo to definiuje "jednostkę" zawierającą akcję do wykonania, na przykład aktywność.

na typie zawartości (NDEF), na technologii (zgodnie z listą technologii NFC) lub na wykryciu tagu; w takim przypadku intencja reprezentująca tag (dodatkowy tag) jest dostarczana do tej metody wywołania zwrotnego *onNewIntent*.

W przykładzie podanym w ramce 2.4, aktywność *TagActivity* deklaruje filtr do wykrywania tagów NFC i filtr do wykrywania technologii NFC, które są wymienione przez dewelopera w pliku w formacie XML znajdującym się w folderze *res/xml*, o którym mowa w atrybutie "android:resource" sekcji "meta-data" w manifeście Androïda.

UWAGA: - Należy zauważyc, że w przykładzie podanym w ramce 2.4 uwzględniono tylko pierwszy filtr, drugi filtr dotyczący wykrywania technologii NFC podano tutaj jako przykład, podobnie jak powiązany z nim znacznik "metadanych".

```
<aktywność
    android:name=".TagActivity"
    android:launchMode="singleTop"
    ...>
    <intent-filter>
        <action android:name="android.nfc.action.TAG_DISCOVERED"/>
        <category android:name="android.intent.category.DEFAULT"/>
    </filtr treści>
    <intent-filter>
        <action android:name="android.nfc.action.TECH_DISCOVERED"/>
        <category android:name="android.intent.category.DEFAULT"/>
    </filtr treści>
    <meta-data android:name="android.nfc.action.TECH_DISCOVERED"
        android:resource="@xml/nfc_tech_filter"/>
    ...
</aktywność
    <!--
```

Ramka 2.4. Android: filtry na przykładzie deklaracji wykrywania NFC

UWAGA: - Tryb uruchamiania *SingleTop* oznacza, że aktywność zostanie uruchomiona tylko raz w stosie wywołań aktywności (na przykład za każdym razem, gdy zostanie wykryte urządzenie NFC, jeśli aktywność jest już w stosie, zostanie umieszczona na pierwszym planie zamiast uruchamiania dwa razy).

Ramka 2.5 to przykład deklaracji listy technologii NFC obsługiwanych przez aplikację w pliku XML (nazwanym w ten sam sposób, w jaki został zadeklarowany w manifeście, na przykład w ramce 2.4: "nfc_tech_filter.xml"):

```
<resources xmlns:xliff="urn:oasis:names:tc:xliff:document:1.2">
<tech-list>
<tech>android.nfc.tech.IsoDep</tech>
<tech>android.nfc.tech.NfcA</tech>
<tech>android.nfc.tech.NfcB</tech>
<tech>android.nfc.tech.NfcF</tech>
<tech>android.nfc.tech.NfcV</tech>
<tech>android.nfc.tech.Ndef</tech>
<tech>android.nfc.tech.NdefFormattable</tech>
<tech>android.nfc.tech.MifareClassic</tech>
<tech>android.nfc.tech.MifareUltralight</tech>
</tech-list>
</resources>
```

Ramka 2.5. Android: lista przykładowych technologii NFC

Dwa proste rozwiązania umożliwiają filtrowanie wykrywania tagów na określonej treści (tj. dla określonej aplikacji):

- odwołując się do domeny (format URL);
- odwołując się do aplikacji.

Filtr w domenie:

Ramka 2.6 pokazuje, jak utworzyć filtr w domenie. Przykład umieszczony jako komentarz pokazuje, jak używać adresu IP do potencjalnych faz testowania przed wdrożeniem nazwy domeny.

UWAGA: - Ta domena może być równie dobrze dowolną domeną wybraną do identyfikacji tagów, które uruchomią aplikację NFC: nie ma kontroli nad ważnością domeny. Tylko tagi, których treść zaczyna się od nazwy domeny wprowadzonej w filtrze, aktywują działanie NFC bez potrzeby działania ze strony użytkownika, z wyjątkiem zbliżenia urządzeń NFC.

```
<intent-filter>
    <action android:name="android.nfc.action.NDEF_DISCOVERED"/>
    <category android:name="android.intent.category.DEFAULT"/>
    <data android:host="www.mbdz-fr.org"
        android:pathPrefix="/application/"
        android:scheme="http"/>
    <!-- Kolejny przykład... -->
    <!-- android:host="134.59.152.102" -->
    <!-- android:scheme="http" -->
    <!-- android:port="8080"/>
</filtr treści>
```

Box 2.6. Android: Filtr wykrywania NFC na przykładzie zawartości NDEF tagów NFC

Filtr NFC w aplikacji:

Aby aktywować działanie, zawartość tagu musi być zakodowana tym samym typem MIME (lub typem mediów¹⁸), co typ zadeklarowany w manifeście (patrz sekcja 2.2.2.1), na przykład poprzez określenie pakietu aplikacji (patrz Ramka 2.7). Spowoduje to automatyczne uruchomienie odpowiedniej aplikacji podezas odczytu tagu, bez interwencji użytkownika.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="org.mbdz.android.tagnfc"
    android:versionCode="1"
    android:versionName="1.0">
    ...
    <activity
        android:name=".TagActivity"
        android:launchMode="singleTop"
        ...>
        <intent-filter>
            <action android:name="android.nfc.action.NDEF_DISCOVERED"/>
            <category android:name="android.intent.category.DEFAULT"/>
            <data android:mimeType="application/org.mbdz.android.tagnfc"/>
        </filtr treści>
    </aktywność>
    ...

```

Ramka 2.7. Android: przykład filtru wykrywania NFC na MIME typu NDEF

18 Zobacz Wikipedię: https://en.wikipedia.org/wiki/Media_type.

2.2.2.1. Odczytywanie tagu NFC

W aktywności zadeklarowanej w manifeście do wykrywania NFC (na podstawie filtra) instancja adaptera NFC, która zapewnia proste metody łączenia z kontrolerem NFC, jest inicjowana podczas tworzenia aktywności.

```
NfcAdapter nfcAdapter = NfcAdapter.getDefaultAdapter(this);
if (!nfcAdapter.isEnabled())
{
    //Poproś użytkownika o aktywację opcji NFC
}
```

Box 2.8. Android: inicjalizacja adaptera NFC

UWAGA - Należy zauważyć, że jeśli nie jest zainstalowany żaden kontroler NFC (w przypadku urządzenia peryferyjnego, które nie obsługuje NFC), instancja adaptera NFC będzie miała wartość null. Konieczne będzie również sprawdzenie, czy opcja NFC została prawidłowo aktywowana przez użytkownika w konfiguracji systemu urządzenia z systemem Android (patrz rysunek 2.22) za pomocą metody *isEnabled()* adaptera, jak pokazano w kodzie źródłowym w ramce 2.8.

```
PendingIntent pendingIntent =
PendingIntent.getActivity(this, 0, new Intent(this.getClass())
addFlags(Intent.FLAG_ACTIVITY_SINGLE_TOP), 0);
```

**Ramka 2.9. Android: Inicjalizacja
PendingIntent w celu wykrycia TAGu NFC**

Instancja *PendingIntent* (patrz Ramka 2.9) aktywności wykrywania NFC z delegacją uprawnień (tj. wykrywanie NFC) mająca na celu odbiór obiektu "EXTRA_TAG" jest również wykonywana podczas tworzenia aktywności (metoda *onCreate*).

Aktywacja wykrywania NFC na pierwszym planie zostanie zainicjowana podczas wybudzania aktywności (patrz metoda *onResume* nadpisująca

przykład w ramce 2.10) z parametrami oczekującego zamiaru odbioru tagu NFC z predefiniowanymi opcjami filtrowania i listą technologii do sprawdzenia (możliwość wartości null, jeśli zostały już zadeklarowane w manifeście).

```
@Override  
public void onResume()  
{  
    super.onResume();  
    // Uwaga: intentFiltersArray i techListsArray mogą być obiektami null.  
    nfcAdapter.enableForegroundDispatch(this, pendingIntent,  
intentFiltersArray, techListsArray);  
}
```

Ramka 2.10. Android: aktywacja wykrywania NFC na pierwszym planie

W zamian wykrywanie NFC na pierwszym planie powinno zostać dezaktywowane (patrz ramka 2.11), gdy działanie zostanie wstrzymane (nie będzie już na pierwszym planie).

```
@Override  
public void onPause()  
{  
    super.onPause();  
    nfcAdapter.disableForegroundDispatch(this);  
}
```

**Ramka 2.11. Android:
dezaktywacja wykrywania
pierwszego planu NFC**

Odzyskiwanie obiektu "EXTRA_TAG" jest delegowane do aktywności NFC poprzez metodę wywołania zwrotnego *onNewIntent* wywoływaną na przykład, gdy aktywność jest uruchamiana w trybie SINGLE_TOP po wywołaniu *startActivity*. Następnie musimy tylko sprawdzić, czy intencja

filtr pasuje do wykrycia urządzenia z obsługą NFC i przetworzenia intencji (np. w metodzie processNfcIntent, jak pokazano w ramce 2.12).

```
@Override  
public void onNewIntent(Intent intent)  
{  
    String action = intent.getAction();  
    if (NfcAdapter.ACTION_TAG_DISCOVERED.equals(action) ||  
        NfcAdapter.ACTION_NDEF_DISCOVERED.equals(ac-  
tion) ||  
        NfcAdapter.ACTION_TECH_DISCOVERED.equals(action)))  
    {  
        //Metoda przetwarzania tagu NFC  
        processNfcIntent(intent);  
    }  
}
```

Ramka 2.12. Android: Odzyskiwanie intencji tagu NFC

Kod przedstawiony w Ramce 2.13 pokazuje, jak uzyskać informacje i zawartość NDEF tagu NFC z aktywności NFC.

```
public void processNfcIntent (Intent intent)  
{  
    //Informacje o tagu  
    Tag = intent.getParcelableExtra(NfcAdapter.EXTRA_TAG); byte[]  
    id = tag.getId();  
    String[] technologies = tag.getTechList();  
    int content = tag.describeContents();  
    Ndef ndef = Ndef.get(tag);  
    boolean isWritable = ndef.isWritable();  
    boolean canMakeReadOnly = ndef.canMakeReadOnly();  
    //Odzyskiwanie wiadomości  
    Parcelable[] rawMsgs = intent.getParcelableArrayExtra(  
    NfcAdapter.EXTRA_NDEF_MESSAGES);  
    NdefMessage[] msgs;  
    //Pętla na rekordach  
    if (rawMsgs != null)  
    {  
        msgs = new NdefMessage[rawMsgs.length];  
        for (int i = 0; i < rawMsgs.length; i++)
```

```
{  
    msgs[i] = (NdefMessage) rawMsgs[i];  
    NdefRecord record = msgs[i].getRecords()[i];  
    byte[] idRec = record.getId();  
    short tnf = record.getTnf();  
    byte[] type = record.getType();  
    String message = record.getPayload().toString();  
    //Używać?  
    //Pozwól Andoidowi wybrać domyślną aplikację, jeśli typ URI...  
    if (Arrays.equals(type, NdefRecord.RTD_URI))  
    {  
        Uri uri = record.toUri();  
        Intent i = new Intent(Intent.ACTION_VIEW);  
        i.setData(uri);  
        startActivity(i);  
    }  
}  
}  
inn  
{  
    //Nieznany typ znacznika, test odzyskiwania zawartości szesnastkowej?  
    byte[] empty = new byte[] {};  
    NdefRecord record = new NdefRecord(NdefRecord.TNF_UNKNOWN,  
        empty, empty, empty);  
    NdefMessage msg = new NdefMessage(new NdefRecord[] {record});  
    msgs = new NdefMessage[] {msg};  
    ...  
}  
//Przetwarzanie informacji...  
}
```

Ramka 2.13. Android: odczyt tagu NFC

Informacje takie jak UID¹⁹ i typ technologii są następnie dostępne z obiektu Tag, a także informacje o zawartości NDEF (pamiętaj, aby sprawdzić, czy tag jest sformatowany w przypadku wykrywania, które nie jest filtrowane na podstawie zawartości NDEF). Przykład pokazuje

19 Uniwersalny unikalny identyfikator tagu chroniony przed zapisem.

sposób, w jaki programista może zapętlić listę komunikatów NDEF (w przypadku kilku rekordów) i jak może przetwarzanie zawartość (tutaj, na przykład, aby wyświetlić URI), zgodnie z typem RTD (patrz sekcja 1.3.3.3.2).

2.2.2.2. Zapis tagów NFC

UWAGA: - Inicjalizacja adaptera NFC oczekującego zamiaru wykrywania NFC, aktywacja i dezaktywacja wykrywania NFC na pierwszym planie, a także odbiór zamiaru są kodowane w ten sam sposób, niezależnie od tego, czy jest to tryb czytnika, czy zapisu (patrz sekcja 2.2.2.1).

2.2.2.2.1. Tworzenie komunikatu NDEF

Tworzenie wiadomości NDEF zostało zilustrowane w Ramce 2.14 za pomocą przykładowej metody *createMyNdefMessage* przyjmującej parametr *String* dla wiadomości tekstowej i inny parametr *String* dla żądanego typu MIME i zwracającej obiekt *NdefMessage*.

```
public NdefMessage createMyNdefMessage(String text, String mimeType)
{
    //Typ wiadomości MIME
    NdefMessage msg = new NdefMessage(NdefRecord.createMime(
        mimeType, text.getBytes())

    // Inna metoda robiąca to samo...
    /**NdefRecord(NdefRecord.TNF_MIME_MEDIA,
        mimeType.getBytes(), new byte[0],
        text.getBytes())**/

    //Aplikacja typu wiadomość
    //na przykład text = "com.mbeds.android.tagnfc"

    //włącza uruchamianie aplikacji po wykryciu tagu
    /**,NdefRecord.createApplicationRecord(text)**/

    // URI typu wiadomości :
    //na przykład text = "http://www.mbeds-fr.org"
    /**,NdefMessage(NdefRecord.createUri(
        NdefRecord.createUri(Uri.encode(text)))**/

);
    return msg;
}
```

Ramka 2.14. Android: Tworzenie komunikatu NDEF

2.2.2.2.2. Zapisywanie wiadomości NDEF na tagu NFC

UWAGA.- Przed zapisem na tagu, tag musi zostać odczytany, co oznacza, że musi zostać odzyskany w taki sam sposób, jak w celu odczytania tagu w metodzie *processNfcIntent* opisanej wcześniej (patrz sekcja 2.2.1), po zbliżeniu go do urządzenia z systemem Android (zwykle czytnik NFC znajduje się z tyłu urządzenia).

Kod zaproponowany w Ramce 2.15 poniżej podaje przykład metody akceptującej wiadomość NDEF jako parametr do zapisu na tagu, a także obiekt "Tag", który został wykryty: tutaj zakładamy, że tag jest zgodny z NDEF (jeśli nie, obiekt NDEF będzie zerowy, ale nadal możemy spróbować go sformatować), następnie musimy się połączyć i przetestować, czy jest zablokowany do zapisu, czy nie, i sprawdzić, czy rozmiar wiadomości do zapisu nie jest większy niż pojemność tagu. Następnie zapis jest możliwy.

```
public static boolean writeTag(final NdefMessage message, final Tag tag)
{
    próba
    {
        int size = message.toByteArray().length;
        Ndef ndef = Ndef.get(tag);
        if (ndef == null)
        {
            //Tagi wymagające formatowania?
            NdefFormatable format = NdefFormatable.get(tag);
            if (format != null)
            {
                próba
                {
                    format.connect();
                    //Formatowanie i pisanie wiadomości:
                    format.format(message);
                    //Przykład tagu zablokowanego podczas zapisu:
                    //formatable.formatReadOnly(message);
                    format.close();
                    return true;
                } catch (Exception e) {
                    return false;
                }
            } else {
        }
    }
}
```

```
        //format == null
        return false;
    }
} else {
    //ndef!=null
    ndef.connect();

    if (!ndef.isWritable())
    {
        return false;
    }

    if (ndef.getMaxSize() < size)
    {
        return false;
    }
    ndef.writeNdefMessage(message);
    ndef.close();
    return true;
}
} catch (Exception e) {
    return false;
}
}
```

Ramka 2.15. Android: zapisywanie wiadomości NDEF na tagu

UWAGA: - W przypadku wersji wcześniejszej niż Ice Cream Sandwich 4.0 (API 14), którą można przetestować w sposób pokazany w przykładzie z ramki 2.16, tworzenie wiadomości NDEF różni się nieznacznie, co ilustruje kod sugerowany w przykładowej metodzie *createMyndefRecord* z ramki 2.17. Należy zauważyć, że ten sam zestaw znaków musi być używany do kodowania i dekodowania komunikatu NDEF.

```
if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.ICE_CREAM_SANDWICH)
{
    //Process later version...
} else {
    //Przetwarzanie wcześniejszej wersji...
}
```

Ramka 2.16. Android: testowanie wersji

```
public NdefRecord createMyNdefRecord(String message)
{
    byte[] langBytes = Locale.ENGLISH
        .getLanguage().getBytes(Charset.forName("US-ASCII"));
    byte[] textBytes = message.getBytes(Charset.forName("UTF-8"));
    char status = (char) (langBytes.length);
    byte[] data = new byte[1 + langBytes.length + textBytes.length];
    data[0] = (byte) status;
    System.arraycopy(langBytes, 0, data, 1, langBytes.length);
    System.arraycopy(textBytes, 0, data, 1 + langBytes.length, textBytes.length);
    message=message.trim();
    NdefRecord ndefRec;
    //RTD do wyboru zgodnie z typem zawartości
    ndefRec =new NdefRecord(NdefRecord.TNF_WELL_KNOWN,
        NdefRecord.RTD_TEXT, new byte[0], data);
    //NdefRecord.RTD_URI, new byte[0], data);
    //NdefRecord.RTD_SMART_POSTER, new byte[0], data);
    ...
    return ndefRec;
}
```

Ramka 2.17. Android: tworzenie rekordu NDEF

2.2.3. Wdrożenie trybu NFC P2P za pomocą Android

Od systemu Android Ice Cream Sandwich w wersji 4.0 (API 14) możliwe jest udostępnianie poczty e-mail, kontaktów i ulubionych z innym telefonem komórkowym z systemem Android, dzięki Android Beam, który jest funkcją nieodłączną dla systemu Android.

Beam nie tylko umożliwia emulowanie sposobu, w jaki tag NFC działa z telefonem komórkowym i udostępnianie wiadomości NDEF (tekst, adres URL, SMS, telefon, wizytówka, podpis, parametry Wi-Fi, aplikacje, akcje itp.), jeśli inny telefon komórkowy z obsługą NFC zostanie zbliżony, wykrywa urządzenie tak, jakby było tagiem NFC, a także umożliwia udostępnianie plików multimedialnych z pamięci telefonu komórkowego, od wersji Android 4.1 Jelly Bean (API 16).

Będziemy zatem musieli zadeklarować uprawnienia, aby uzyskać dostęp do zewnętrznej pamięci masowej w manifeście, jak pokazano w ramce 2.18.

```
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>
```

Ramka 2.18. Android: deklaracja uprawnień dostępu do pamięci masowej

W przeciwnieństwie do trybu czytnika/nagrywarki, który umożliwia rozpoczęcie aktywności bez konieczności uruchamiania aplikacji przez użytkownika zgodnie z filtrem NFC zadeklarowanym w manifeście do wykrywania NFC, Android *Beam* działa tylko wtedy, gdy aktywność jest w toku (na pierwszym planie): nie ma filtra na aktywność, ale opcja "Beam" musi być aktywowana w parametrze systemowym telefonu komórkowego z Androidem, a dostępność Beam musi być testowana programowo, jak kod w pudełku 2.19 pokazuje.

```
If (!nfcAdapter.isNdefPushEnabled())
{
    //Poproś użytkownika o aktywację opcji Beam
}
```

Ramka 2.19. Android: sprawdź, czy opcja wiązki jest aktywna

2.2.3.1. Używanie Android beam do wysyłania wiadomości NDEF

Aby wysyłać wiadomości NDEF, wystarczy zaimplementować funkcję *setNdefPushMessage* adaptera NFC (patrz ramka 2.20):

```
// Odzyskiwanie adaptera NFC
NfcAdapter nfcAdapter = NfcAdapter.getDefaultAdapter(getApplicationContext());
//Użycie poprzedniej metody (zapis znaczników) w celu dostosowania do RTD
NdefMessage msg = createMyNdefMessage("Hello World!", "text/plain");
//Wyślij wiadomość na urządzenie P2P.
//użytkownik musi dotknąć, aby zatwierdzić transfer
nfcAdapter.setNdefPushMessage(msg, this);
```

Ramka 2.20. Android: wysyłanie wiadomości NDEF w P2P z beam

2.2.3.2. Korzystanie z *Android beam* z wywołaniami zwrotnymi

Inna, bardziej dynamiczna metoda polega na wykorzystaniu wywołań zwrotnych do wyzwalania wysyłania wiadomości NDEF po wykryciu bliskości urządzenia NFC.

```
public class TagActivity extends Activity implements CreateNdefMessageCallback,  
OnNdefPushCompleteCallback  
{  
    @Override  
    protected void onCreate(Bundle savedInstanceState)  
    {  
        super.onCreate(savedInstanceState);  
        NfcAdapter nfcAdapter =  
            NfcAdapter.getDefaultAdapter(getApplicationContext());  
        // Subskrypcja dla wywołania zwrotnego tworzenia wiadomości  
        nfcAdapter.setNdefPushMessageCallback(this, this);  
        // Subskrypcja wywołania zwrotnego zakończenia wysyłania wiadomości  
        nfcAdapter.setOnNdefPushCompleteCallback(this, this);  
    }  
  
    @Override  
    // Ta metoda jest używana po wykryciu urządzenia NFC P2P.  
    public NdefMessage createNdefMessage(NfcEvent event)  
    {  
        // Użyj wcześniej utworzonej metody:  
        NdefMessage msg = createMyNdefMessage("Hello World!", "text/plain");  
        return msg;  
    }  
  
    @Override  
    // Ta metoda jest używana po odebraniu wiadomości.  
    public void onNdefPushComplete(NfcEvent event)  
    {  
        // Powiadomienie użytkownika  
    }  
}
```

Ramka 2.21. Android: Implementacja wiązki Android z wywołaniami zwrotnymi

Ta implementacja przedstawiona w ramce 2.21 ma kilka zalet, między innymi fakt, że wiemy o początku i końcu wymiany.

2.2.3.3. Przesyłanie plików za pomocą Android beam

Funkcja przesyłania plików NFC za pomocą beam jest metodą push, która przyjmuje URI pliku do przesłania jako parametr. Najpierw musimy utworzyć instancję obiektu File z katalogiem, w którym znajduje się plik, a następnie kolejną z obiektem File i nazwą pliku (w formacie ciągu znaków) wskazaną w kodzie przedstawionym w Ramce 2.22.

```
//Directory hostujący plik do przesłania
Plik myFolder = Environment.getExternalStoragePublicDirectory(
    //Folder zdjęć
    Environment.DIRECTORY_PICTURES);
//Folder zdjęć
//Environment.DIRECTORY_DCIM);
//Folder wideo
//Environment.DIRECTORY_MOVIES);
//Folder Muzyka
//Environment.DIRECTORY_MUSIC);
//Folder Downloads
//Environment.DIRECTORY_DOWNLOADS);
//Folder Dokumenty
//Environment.DIRECTORY_DOCUMENTS);
//lub katalog o nazwie...
//Environment.getExternalStorageDirectory() + "/MyDirectory";

//Nazwa pliku do przesłania wraz z jego rozszerzeniem
String myFileName = "monFichier.extension";
//Pełny obiekt pliku
File myFile = new File(myFolder, myFileName);
//Upewnij się, że plik jest dostępny dla wszystkich w trybie czytnika
myFile.setReadable(true, false);
//Udostępnienie pliku przez Beam dla transferu NFC.
nfcAdapter.setBeamPushUris(new Uri[] {Uri.fromFile(myFile)}, this);
```

Box 2.22. Android: przesyłanie plików w P2P za pomocą beam

UWAGA - W przypadku *Beam* użytkownik wysyłający wiadomość/plik do przesłania jest proszony o "dotknietie" ekranu w celu rozpoczęcia przesyłania (gdy urządzenie źródłowe i docelowe znajdują się w pobliżu na czas przesyłania).

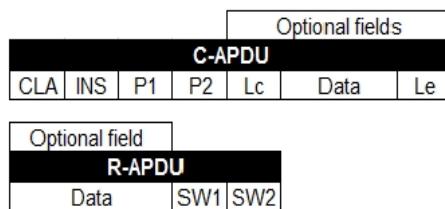
Powiadomienie o postępie przesyłania pojawia się w obszarze powiadomień urządzenia docelowego, a powiadomienie pokazuje zakończenie przesyłania: użytkownik otrzymujący je jest następnie proszony o "dotknięcie" ekranu, aby wyświetlić zawartość, która została przesłana i pobrana na urządzenie docelowe.

2.2.4. Wdrażanie

Karta NFC emulacja tryb z Android

Specyfiką trybu emulacji karty NFC jest adresowanie tagów typu 4 (patrz sekcja 1.3.3.3.1) i komunikacja oparta na protokole APDU dla kart inteligentnych dziedziczących po JavaCardTM środowiska. Usługi są hostowane w i wykonany w w karty inteligentnej lub systemu operacyjnego SE. Interfejs APDU jest zarządzany (implementowany) w (zewnętrznym) punkcie wejścia klas JavaCardTM dziedziczących z nadklasy Applet w metodzie *process*. Metoda ta otrzymuje jako parametr polecenie APDU (patrz rysunek 2.24) i przetwarza je.

Odpowiedź APDU jest następnie wysyłana z powrotem (w zamian lub na żądanie z innym APDU) do aplikacji klienckiej, która ustanawia sesję komunikacji z kartą inteligentną.



Rysunek 2.24. Struktura APDU

Odczyt kart inteligentnych lub komunikacja SE w trybie emulacji karty NFC to komunikacja typu master-slave, w której inicjatorem jest aplikacja kliencka (master), która jest zewnętrzna w stosunku do układu elektronicznego, wysyłająca polecenia APDU i odbierającą odpowiedzi APDU od usługi hostowanej w układzie (slave).

2.2.4.1. Polecenie APDU "SELECT": wybór aplikacji

Aby komunikować się z usługą chipa, należy znać jego identyfikator AID²⁰: rozmiar identyfikatora AID wynosi od 5 do 16 bajtów; składa się on z identyfikatora dostawcy (Registered ID) zarządzanego przez organy regulacyjne oraz z zastrzeżonego rozszerzenia aplikacji (Proprietary application Identifier eXtension) zarządzanego przez dostawcę usług.

Konfiguracja sesji komunikacji z usługą wbudowaną w kartę jest inicjowana poleceniem instrukcji SELECT standardu APDU, którego ładunek jest wypełniony identyfikatorem AID docelowej usługi. Na przykład w ramce 2.23 pokazano konstrukcję instrukcji *SELECT APDU* w celu zainicjowania komunikacji z usługą, której AID (fikcyjny) to "F00000000000000000000000000000001": oczekiwana odpowiedź będzie zawierać tylko kod stanu (powodzenie lub niepowodzenie) wykonania polecenia.

```
byte[] SELECT = {  
    //CLA Class: Karty Moneo, Mastercard, Visa©...  
    (bajt) 0x00,  
    //INS: Instrukcja SELECT  
    (bajt) 0xA4,  
    //P1: Parametr 1  
    (bajt) 0x04,  
    //P2: Parametr 2  
    (bajt) 0x00,  
    //Długość polecenia Lc: 10 bajtów  
    (bajt) 0x0A,  
    //Ladunek: AID  
    (byte) 0xF0,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x01  
    //Le Długość oczekiwanej odpowiedzi: 0 bajtów  
    (bajt) 0x00,  
};
```

Ramka 2.23. Konfigurowanie instrukcji *SELECT* standardu APDU

20 Identyfikator apletu: W JavaCard™ każdy aplet jest identyfikowany dzięki swojemu identyfikatorowi AID (konwencja nazewnictwa zgodna ze standardem ISO 7816).

2.2.4.2. Komunikacja z tagiem NFC w trybie emulacji karty

Klasa Android *IsoDep* umożliwia operacje w trybie emulacji karty NFC przy użyciu protokołu określonego w normie ISO 14443-4 (patrz sekcja 1.3.4).

Ramka 2.24 pokazuje, jak po odczytaniu (patrz sekcja 2.2.2.1) typu znacznika ISO 14443-4 kart inteligentnych NFC (zgodnie z technologią określoną w filtrze, na przykład patrz sekcja 2.2.2), aplikacja Androïd staje się terminalem, który może wysyłać polecenia i odbierać odpowiedzi APDU z karty (lub jej emulacji) dzięki obiekowi *IsoDep*.

```
IsoDep card = IsoDep.get(tag);
card.connect();

/Wysłanie polecenia APDU
byte[] result = card.transceive(SELECT);

/Sprawdź odpowiedź
if (!(result[0] == (byte) 0x90 && result[1] == (byte) 0x00)) {
    //Zarządzaj błędem
    ...
}
...
card.close();
```

Ramka 2.24. Komunikacja z tagiem NFC w trybie emulacji karty

2.2.4.3. Komunikacja z SE

W sekcji 2.2.4.2 zobaczyliśmy, w jaki sposób możemy zaimplementować komunikację z tagiem typu karty inteligentnej NFC, gdy zostanie on odczytany przez aplikację Androïd po zbliżeniu go do urządzenia z systemem Androïd obsługującym NFC. Ta aplikacja może wysyłać polecenia APDU do apletu zainstalowanego w chipie tagu i otrzymywać odpowiedź zwrotną. Gdy aplet jest hostowany w chipie SE wewnątrz telefonu komórkowego, aplikacja działająca w systemie operacyjnym hosta może być zmuszona do komunikowania się z usługami SE. Może to być interfejs użytkownika końcowego lub interfejs dla zdalnego systemu: na przykład w przypadku aplikacji typu portfel.

która umożliwia użytkownikowi sprawdzenie jego ostatniej transakcji, lub aplikacja proxy do zdalnego zarządzania cyklem życia usług SE.

UWAGA - Ponieważ dostęp do SE jest ograniczony, komunikacja z usługami SE wymaga uprawnień i poziomów bezpieczeństwa specyficznych dla implementacji SEI, usługi i funkcji, do których uzyskuje się dostęp. Tutaj skupiamy się tylko na implementacji komunikacji, ale musimy podkreślić oczywiste znaczenie bezpieczeństwa, bez którego SE nie byłby SE.

Komunikacja z SE odbywa się za pośrednictwem API (zewnętrznej biblioteki), którego dostępność zależy od wersji systemu Android, modelu urządzenia i konfiguracji SE.

2.2.4.3.1. Interfejs API "nfc_extras" systemu Android

Biblioteka Android "nfc_extras.jar²¹", która wymaga deklaracji w folderze manifestu (ramka 2.25), zawiera singleton *NfcAdapterExtras* (patrz ramka 2.26) i zapewnia statyczne metody dostępu i wymiany komunikatów APDU z aplreami eSE reprezentowanymi przez klasę *NfcExecutionEnvironment*.

```
...
<uses-permission android:name="android.permission.NFC"/>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<używa uprawnienia android:name="android.permission.WRITE_SECURE_SETTINGS"/>
<uses-permission android:name="android.permission.NFC"/>
<uses-permission android:name="com.android.nfc.permission.NFCEE_ADMIN"/>
...
<aplikacja...
    ...
    <uses-library android:name=
        "com.google.android.nfc_extras" android:required="true"/>
</aplikacja>
```

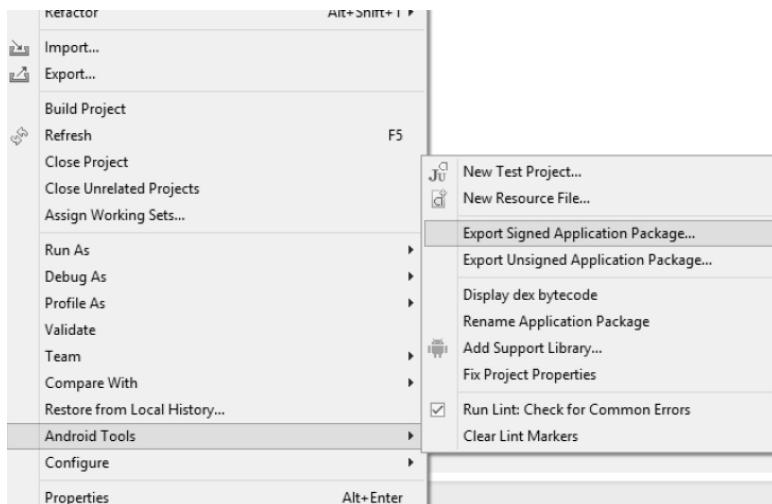
Ramka 2.25. Deklaracje w pliku AndroidManifest.xml

²¹ Źródła dostępne tutaj: http://grepcode.com/file/repository.grepcode.com/java/ext/com.google.android/android/2.3.7_r1/com/android/nfc_extras/NfcAdapterExtras.java.

UWAGA: - Interfejs API *nfc_extras* dostępny w systemie Android Ice Cream Sandwich (ICS) niższego poziomu niż *IsoDep* również umożliwia aktywację i dezaktywację trybu emulacji karty NFC.

```
NfcAdapterExtras adapterExtras =  
    NfcAdapterExtras.get(NfcAdapter.getDefaultAdapter(context));  
NfcExecutionEnvironment nfcEe =  
    adapterExtras.getEmbeddedExecutionEnvironment();  
nfcEe.open();  
byte[] response = nfcEe.transceive(APDUcommand);  
nfcEe.close();
```

Ramka 2.26. Przykład użycia klasy "NfcAdapterExtras"



Rysunek 2.25. Eksportowanie podpisanej aplikacji

Od wersji 4.0.4 Android ICS (API 15) podpis aplikacji nie jest wystarczający: Android używa certyfikatu wygenerowanego²² z kluczem prywatnym w celu (samodzielniego) podpisywania aplikacji (patrz rysunek 2.25). Aplikacja Android uzyskująca dostęp do SE (tj. Google Wallet) była

22 Zobacz podpis aplikacji na stronie Android dla programistów: <http://developer.android.com/tools/publishing/app-signing.html>.

zadeklarowany w pliku "nfcee_access.xml" znajdującym się w katalogu systemowym "/etc" smartfona (patrz Ramka 2.27), który wymaga dostępu "superuser²³" (specjalne konto użytkownika z podwyższonymi uprawnieniami nadanymi w systemie).

```
<?xml version="1.0" encoding="utf-8"?>
<resources xmlns:xliff="urn: oasis:names:tc:xliff:document:1.2">
<!-- ! Podpis aplikacji wygenerowany za pomocą Keytool -->
<signer android:signature="XXXXXXXXXXXXXXXXXXXXXX">.
<package android:name="org.mbd.s.android.tagnfc">.
</package></signer>
...
</resources>
```

Ramka 2.27. Deklaracje w pliku nfcee_access.xml

2.2.4.3.2. SmartCard API (OMAPI)

SmartCard API for Android jest implementacją standardu *Open Mobile API* (OMAPI, patrz sekcja 1.3.6), którego projekt *seek-for-android* jest opublikowany na [github²⁴](#); API umożliwia aplikacjom mobilnym Android komunikację z SE (w trybie emulacji karty). Ten interfejs API można dodać do listy pakietów w menedżerze SDK Androida (patrz sekcja 2.1.2.1): aby to zrobić, musimy dodać witrynę pobierania za pomocą menu *Narzędzia|Dodane witryny|Witryny zdefiniowane przez użytkownika* w menedżerze SDK i dodać adres URL z pola 2.28, gdzie XX należy zastąpić numerem Android SDK, z którym projekt zostanie skompilowany (patrz rysunek 2.26).

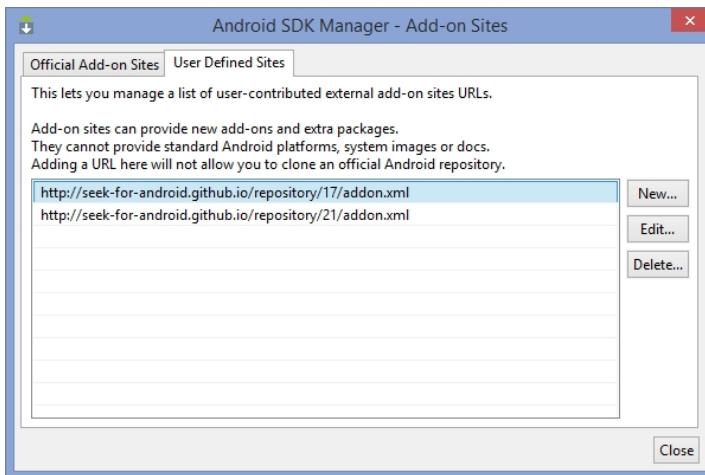
```
http://seek-for-android.googlecode.com/svn/trunk/repository/XX/addon.xml
```

Ramka 2.28. Pobieranie adresu URL dla interfejsu API SmartCard

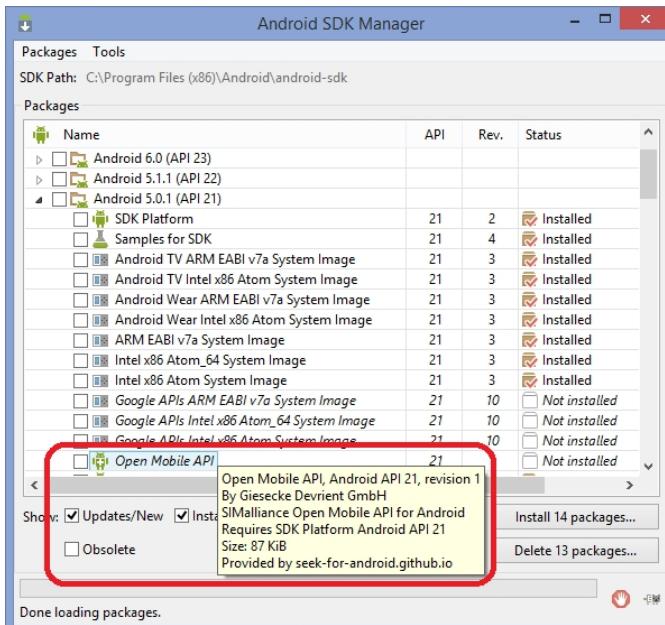
Następnie na liście SDK pojawi się pakiet OMAPI (patrz rysunek 2.27).

23 W systemie Android superużytkownik jest nazywany "Root". Metoda z komentarzami jest sugerowana na forum CNET pod tym adresem URL: <http://forums.cnetfrance.fr/topic/193517-comment-rooter-android--methode-universelle-de-root>.

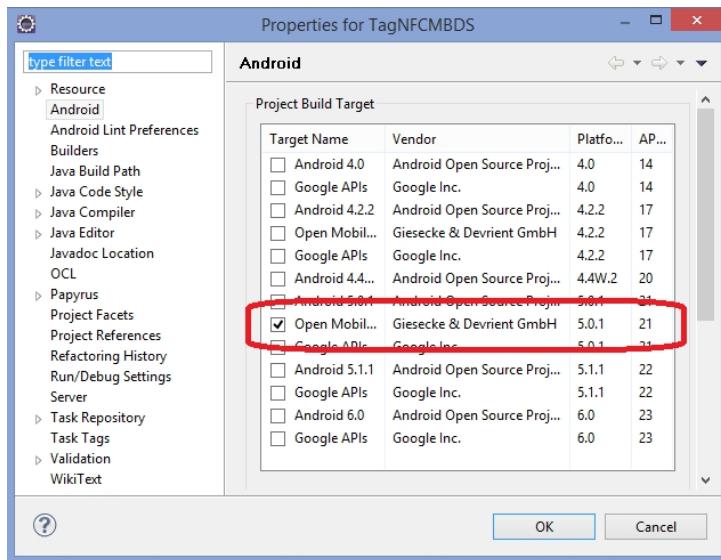
24 Szukaj projektu Android: <https://github.com/seek-for-android>.



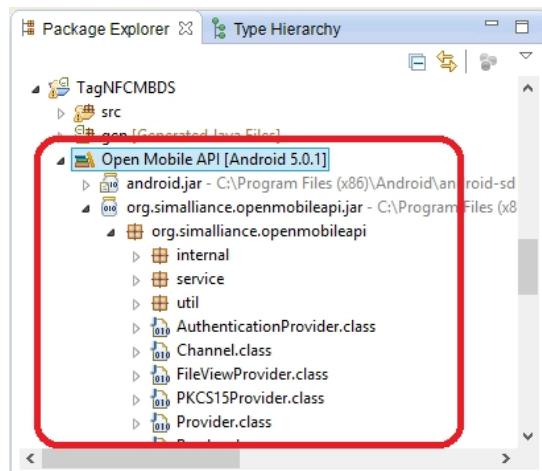
Rysunek 2.26. Dodawanie pakietów w menedżerze Android SDK



Rysunek 2.27. Pakiet Open Mobile API



Rysunek 2.28. Właściwości projektu Android: wersja docelowa



Rysunek 2.29. Biblioteka Open Mobile API

Po zainstalowaniu pakietu *Open Mobile API* możemy wybrać go w wersji docelowej we właściwościach projektu Android (patrz rysunek 2.28)

aby uzyskać dostęp do interfejsów *SmartCard API* i skompilować projekt z wymaganymi bibliotekami (patrz rysunek 2.29).

Korzystanie z *Open Mobile API* musi być zadeklarowane w pliku *AndroidManifest*, podobnie jak uprawnienia dostępu do SE (patrz Ramka 2.29).

```
<uses-library android:name="org.simalliance.openmobileapi" android:required="true"/>
<uses-permission android:name="org.simalliance.openmobileapi.SMARTCARD"/>
```

Ramka 2.29. Deklaracje Open Mobile API (OMAPI)

Implementacja łączności SE w Android Activity²⁵ :

Ramka 2.30 pokazuje, w jaki sposób dostęp do SE odbywa się za pośrednictwem usługi zadań w tle. Gdy usługa jest dostępna, powiadamia aktywność, implementując metodę "serviceConnected" interfejsu "SEserviceCallBack".

```
import org.simalliance.openmobileapi.*;

public class SeActivity extends Activity implements SEService.CallBack {
    private SESERVICE seService;
    ...
    //Zarządzaj połączeniem w celu uzyskania dostępu do usługi SE:
    //inicjowanie usługi z kontekstem i wywołaniem zwrotnym
    try {
        seService = new SESERVICE(this, this);
    } catch (SecurityException e) {
        // Przetwarzanie wyjątku
    } catch (Exception e) {
        // Przetwarzanie wyjątku
        ...
    }
    ...
    //Wywołanie zwrotne połączenia
    public void serviceConnected(SESERVICE service)
```

²⁵ Zainspirowany źródłem, które można znaleźć pod tym adresem URL:
<http://code.google.com/p/seek-for-android/wiki/UsingSmartCardAPI>

```
{  
    /*Komunikacja z bezpiecznym elementem  
    próba  
    {  
        Reader[] readers = seService.getReaders();  
  
        //Testowanie obecności czytnika NFC (tj. wewnętrznego)  
        if (readers.length < 1)  
        {  
            powrót;  
        }  
        //Na przykład używany jest pierwszy czytnik (w zależności od docelowego SE)  
        Session session = readers[0].openSession();  
        //Otwarcie kanału za pomocą AID apletu  
        Channel = session.openLogicalChannel(new byte[] { (byte) 0xD2,  
            0x76, 0x00, 0x01,  
            0x18, 0x00, 0x02, (bajt) 0xFF,  
            0x49, 0x50, 0x25, (bajt) 0x89,  
            (bajt) 0xC0, 0x01, (bajt) 0x9B, 0x01  
        });  
  
        //Testowanie komunikacji z apletem  
        bajt sw1 = (bajt) 0x90; bajt  
        sw2 = 0x10;  
        byte[] data = new byte[3] {0x00, 0x00, 0x00};  
        byte[] respApdu = channel.transmit(new byte[] {sw1, sw2, data[0],  
            data[1], data[2]});  
  
        channel.close();  
  
        //Sprawdź zwrócony kod statusu  
        Jeśli (respApdu[0]===(byte) 0x90 i respApdu[1]== 0x00)  
        {  
            //Pobierz część "dane"  
            data = new byte[respApdu.length - 2];  
            System.arraycopy(respApdu, 0, data, 0, respApdu.length - 2);  
            //Przetwarzanie odpowiedzi  
            ...  
        } else {  
            //Zarządzaj błędem  
        }  
        ...  
    } catch (Exception e) {  
}
```

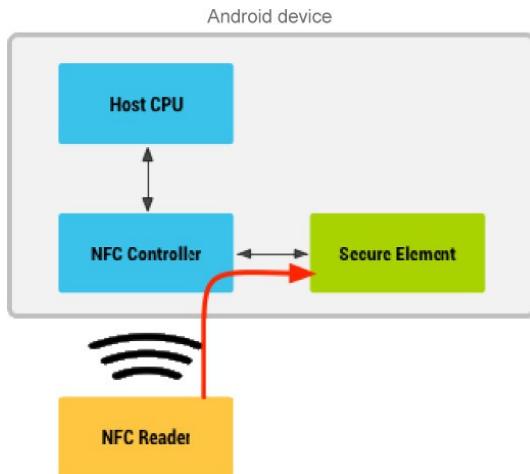
```
//Zarządzanie błędem  
...  
}  
}  
//Nie zapomnij dezaktywować usługi  
@Override  
protected void onDestroy()  
{  
    if (seService != null && seService.isConnected())  
    {  
        seService.shutdown();  
    }  
    super.onDestroy();  
}  
}
```

Ramka 2.30. Dostęp do SE za pomocą OMAPI

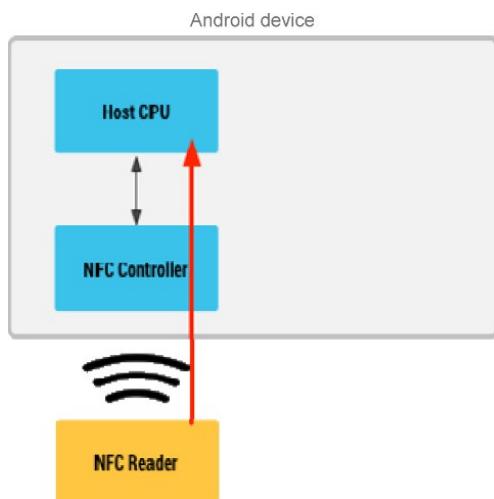
2.2.5. Tworzenie usług NFC z systemem Android HCE

Do tej pory widzieliśmy, że rozwój trybu emulacji karty z Andoidem polegał na implementacji (Android) aplikacji klienckiej usługi wykonywanej w systemie operacyjnym SE hostowanego w komponencie sprzętowym typu smart card: klasa *IsoDep* umożliwia implementację aplikacji Android działających jako czytnik NFC zdolny do komunikacji z tagiem typu smart card NFC (w tym SE innego urządzenia) wykrytym, gdy inne urządzenie jest zbliżone, podczas gdy *OMAPI* API umożliwia komunikację z SE hostowanym w tym samym urządzeniu co aplikacja kliencka.

Rozwój trybu HCE jest specyfiką trybu emulacji karty NFC, w którym usługa implementująca interfejs APDU działa jako SE. Jednak w przeciwieństwie do usługi wykonywanej w SE, usługa HCE jest wykonywana w systemie operacyjnym Android, podobnie jak każda inna usługa działająca w tle systemu Android: z "zewnętrznego" punktu widzenia czytnika kart inteligentnych NFC usługa HCE jest "postrzegana" jako karta inteligentna NFC; tylko implementacja aplikacji klienckiej działającej na tym samym urządzeniu z Andoidem różni się od tej, która komunikuje się ze sprzętowym SE.



Rysunek 2.30. *Android: routing APDU w kierunku SE*



Rysunek 2.31. *Android: routing APDU z HCE*

Jak pokazano na rysunku 2.30, gdy usługa jest hostowana w SE, routing poleceń APDU nie przechodzi przez system operacyjny SE.

Komunikacja między czytnikiem NFC a usługą chipową nie jest zatem "widoczna" z systemu operacyjnego Android, w przeciwieństwie do trybu HCE: od wersji Android KitKat (4.4) z HCE, usługi NFC w trybie emulacji karty są deklarowane w systemie operacyjnym Android; następnie instrukcja APDU SELECT wskazująca AID usługi docelowej umożliwia systemowi Android określenie routingu poleceń APDU, jak pokazano na rysunku 2.31.

UWAGA: - Tryb HCE ma wady i zalety: ponieważ HCE nie jest oparty na sprzętowym SE i korzysta z dostępu do wszystkich funkcji urządzenia hosta, usługa HCE ma pełny dostęp do łączności systemu operacyjnego Android, uwalniając w ten sposób dostawców usług od złożonego i ograniczającego zarządzania cyklem życia usługi SE, spowodowanego zamkniętym środowiskiem i ograniczonym dostępem SE, co wymaga przejścia przez jedną lub kilka stron trzecich (wystawca SE i/lub TSM) i uniknięcia dodatkowych kosztów, które to powoduje. Usługi HCE są dostępne za pośrednictwem systemu informatycznego, podobnie jak każda inna aplikacja mobilna, i mogą być zarządzane kompleksowo przez dostawcę usług, bez konieczności korzystania z usług strony trzeciej. Jednak HCE nie korzysta z zabezpieczeń nieodłącznie związanych z technologią kart inteligentnych, a dostawca usług HCE musi zintegrować zabezpieczenia we wszystkich aspektach swojego systemu. Ponadto, w przeciwieństwie do usług NFC hostowanych w SE, które będą działać po wyłączeniu smartfona lub rozładowaniu baterii, usługa HCE jest dostępna tylko wtedy, gdy system działa (jest włączony).

2.2.5.1. Deklaracje dla HCE w pliku *AndroidManifest.xml*.

Gdy projekt Android ma jedną (lub kilka) usług HCE, należy oczywiście zadeklarować uprawnienie NFC, ale także użycie funkcji HCE (patrz ramka 2.31).

```
<uses-permission android:name="android.permission.NFC" android:required="true"/>
<uses-feature android:name="android.hardware.nfc.hce" android:required="true"/>
```

Ramka 2.31. Deklaracja dla HCE w *AndroidManifest.xml*

Podobnie jak w przypadku każdej aktywności systemu Android, usługi działające w tle systemu Android są deklarowane w sekcji *aplikacji* manifestu (patrz sekcja 2.1.2.3.4) przy użyciu tagu *usługi*. Atrybut *android:permission* umożliwia wskazanie uprawnienia *BIND_NFC_SERVICE* do nawiązania połączenia z usługą działającą w innym wątku wykonawczym, w tle, innym niż główny wątek aktywności. Ponieważ jest to usługa HCE, należy dodać filtr *HOST_APDU_SERVICE*, który pokaże systemowi, że jednostki APDU muszą być kierowane do tej usługi zgodnie z identyfikatorami AID dostarczonymi w pliku XML załącznika (patrz przykład podany w ramce 2.33) i odniesionymi w znaczniku *metadanych* "host apdu service", jak pokazano w kodzie w ramce 2.32.

```
<usługa
    android:name="my.app.package.MyHCEService"
    android:permission="android.permission.BIND_NFC_SERVICE">
    <intent-filter>
        <action
            android:name="android.nfc.cardemulation.action.HOST_APDU_SERVICE"
            CE"/>
        <category android:name="android.intent.category.DEFAULT"/>
    </filtr treści>
    <meta-data android:name="android.nfc.cardemulation.host_apdu_service"
        android:resource="@xml/my_host_metadata"/>
</service>
```

Ramka 2.32. Deklarowanie usługi HCE (AndroidManifest.xml)

W przykładzie przedstawionym w Ramce 2.32, aplikacja posiada klasę, tutaj nazwaną "MyHCEService", dziedziczącą z nadklasy Android *HostApduService*. Usługi implementujące interfejs APDU komunikacji NFC w trybie emulacji karty zarządzane przez usługę HCE "MyHCEService" są opisane w pliku XML, tutaj o nazwie "my_host_metadata", utworzonym w podfolderze *res/xml/* widoku drzewa projektu aplikacji (patrz sekcja 2.1.2.3.3), zasugerowany jako przykład w Ramce 2.33: w tym przykładzie tekst opisujący usługi i grupę jest odzyskiwany w pliku string zasobów projektu (patrz sekcja 2.1.2.3.3) deklarującym dwie usługi routingu APDU przeznaczonych dla AID (tutaj fikcyjnego) "F00000000000000000000000000000001" i "F00000000000000000000000000000002".

```
<host-apdu-service xmlns:android="http://schemas.android.com/apk/res/android"
    android:description="@string/my_service_description"
    android:requireDeviceUnlock="false">
    <aid-group
        android:category="other"
        android:description="@string/my_group_description">
        <aid-filter android:name="F00000000000000000001"/>
        <aid-filter android:name="F00000000000000000002"/>
        ...
    </aid-group>
</host-apdu-service>
```

Ramka 2.33. Przykład opisu usługi Android HCE

UWAGA: - Tylko dwie kategorie są zarządzane przez system Android: kategoria usług płatniczych (CATEGORY_PAYMENT) i inne usługi (CATEGORY_OTHER).

2.2.5.2. Implementacja nadklasy Android HostApduService

Usługa HCE w systemie Android rozszerza nadkласę HostApduService; klasa Applet z JavaCardTM implementująca usługę HCE wymaga nadpisania metody *processCommandApdu* otrzymującą polecenie APDU jako parametr (patrz sekcja 2.2.4), co ilustruje próbka kodu przedstawiona w ramce 2.34. Usługa HCE działa jak *aplet*: po otrzymaniu polecenie APDU jest analizowane i przetwarzane przez usługę HCE, a następnie odpowiedź jest zwracana do aplikacji nadawcy polecenia, w tym kod stanu wykonania.

```
import android.nfc.cardemulation.HostApduService;
import android.os.Bundle;

Klasa publiczna MyHCEService rozszerza HostApduService
{

    @Override
    public byte[] processCommandApdu(byte[] commandApdu, Bundle extras)
    {
        /Przeanalizowanie i przetworzenie poleceniaApdu
        return byteResponse;
    }
}
```

Ramka 2.34. Przykład wdrożenia usługi HCE dla systemu Android

Jak wspomniano wcześniej, zaletą implementacji HCE w porównaniu do JavaCard™ jest to, że korzysta ona ze wszystkich zaawansowanych funkcji systemu Android, a zwłaszcza z dostępu do łączności urządzenia, na którym jest wykonywana: w ten sposób polecenia APDU mogą być przetwarzane lokalnie, ale mogą być również wysyłane do zdalnego serwera w celu przetworzenia na zewnątrz. Co więcej, HCE umożliwia przechowywanie historii transakcji bez ograniczeń zasobów pamięciowych SE, a dodatkowe informacje mogą być brane pod uwagę podczas przetwarzania poleceń, takich jak lokalizacja lub znacznik czasu transakcji.

UWAGA: - W przypadku HCE dostawca usług musi upewnić się, że usługa ma wystarczający poziom bezpieczeństwa, na przykład poprzez wdrożenie mechanizmów szyfrowania i uwierzytelniania.

2.2.5.3. Komunikacja z usługą HCE z systemu operacyjnego Android

Podobnie jak usługa SE, dostęp do usługi HCE można uzyskać z pośrednictwem urządzenia zewnętrznego działającego jako czytnik kart inteligentnych NFC (patrz sekcja 2.2.4.2). Nie dotyczy to aplikacji uruchomionej na tym samym urządzeniu, na którym działa usługa HCE: OMAPI umożliwia ogólną komunikację z usługą SE (patrz sekcja 2.2.4.3.2) niezależnie od konfiguracji (karta SIM, eSE lub microSD), ale nie z usługą HCE.

Aby komunikować się z usługą HCE, musimy użyć mechanizmów komunikacji międzyprocesowej Androida: komunikacja może odbywać się za pomocą *intencji* (patrz sekcja 2.1.3) filtrowanej na jednej lub kilku predefiniowanych akcjach identyfikowanych przez ciąg znaków kodu (patrz Ramka 2.35). *Intencja* może zawierać parametry w *Bundle*, a następnie jest transmitowana przez system w celu odebrania i przetworzenia przez *BroadcastReceiver* z uwzględnieniem akcji zdefiniowanych w filtrze (patrz ramka 2.36).

```
Intent myIntent = new Intent("myfilter.on.myHCEService.action");
myIntent.putExtra(myParamKey, myParamValue);
context.sendBroadcast(myIntent);
```

Ramka 2.35. Przykład wysyłania przefiltrowanej intencji dotyczącej wcześniejszej określonego działania

UWAGA - Korzystanie z *BroadcastReceiver* nie jest jedynym rozwiązaniem. Na p r z y k ł a d m o ż e m y z n a l e ć i n n e r o z w i ąż a n i a²⁶ o p a r t e n a A n d r o i d *Messenger*²⁷.

Następnie wystarczy zaimplementować *BroadcastReceiver* dedykowany do odbierania intencji dla usługi HCE, na przykład w samej usłudze (patrz Ramka 2.36).

```
Statyczna klasa publiczna MyInnerReceiver rozszerza BroadcastReceiver
{

    @Override
    public void onReceive(Context context, Intent myIntent)
    {
        if (myHceServiceInstance==null)
        {
            //Process the case in which HCE service is not started?
        }
        //Pobierz odebrany pakiet (pasujący do filtra akcji)
        if (myIntent != null &&
            myIntent.getAction().equals("myfilter.on.myHCEService.action"))
        {
            Bundle extras = myIntent.getExtras();
            if (extras != null)
            {
                myStringParam = extras.getString(myParamKey);
                //Przetwarzanie otrzymanych parametrów
            }
        }
    }
};
```

Ramka 2.36. Android: implementacja odbiornika BroadcastReceiever

Aby intencje mogły być kierowane do *odbiornika BroadcastReceiever*, ten ostatni musi wcześniej zarejestrować się w systemie. Android oferuje dwa

26 Przykład implementacji (jeden z kilku) można znaleźć pod adresem: <https://android.googlesource.com/platform/frameworks/base/+/51b6322/core/java/android/nfc/cardemulation/HostApduService.java>.

27 <http://developer.android.com/reference/android/os/Messenger.html>.

rozwiązania deklarowania *BroadcastReceiver*: (i) deklaracja statyczna w pliku manifestu (patrz Ramka 2.37) lub (ii) deklaracja dynamiczna podczas przetwarzania (patrz Ramka 2.38).

```
<receiver adroid:name = "com.example.MyHCEService$MyInnerReceiver"
    android:enabled="true"
    android:exported="true">
    <intent-filter>
        <action android:name = "myfilter.on.myHCEService.action"/>
        <category android:name = "android.intent.category.DEFAULT"/>
    </filtr treść>
</odbiornik>
```

Ramka 2.37. Statyczna deklaracja odbiornika BroadcastReceiver

UWAGA - W przykładzie w ramce 2.37 klasa "MyInnerReceiver" jest podklassą usługi "MyHCEService": aby zadeklarować podklassę w manifeście Androida, nazwa zawiera ścieżkę pakietu, oddzielając klasę nadziedną od podklasy znakiem "\$".

```
@Override
public void onCreate()
{
    super.onCreate();
    myReceiver = new MyInnerReceiver();
    IntentFilter myFilter = new IntentFilter("myfilter.on.myHCEService.action");
    registerReceiver(myReceiver, myFilter);
}
@Override
public void onDestroy()
{
    unregisterReceiver(myReceiver);
    super.onDestroy();
}
```

Ramka 2.38. Dynamiczna rejestracja odbiornika BroadcastReceiver

UWAGA - W przykładzie w ramce 2.38 rejestracja jest przeprowadzana zgodnie z metodą *onCreate*, a wyrejestrowanie jest przeprowadzane

zgodnie z metodą *onDestroy* w przypadku usługi. W przypadku aktywności wolimy używać metod *onResume* i *onPause*, bardziej dopasowanych do cyklu życia aktywności Androida (patrz sekcja 2.1.4.1).

3

Przypadki użycia NFC

Aplikacje NFC są wdrażane w trzech różnych trybach:

- Aplikacje NFC wykorzystujące tryb czytnika/zapisu do odczytu informacji (np. adresu URL, kodu aktywacyjnego);
- Tryb NFC P2P do wymiany informacji (np. wymiana plików komputerowych) lub do parowania dwóch urządzeń NFC (np. wymiana danych połączenia);
- Transakcje aplikacji NFC oparte na trybie emulacji karty NFC i komunikacja z usługami oferującymi interfejsy oparte na standardzie kart inteligentnych (np. dla transakcji płatności mobilnych, e-biletów, identyfikacji lub kontroli dostępu).

W tym rozdziale przedstawiamy konkretne przykłady zastosowania standardu NFC, ilustrujące każdy z trzech trybów działania NFC w klasycznym przypadku użycia. Musimy jednak pamiętać, że NFC można zastosować w każdym sektorze w różnych przypadkach użycia.

3.1. Korzystanie z trybu czytnika/zapisu NFC

Standardowy tryb czytnika/zapisu NFC jest podobny do przypadku użycia kodów QR, z tą zaletą, że użytkownik nie musi wybierać programu ani wyświetlać obrazu, aby uzyskać dostęp do treści, ale po prostu przynieść urządzenie mobilne NFC (tj. działające jako czytnik) do czytnika.

Anne-Marie Lesas i Serge Miranda.

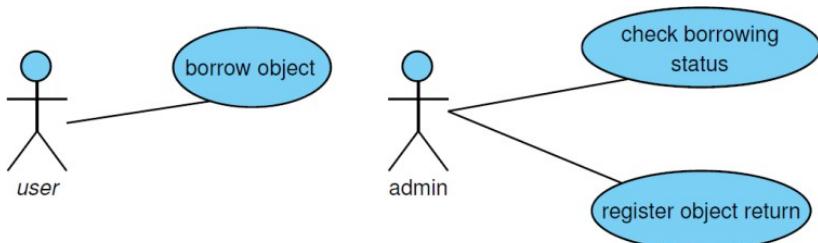
© ISTE Ltd 2017. Opublikowane przez ISTE Ltd i John Wiley & Sons, Inc.

w pobliżu "docelowego" obiektu, który ma zostać odczytany (np. tagu NFC). Powoduje to automatyczne uruchomienie aplikacji mobilnej powiązanej z zawartością tagu (patrz sekcja 2.2.2.1). Obiekt docelowy zawiera komunikat NDEF; może to być dowolny obiekt materialny, na przykład budynek, drzwi, plakat lub książka. Zawartość tagu NFC (np. adres URL i kod) działa jako łącze z informacjami w świecie cyfrowym (np. historia dzieła sztuki w muzeum, przewodnik użytkownika, informacje turystyczne lub o miejscu, video / muzyka itp.) lub jako wyzwalaacz akcji "uruchamianej" po "dotknięciu" (np. w celu otwarcia drzwi, wyłączenia urządzenia lub śledzenia wizyty użytkownika).

Tag NFC jest unikalnym identyfikatorem "oznaczonego" obiektu (tj. ze względu na jego UID), a urządzenie mobilne NFC jest identyfikatorem użytkownika końcowego (tj. IMEI i dane logowania) oraz kontekstu (tj. "tu i teraz": czasowość, lokalizacja itp.). Ta właściwość precyzyjnej kontekstualizacji zdarzenia pozwala na tworzenie zaawansowanych przypadków użycia (np. płatności mobilne i kontrola dostępu).

3.1.1. Przypadek użycia: zarządzanie pożyczkami na sprzęt

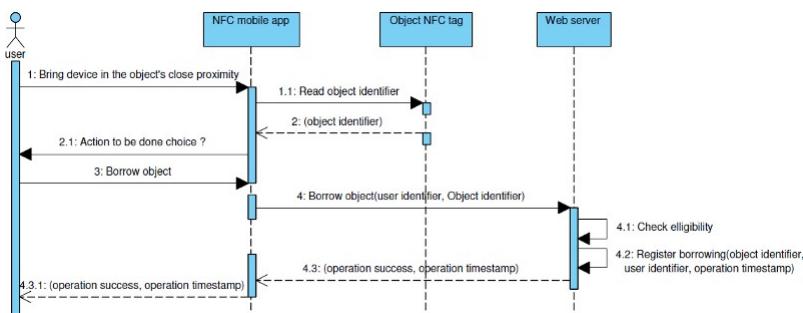
Proponujemy scenariusz zarządzania wypożyczaniem sprzętu przy użyciu trybu czytnika/nadajnika NFC. Przypadek użycia obejmuje użytkownika działającego jako menedżer (zwany tutaj "administratorem") i użytkownika pożyczającego (zwanego tutaj "użytkownikiem") posiadającego smartfon NFC i coś do wypożyczenia; może to być dowolny rodzaj obiektu (np. książka, sprzęt, pojazd współdzielony lub pokój hotelowy) z powiązanym z nim tagiem NFC.



Rysunek 3.1. Przypadek użycia zarządzania pożyczkami

System obejmuje:

- aplikacja mobilna NFC dla użytkownika przeznaczona do rejestracji wypożyczenia obiektu;
- aplikacja mobilna NFC dla administratora przeznaczona do sprawdzania rejestracji wypożyczeń i rejestrowania zwrotów obiektów;
- serwer sieciowy ze sparowanym interfejsem API usług sieciowych i jednostką pamięci (bazą danych) do zarządzania informacjami. Zakładamy, że dostępne będą dwa interfejsy użytkownika (UI): (1) interfejs zarządzania pożyczkami dla administratorów do kontrolowania zwrotów i (2) interfejs użytkownika do sprawdzania historii pożyczek: Ponieważ ten aspekt nie ma żadnego wpływu na przypadek użycia NFC, nie jest tutaj analizowany;
- obiekt komunikacyjny posiadający interfejs typu tag NFC.

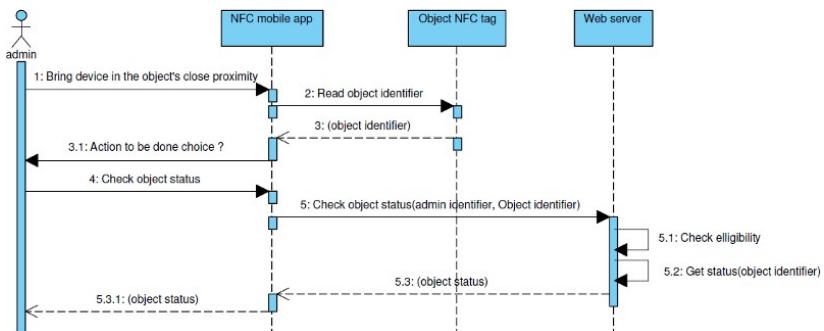


Rysunek 3.2. Pożyczanie obiektów

Rysunek 3.2 przedstawia interakcje między komponentami systemu, gdy użytkownik pożyczca obiekt:

- (1) Użytkownik dotyka interfejsu NFC przedmiotu, który chce pozyczyć, za pomocą swojego telefonu komórkowego NFC;
 - (1.1) aplikacja odczytuje identyfikator obiektu (może to być UID lub kod zakodowany wcześniej na tagu);

- (2.1) aplikacja prosi o wykonanie akcji (w rzeczywistości aplikacja może oferować kilka opcji, takich jak na przykład wyświetlenie informacji o obiekcie lub historii wypożyczeń użytkownika);
- (3) użytkownik wybiera opcję wypożyczenia przedmiotu;
- (4) aplikacja łączy się z serwerem, aby zarejestrować pożyczkę;
- (4.1) serwer sprawdza kwalifikowalność żądania (w rzeczywistości system sprawdza, czy użytkownik może wypożyczyć obiekt i czy obiekt nie został już wypożyczony w innym miejscu);
- (4.2) wypożyczenie jest rejestrowane w bazie danych;
- (4.3) serwer informuje aplikację, że wypożyczenie zostało poprawnie zarejestrowane;
- (4.3.1) aplikacja informuje użytkownika, że wypożyczenie obiektu zostało pomyślnie zarejestrowane.

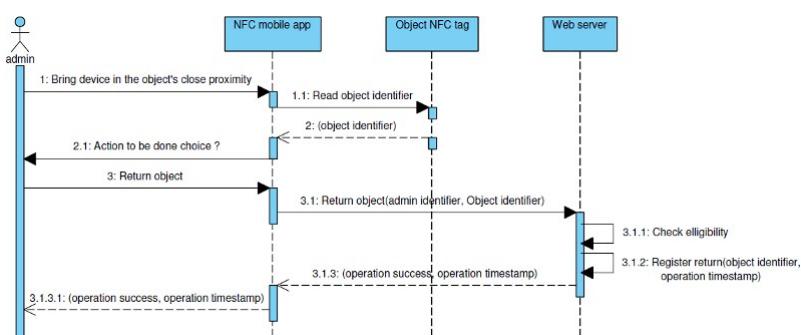


Rysunek 3.3. Sprawdzanie statusu obiektu

Obiekt (wypożyczony lub nie) może zostać sprawdzony w dowolnym momencie przez administratora wypożyczeń (na przykład w celu sprawdzenia, czy użytkownik prawidłowo zarejestrował wypożyczenie), jak pokazano na rysunku 3.3:

- (1) Administrator dotyka interfejsu NFC obiektu, aby sterować nim za pomocą swojego urządzenia mobilnego NFC;
- (2) aplikacja odczytuje identyfikator obiektu;

- (3.1) aplikacja prosi o wykonanie akcji;
 - (4) administrator wybierze opcję sprawdzenia statusu obiektu;
 - (5) aplikacja łączy się z serwerem w celu zebrania danych o obiekcie;
 - (5.1) serwer sprawdza kwalifikowalność żądania;
 - (5.2) serwer zbiera dane na temat stanu obiektu (lub jego historii);
 - (5.3) serwer informuje aplikację, że wypożyczenie zostało dokonane.
- pomyślnie zarejestrowany;
- (5.3.1) aplikacja wyświetla informacje o obiekcie.



Rysunek 3.4. Zwrot obiektu

Zwrot wypożyczonego obiektu jest wykonywany przez administratora wypożyczeń, gdy obiekt zostanie fizycznie zwrócony przez użytkownika (pożyczkobiorcę) (patrz rysunek 3.4):

- (1) Administrator dotyka interfejsu NFC obiektu, aby sterować nim za pomocą swojego telefonu komórkowego NFC;
 - (1.1) aplikacja odczytuje identyfikator obiektu;
 - (2) aplikacja prosi o wykonanie akcji;
 - (3) menedżer wybiera opcję zwrotu obiektu;
 - (3.1) aplikacja łączy się z serwerem w celu zarejestrowania zwrotu obiektu;

- (3.1.1) serwer sprawdza kwalifikowalność żądania; (3.1.2) serwer rejestruje zwrot obiektu;
- (3.1.3) serwer informuje aplikację, że zwrot został pomyślnie zarejestrowany;
- (3.1.3.1) aplikacja informuje użytkownika, że zwrot został pomyślnie zarejestrowany.

Ten przykład pokazuje, jak proste jest przeprowadzanie transakcji za pomocą technologii NFC, nawet bez kosztownego profesjonalnego urządzenia. Siła takiego systemu jest wzmacniana przez włączenie geolokalizacji, zapewnianej przez GPS telefonów komórkowych podczas żądań transakcji; w ten sposób można dodać reguły wnioskowania do sprawdzania kwalifikowalności transakcji i/lub można wdrożyć dodatkowe funkcje związane z identyfikowalnością.

3.2. Korzystanie z trybu NFC P2P

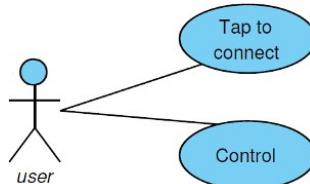
Najprostszy przypadek użycia trybu P2P standardu NFC polega na wymianie danych między dwoma urządzeniami peryferyjnymi NFC. Może to być po prostu udostępnianie multimediów między dwoma użytkownikami, tj. gdy pierwszy użytkownik rozpoczyna wymianę, wybierając wcześniej dokument, który chce udostępnić urządzeniu docelowemu drugiego użytkownika. Następnie obaj użytkownicy zbliżą swoje urządzenia do siebie i nastąpi transfer NFC z urządzenia inicjującego do urządzenia docelowego.

3.2.1. Przypadek użycia: Parowanie NFC

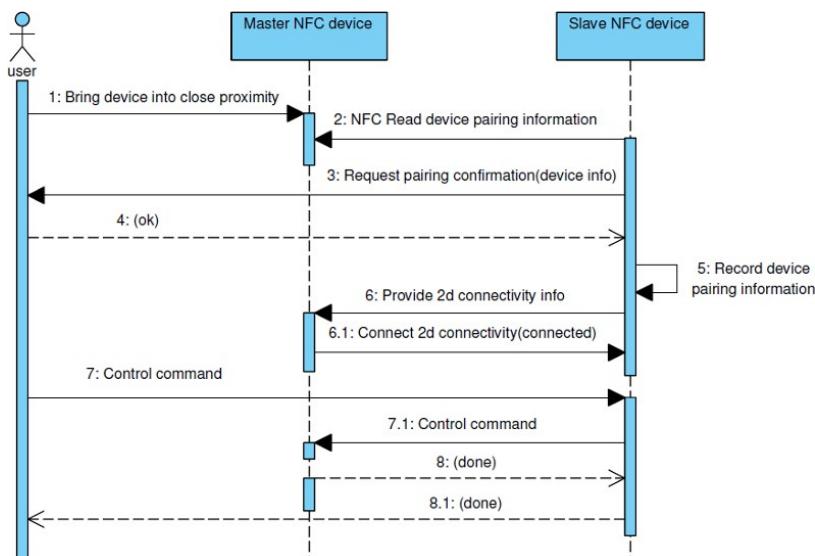
Przypadek użycia trybu NFC P2P omówiony w tej sekcji obejmuje użytkownika posiadającego dwa urządzenia obsługujące NFC z inną szybką łącznością (np. Bluetooth® i Wi-Fi):

- urządzenie "master" posiada interfejs użytkownika (na przykład smartfon);
- urządzenie "podrzędne" nie musi mieć interfejsu użytkownika (na przykład głośnik, ale może to być dowolny obiekt: ekspres do kawy, samochód, dom itp.)

W tym przypadku urządzenie "główne" (smartfon) działa jako "uniwersalny pilot zdalnego sterowania", umożliwiając użytkownikowi łączenie i kontrolowanie "obiektów", a technologia NFC jest "uniwersalnym złączem" do parowania urządzeń z "zerową konfiguracją ręczną".



Rysunek 3.5. Przypadek użycia parowania



Rysunek 3.6. Interakcje scenariusza parowania

Interakcje naszego scenariusza parowania (patrz rysunek 3.6.) pokazują, w jaki sposób można skonfigurować drugą łączność za pomocą NFC:

- (1) użytkownik zbliża urządzenie peryferyjne slave do urządzenia sterującego (master);

(2) urządzenie główne (np. smartfon) odczytuje informacje o parowaniu urządzenia, które ma zostać sparowane dzięki łączności NFC;

(3) Ponieważ jest to pierwszy kontakt, urządzenie nadzędne żąda potwierdzenia od użytkownika, który chce sparować się z urządzeniem podrzędnym (po pierwszym sparowaniu połączy się automatycznie, bez potrzeby potwierdzania);

(4) użytkownik potwierdzi parowanie;

(5) urządzenie nadzędne rejestruje parametry urządzenia podrzędnego w celu późniejszego ich ponownego wykorzystania (np. dane uwierzytelniające i sterujące);

(6) w zależności od informacji o parowaniu, urządzenie nadzędne wysyła informacje, aby umożliwić urządzeniu podrzędnemu korzystanie z drugiej łączności (ta część jest do oceny dewelopera za pomocą forum standardu NFC wspomnianego w sekcji 1.3.4 i zależy od interfejsu API dostępnego na platformie programistycznej);

(6.1) urządzenie podrzęenne łączy się z urządzeniem nadzędnym za pośrednictwem drugiego połączenia;

(7) Użytkownik może następnie wysyłać polecenia sterujące do urządzenia podrzędnego za pośrednictwem interfejsu użytkownika urządzenia nadzędnego;

(8) urządzenie główne wysyła polecenia do urządzenia podrzędnego zgodnie z wzorcem dostarczonym przez zebrane dane podczas inicjowania parowania (z NFC), za pośrednictwem drugiej łączności;

(8.1) urządzenie podrzęenne powiadamia urządzenie nadzędne o pomyślnym wykonaniu polecenia; użytkownik jest powiadamiany przez urządzenie nadzędne.

3.3. Korzystanie z emulacji karty NFC tryb

Tryb emulacji karty NFC charakteryzuje się warstwą APDU (protokół kart inteligentnych i platformy JavaCardTM). Technologia NFC jest zatem bezstykową warstwą transportową dla usług osadzonych w chipie zwanym "bezpiecznym elementem": SE (patrz sekcja 1.3.2.2). Gdy SE jest zintegrowany z urządzeniem mobilnym (tj. w smartfonie lub tablecie), do komunikacji z SE niezbędny jest interfejs API dostępny z systemu operacyjnego telefonu komórkowego. Ten interfejs API nie zawsze jest dostępny dla

programistów. Na platformie Android (patrz sekcja 2.2.4), OMAPI

(zob. sekcja 1.3.6) umożliwia komunikację z SE w trzech formach sprzętowych (SIM, eSE i microSD), niezależnie od konfiguracji, za pośrednictwem wewnętrznego czytnika (zob. sekcja 2.2.4.3.2). Teoretycznie wszystkie istniejące usługi oparte na karcie intelligentnej (np. karty kredytowe, karty transportowe, karty identyfikacyjne, karty abonamentowe i karty ubezpieczenia zdrowotnego) mogą być hostowane w SE połączonym wewnętrznie z aplikacją korzystającą z zaawansowanych funkcji urządzenia mobilnego.

Aplikacja w trybie emulacji karty NFC składa się z usługi na karcie (z ograniczonym dostępem) działającej w SE oraz aplikacji uruchomionej w systemie operacyjnym urządzenia mobilnego.

Dla przypadku użycia podanego jako przykład w sekcji 3.3.1, zajmujemy się scenariuszem portfela cyfrowego.

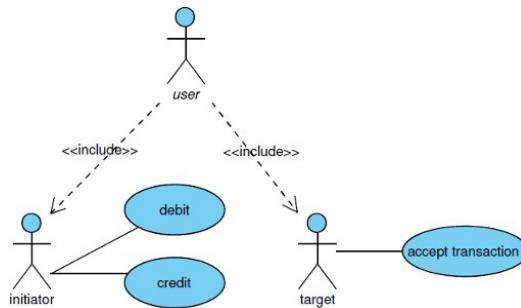
3.3.1. Przypadek użycia: portfel cyfrowy w SE

Przedstawiamy oryginalny przypadek użycia¹ portfela cyfrowego jako przykład dla trybu emulacji karty NFC. Portfel cyfrowy może działać jako alternatywna wirtualna waluta dla transakcji społecznościowych; może więc być walutą barterową między dwiema osobami w społeczności lub dedykowaną walutą wirtualizującą bony usługowe lub punkty lojalnościowe (np. bony na posiłki, bony wakacyjne, bony podarunkowe), które są cenne tylko w określonym kontekście.

System obejmuje dwóch użytkowników wyposażonych w urządzenie NFC, działających odpowiednio jako inicjator i cel transakcji debetowej/kredytowej (patrz rysunek 3.7), gdzie użytkownik docelowy autoryzuje transakcję poprzez zbliżenie i "dotknietcie" telefonu komórkowego inicjatora. Portfel cyfrowy użytkownika płacącego zostanie następnie obciążony, a portfel użytkownika obciążającego otrzyma tę samą kwotę. Zakładamy zerowe saldo początkowe i możliwość ujemnego salda (tj. w zrównoważonym systemie gospodarczym waga musi równoważyć się z transakcjami). System zawiera aplikację działającą jako interfejs użytkownika do zarządzania

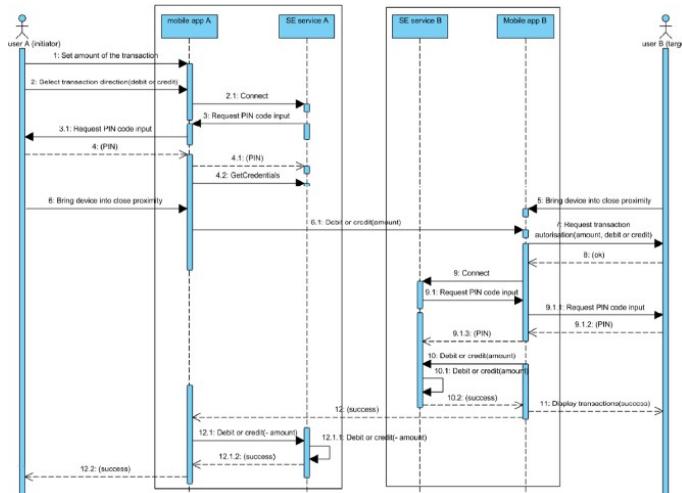
¹ Wszelkie prawa własności intelektualnej zastrzeżone dla autora.

transakcje debetowe/kredytowe łączące się z usługą portfela cyfrowego hostowaną w SE.



Rysunek 3.7. Przypadek użycia portfela cyfrowego

UWAGA: - W tym przykładzie zapomnieliśmy o bardziej złożonych aspektach zarządzania cyklem życia usługi SE na karcie, która jest znormalizowana i dobrze udokumentowana przez GlobalPlatform (patrz sekcja 1.3.5).



Rysunek 3.8. Scenariusz interakcji portfela cyfrowego z SE

Rysunek 3.8 przedstawia interakcje między dwoma urządzeniami mobilnymi NFC podczas transakcji debetowej/kredytowej:

- (1) użytkownik A (inicjator) wprowadza kwotę transakcji;
- (2) użytkownik A wprowadza kierunek transakcji (debit lub kredyt); (2.1) aplikacja mobilna użytkownika A łączy się w emulacji karty NFC z usługą portfela cyfrowego hostowaną w SE telefonu komórkowego użytkownika A za pośrednictwem czytnika wewnętrznego (tj. za pośrednictwem interfejsu API komunikacji SE);
 - (3) SE A wysyła żądanie uwierzytelnienia za pomocą kodu PIN; (3.1) aplikacja mobilna A prosi użytkownika A o wprowadzenie kodu PIN;
 - (4), (4.1) użytkownik A wprowadza swój kod PIN, który jest przesyłany do usługi SE A;
 - (5)(6) użytkownicy A i B zbliżają swoje urządzenia mobilne do siebie;
 - (6.1) Aplikacja mobilna A wysyła żądanie obciążenia/kredytu do aplikacji mobilnej B za pośrednictwem trybu NFC P2P;
 - (7) aplikacja mobilna B prosi użytkownika B o autoryzację transakcji kredytowej/debetowej;
 - (8) użytkownik B autoryzuje transakcję;
 - (9) aplikacja mobilna B łączy się w trybie emulacji karty NFC z usługą portfela cyfrowego hostowaną w SE B;
 - (9.1) SE B odsyła żądanie identyfikacji za pomocą kodu PIN; (9.1.1) aplikacja mobilna B prosi użytkownika B o wprowadzenie kodu PIN;
 - (9.1.2), (9.1.3) użytkownik B wprowadza swój kod PIN, który jest wysyłany do serwisu SE B;
 - (10) aplikacja mobilna B wysyła z powrotem polecenie obciążenia/kredytowania do usługi SE B;
 - (10.1) usługa portfela cyfrowego SE B rejestruje transakcję debetową/kredytową;
 - (10.2), (11) usługa SE B powiadamia, że transakcja debetowa/kredytowa została pomyślnie przetworzona, o czym aplikacja mobilna powiadamia użytkownika B;

(12) aplikacja mobilna B powiadamia aplikację mobilną A, że transakcja powiodła się przy użyciu trybu NFC P2P (w tym momencie oba urządzenia m u s zą s ię zetknąć);

(12.1), (12.1.1) aplikacja mobilna A wysyła żądanie odwrócenia płatności (obciążenie/kredyt) tej samej kwoty (tj. znaku odwrotnego) do usługi portfela cyfrowego SE A, która rejestruje transakcję;

(12.1.2), (12.2) usługa SE A powiadamia aplikację mobilną A o powodzeniu transakcji, o czym powiadamiany jest użytkownik A.

UWAGA - W naszym oryginalnym przykładzie użycia trybu emulacji karty NFC usługa SE nie jest bezpośrednio dostępna z zewnątrz. Proponowana przez nas konfiguracja jest jednak w pełni dostosowana do transakcji przetwarzanych za pomocą terminala podłączonego do bezstykowego czytnika NFC.

3.4. Korzystanie z trybu HCE

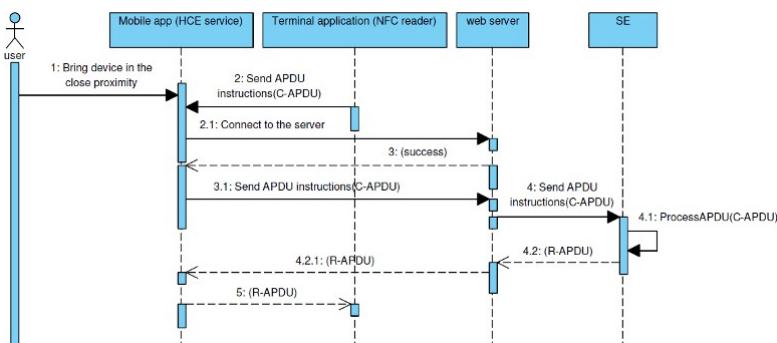
Tryb emulacji karty opartej na hoście (HCE) jest szczególnym przypadkiem trybu emulacji karty NFC: usługa SE na karcie jest zastąpiona usługą działającą w tle w systemie operacyjnym urządzenia mobilnego, która ma interfejs zdolny do zarządzania APDU (patrz sekcja 2.2.5). Tryb HCE nie wymaga specjalnego interfejsu API do komunikacji z SE (ponieważ nie jest zaangażowany SE) i korzysta z dostępu do zaawansowanych funkcji urządzenia mobilnego (w przeciwieństwie do usług SE działających w ograniczonym środowisku mikroukładu karty inteligentnej). Co więcej, zarządzanie cyklem życia usługi HCE jest takie samo jak zarządzanie zwykłą aplikacją mobilną, co znacznie upraszcza jej implementację i nie wymaga współpracy z osobami trzecimi (tj. TSM/SEI).

Przypadek użycia trybu emulacji karty NFC implementującego SE zaproponowany w poprzedniej sekcji (patrz sekcja 3.3.1) można również wdrożyć w ten sam sposób z usługą HCE zamiast usługi SE na karcie. W przykładzie użycia HCE zaproponowanym w sekcji 3.4.1, usługa HCE działa jako brama dla SE dostępnego zdalnie w chmurze.

3.4.1. Przypadek użycia: SE w chmurze z HCE

Nasz scenariusz wykorzystania trybu HCE opisuje interakcje przypadku użycia, w którym mobilna usługa HCE odbiera instrukcje APDU z terminalowego czytnika NFC i wysyła je do zdalnej usługi SE hostowanej w chmurze:

- (1) użytkownik zbliża swoje urządzenie mobilne NFC do czytnika NFC;
- (2) aplikacja terminalowa połączona z czytnikiem NFC wysyła instrukcje APDU do aplikacji mobilnej (odebrane przez usługę HCE);
- (2.1), (3) aplikacja mobilna łączy się ze zdalnym serwerem internetowym;
- (3.1) aplikacja mobilna przekazuje instrukcję APDU (lub sekwencję instrukcji) do serwera WWW;
- (4) serwer WWW wysyła instrukcję (C-APDU) do hostowanej usługi SE;
- (4.1), (4.2) usługa SE przetwarza instrukcje i zwraca odpowiedź (R-APDU);
- (4.2.1) serwer WWW zwraca odpowiedź APDU do aplikacji mobilnej;
- (5) aplikacja mobilna zwraca odpowiedź APDU do aplikacji terminalowej.



Rysunek 3.9. Scenariusz interakcji SE w chmurze z HCE

UWAGA: - To rozwiązanie przypadku użycia SE w chmurze przy użyciu HCE wymaga łączności z Internetem.

Wnioski

Smartfony wyposażone w standard NFC, będący połączeniem technologii RFID i kart inteligentnych z wymogiem bliskiej odległości, są potężnymi inicjatorami interakcji między obiektami materiałnymi, stając się komunikacją i usługami cyfrowymi. Trzy tryby NFC obejmują szeroką gamę przypadków użycia z nieograniczonimi zastosowaniami i nieograniczoną wyobraźnią.

Za pomocą szybkiego i prostego gestu zbliżeniowego (3 "S" i "Tap'n Play", patrz Wprowadzenie), użytkownik może uzyskać dostęp do geolokalizowanych, spersonalizowanych, kontekstowych i multimedialnych informacji znajdujących zastosowanie w turystyce, rekreacji, kulturze, technologii, edukacji, handlu itp. Przykłady te nie są organiczne, ponieważ zdarzenie NFC charakteryzuje się przestrenną czasowością ("tu i teraz") i zidentyfikowanymi komponentami (takimi jak tag identyfikowany przez jego UID lub jego zawartość, lub posiadacz urządzenia mobilnego identyfikowany przez jego IMEI) w trybie czytnika / zapisu NFC.

W trybie P2P, NFC osiąga nowy wymiar i pozwala na sparowanie dwóch urządzeń, które następnie będą w stanie wymieniać dane (np. Wi-Fi, Bluetooth® i Li-Fi), gdy inicjator zdecyduje się to zrobić (jego wola jest indukowana w fakcie, że zbliża dwa urządzenia do siebie, z lub bez konfiguracji podczas pierwszego kontaktu).

W trybie emulacji karty, NFC nie tylko pozwala zabezpieczyć transakcje poprzez implementację protokołów i kryptograficznych

Techniki odziedziczone po technologiach kart inteligentnych, ale wnoszą również znaczną wartość dodaną do statycznych usług hostowanych na kartach inteligentnych, które nie są zbyt ergonomiczne i efemeryczne (np. karty kredytowe / debetowe, karty lojalnościowe, karty ubezpieczenia zdrowotnego, karty transportowe, karty dostępu, karty identyfikacyjne i inne dokumenty cyfrowe); wszystkie usługi kart inteligentnych mogą zostać zdematerializowane w SE i korzystać z łatwości obsługi, mobilności i inteligencji smartfonów.

W rozdziale 1 poświęconym najnowocześniejszym rozwiązańom NFC omówiliśmy standard NFC znaryzuowany przez forum NFC od 2004 r., który może być stosowany w wielu sektorach. Jednak w najbardziej zauważalnym zastosowaniu, wdrożenie bezpiecznych usług NFC (w trybie emulacji karty) wymaga zbieżnej interoperacyjności liderów branży w heterogenicznych sektorach (dostawcy usług, informatyka, producenci, telekomunikacja, banki i finanse, rządy...); rynek rozwija się w kierunku standaryzacji bezpiecznych interfejsów zarządzania skoncentrowanych przez organizację GlobalPlatform wokół złożonego ekosystemu.

W rozdziale 2 szczegółowo przedstawiliśmy wprowadzenie do programowania mobilnego NFC z systemem Android w trzech trybach NFC. Przykłady kodowania pomogą deweloperowi w implementacji własnych aplikacji NFC na urządzeniu z systemem Android wyposażonym w technologię NFC.

W rozdziale 3 omówiliśmy przykłady konkretnych przypadków użycia w trzech trybach NFC:

- Zilustrowaliśmy tryb czytnika / zapisu NFC scenariuszem NFC zastosowanym do wypożyczania obiektów;
- Dla trybu P2P zaproponowaliśmy scenariusz parowania między dwoma urządzeniami NFC;
- Tryb emulacji karty w standardzie NFC został zilustrowany przykładem cyfrowego portfela z dwoma przypadkami użycia:
 - przy użyciu SE (tj. z usługą uruchomioną w SE),
 - przy użyciu trybu HCE (tj. z usługą działającą w systemie urządzenia mobilnego i portfelem zarządzanym w chmurze).

Podsumowując, niniejsza książka zapewnia wszystkie podstawy do dobrego zrozumienia tworzenia aplikacji NFC (z systemem Android) i daje czytelnikowi pełną autonomię.

Teraz Twoja kolej na oferowanie innowacyjnych usług dzięki standardowi NFC.

Bibliografia

- [AHS 12] AHSON S.A., ILYAS M., *Near Field Communication Handbook*, CRC Press, Boca Raton, 2012.
- [AIL 07] AILISTO H., MATINMIKKO T., HÄIKIÖ J. et al., *Physical Browsing with NFC Technology*, VTT Tiedotteita, Finlandia, 2007.
- [ATT 13] ATTOUR A., DELLA PERUTA M., Le rôle des connaissances architecturales dans l'élaboration de la plateforme technologique d'un écosystème en émergence : le cas des plateformes NFC, GREDEG Working Papers Series, Francja, 2013.
- [CHA 10] CHAIX L., TORRE D., "Different models for mobile payment", European Research Group, Monnaie Banque Finance, Bordeaux, Francja, 2010.
- [COS 12] COSKUM V., OK K., OZDENIZCI B., *Near Field Communication, From Theory to Practice*, Wiley, 2012.
- [COS 13] COSKUM V., OK K., OZDENIZCI B., *Professional NFC Development for Android*, Wiley, 2013.
- [FIN 10] FINKENZELLER K., *RFID Handbook: Fundamentals and Applications in Contactless Smart Cards, Radio Frequency Identification and Near-Field Communication*, 3rd ed., Wiley, 2010.
- [HAJ 09] HAJJI F., ZIAD A., DE LAZZARI T. et al., "A multi-touch NFC virtual poster platform for m-learning: Al Andalus project", *M-learning IADIS International Conference*, Barcelona, Hiszpania, luty 2009.

- [IGO 14] IGOE T., COLEMAN D., JEPSON B., *Rozpoczęcie NFC, Near Field Communication with Arduino, Android, and PhoneGap*, O'Reilly, 2014.
- [JIT 13] JITHESH S., ANOOP N., NAVIN N. et al., *A Comprehensive Guide to Enterprise Mobility*, CRC Press, Boca Raton, 2013.
- [KNU 68] KNUTH D.E., *The Art of Computer Programming, Fundamental Algorithms*, vol. 1, Addison-Wesley, 1968.
- [LES 14a] LESAS A.M., MIRANDA S., RENAUT B. et al., "WOLF: a research Platform to write NFC secure applications on top of multiple Secure Elements", *International Journal of Advanced Computer Science and Applications*, vol. 5, nr 8, s. 20-31, 2014.
- [LES 14b] LESAS A.M., "A fast implementation of TSM web services within WOLF API for F1RST android mobile-NFC services", University of Nice Sophia Antipolis, Francja, 2014.
- [LES 14c] LESAS A.M., "F1RST research project mobiquitous NFC financial services for unbanked people", *Konferencja WIMA*, Monako, kwiecień 2014.
- [LES 16] LESAS A.M., "Détecer et moniturer les séismes grâce aux capteurs embarqués dans les smartphones", *INFomatique des ORganisation et Systèmes d'Information et de Décision (INFORSID)*, Grenoble, Francja, maj 2016.
- [MIR 09] MIRANDA S., "Robotics platform integrating wireless communicating objects for assisted living (RoboDOMO project)", *Keynote WASM Conference*, Oulu, Finlandia, listopad 2009.
- [MIR 11a] MIRANDA S., LESAS A.M., "Architecture logicielle du projet VAMP: Plate-forme mobiquitaire embarquée à bord de véhicules automobiles", *Génie Logiciel*, vol. 103, pp. 38-48, 2011.
- [MIR 11b] MIRANDA S., PASTORELLY N., ISHKINA E. et al., "Lessons inferred from NFC mobiquitous innovative information service prototyping at UNS", w *Ingénierie des systèmes d'information*, vol. 16, no. 4, pp. 49-62, 2011
- [MIR 14a] MIRANDA S., PAPETTI C., "Les nouveaux paradigmes du tourisme mobiquitaire", w CHRISTOFLE S. (red.), *Tourisme, destinations et entreprises. Leadership dans le luxe, l'"évenementiel" et les TIC*, Mondes du tourisme, France, 2014.

- [MIR 14b] MIRANDA S., PAPETTI C., SOK K.H. *et al.*, "Une application de gestion de tags pour le tourisme mobiquitaire: application au projet Thom au Cambodge", w SPINDLER J., CLERGEAU C. (red.), *l'Immatériel Touristique*, L'Harmattan, Francja, 2014.
- [MIR 14c] MIRANDA S., "Homo mobiquitus, communacteur sur les territoires", w CARMES M., NOYER J.M. (red.), *Devenirs Urbains*, Presse des Mines, Francja, 2014.
- [MIR 16] MIRANDA S., "BIG DATA et Innovation Spiraliste (Big Data and Connected Objects II)", *7th International Research Meeting in Business and Management (IRMBAM-2016)*, Nicea, Francja, 2016.
- [NAR 11] NARNI-MANCINELLI G., BENOUALI H., LEITZELMAN M. *et al.*, "MBDS2.0, plateforme générique de gestion de tags NFC et 2D pour des espaces culturels intelligents et communautaires 2.0", *Ingénierie des systèmes d'information*, vol. 16, no. 4, pp. 49-62, 2011.
- [PAR 12] PARET D., BOUTONNIER X., HOUITI Y., *NFC Near Field Communication, Principes et applications de la communication en champ propre*, Dunod, Paris, 2012.
- [PAS 11] PASTORELLY N., BENOUALI H., LEBLANC C. *et al.*, "Nice Future Campus, pakiet usług NFC w wirtualnej karcie studenta", *Ingénierie des systèmes d'information*, vol. 16, no. 4, pp. 64-86, 2011.
- [TUI 09] TUIKKA T., ISOMURSU M., *Touch the Future with a Smart Touch*, VTT Tiedotteita, Finlandia, 2009.

Indeks

A, B, C

kontrola dostępu, 8, 9, 12, 23, 38, 40, 107, 108
aktywny, 6, 7, 9, 18, 25, 28, 45
analogowy, 18-20, 31
Android, 45
 Zestaw narzędzi rozwojowych (ADT), 46, 50-53, 56, 68, 70
Aplet, 24, 87, 89, 90, 101
 Identyfikator (AID), 88, 99,
100 Jednostka danych protokołu aplikacji (APDU), 35, 38, 40, 87, 88, 97, 98, 114, 118, 119
asymetryczny, 11, 12
ATAWAD, 2
autoryzacja, 9, 12, 41, 117
tryb autoryzowany (podwójny), 41 big data, 2, 5
BroadcastReceiver, karta 102-104
 emulacja, 22-25, 31, 36, 45, 89, 91, 92, 97, 99, 100, 107, 114-118
 specyfikacje, 23, 36
Cloud, 24, 118-120
złącze, 15, 113

D, E, G

tryb delegowany, 41
zdematerializowane, 122
specyfikacje urządzenia, 36
ECMA-340, 17
ekosystem, 1, 2, 4, 16, 36, 40, 41
Europary MasterCard Visa (EMV), 11, 23, 25
geolokalizacja, 5, 17, 112
GlobalPlatform (GP), 24, 36-42, 116
Google Play, 47, 59

H, I, J

emulacja karty hosta (HCE), 24, 97-103, 118, 119, 120
identyfikacja, 1, 8, 10, 13, 24, 107, 115, 117
 wdrożenie, 28, 29, 37, 61, 85, 90, 92, 97, 101-103, 118
system informacyjny, 13, 99
inicjator, 17, 18, 21, 25, 28, 31, 45, 112, 115, 117

integralność, 9, 10, 13
interakcja, 3, 17, 37, 61, 109,
113, 116, 119
interoperacyjność, 16,
36 ISO/IEC
14443, 17, 26, 28, 29, 31, 35
15693, 19, 28
18092, 17, 26, 28, 29, 31
7816, 23, 35
IsoDep, 89, 91, 97
JavaCard™, 23, 24, 87, 101, 102,
114
JIS X 6319-4, 35

L, M, N

cykl życia, 11, 38, 40-42, 61, 66,
90, 99, 105, 116, 118
lokalizacja, 49, 65, 102, 108
protokół kontroli łączna
logicznego
(LLCP), 25, 29, 30, 45
manifest, 56, 58-60, 64, 71, 73-
77, 84, 90, 100, 104
mobilny, 2, 3, 5, 13, 16
mobilność, 2-4, 122
modulacja, 17-20, 28, 31
Płatności mobilne, 5, 13, 15, 107,
108 NFC
format wymiany danych
(NDEF), 22, 31
Forum, 17, 19, 22, 25, 26-30,
33-36
NFC-A, 18, 21, 31
NFC-B, 18, 31
NFC-C, 31
NFC-F, 19
NFCIP-1, 27, 29
NFCIP-2, 28
NFC-V, 28
N-Mark, 27

O, P, R

Open Mobile API (OMAPI), 42, 92-
95
parowanie, 1, 27, 36, 112-114, 122
pasywny, 1, 7, 8, 17, 18, 21, 25, 27,
28, 45, 71
peer-to-peer (P2P), 6,
klucz prywatny, 10, 12, 91
programowanie,
46 publiczne
klucz, 10-12
kluczowa infrastruktura, 10
identyfikacja radiowa (RFID), 1
czytelnik/pisarz, 31, 45, 71-75, 84,
107-112
definicja typu rekordu (RTD), 22,
36
routing, 98-100

S, T

bezpieczny element, 9, 23, 114
bezpieczeństwo, 9-11, 36-38, 42, 47,
90,
99, 102
serwer, 3, 6, 22, 40, 102, 109-112,
119
sygnał, 7, 15, 18-20, 28, 31
SIMAlliance, 42-43
proste
tryb, 41
Protokół wymiany NDEF
(SNEP), 31
karta inteligentna, 8-12, 37, 87, 107
standaryzacja, 16, 24, 25, 36
symetryczny, 10-12
zaufane środowisko wykonawcze
(TEE), 39

identyfikowalność, 1, 5, 8,
38, 112
transmedia, 5
zaufany menedżer usług
(TSM), 39-42
typ
 1 znacznik, 35
 2 tagi, 35
 3 znaczniki, 27, 35
 4 tagi, 35
format nazwy (TNF), 33

U, V

unikalny identyfikator (UID), 8, 79,
109
przypadek użycia, 107-112, 115, 119
wirtualne pieniądze, 4

Inne tytuły od



w

Systemy informatyczne, sieć i przetwarzanie
wszechobecne

2016

BEN CHOUIKHA Mona

Projektowanie organizacyjne dla zarządzania wiedzą

BERTOLO David

*Interakcje na tabletach cyfrowych w kontekście nauki geometrii 3D
(Zestaw interakcji człowiek-maszyna - tom 2)*

BOUVARD Patricia, SUZANNE Hervé

Rozwój inteligencji zbiorowej w biznesie

EL FALLAH SEGHROUCHNI Amal, ISHIKAWA Fuyuki, HÉRAULT Laurent,

TOKUDA Hideyuki

Czynniki wspomagające inteligentne miasta

FABRE Renaud, we współpracy z MESSERSCHMIDT-MARIET Quentin, HOLVOET Margot

Nowe wyzwania dla wiedzy

GAUDIELLO Ilaria, ZIBETTI Elisabetta *Nauka robotyki, z robotyką, przez robotykę (Zestaw interakcji człowiek-maszyna - tom 3)*

HENROTIN Joseph

Sztuka wojny w erze sieci (zestaw technologii intelektualnych - tom 1)

KITAJIMA Munéo

Pamięć i wybór działań w interakcji człowiek-maszyna (Zestaw interakcji człowiek-maszyna - tom 1)

LAGRAÑA Fernando

E-mail i zmiany zachowań: Wykorzystanie i nadużywanie komunikacji elektronicznej

LEIGNEL Jean-Louis, UNGARO Thierry, STAAR Adrien

Transformacja cyfrowa

MONINO Jean-Louis, SEDKAOUI Soraya

Big Data, otwarte dane i rozwój danych (zestaw inteligentnych innowacji - tom 3)

NOYER Jean-Max

Transformacja inteligencji zbiorowej (zestaw technologii intelektualnych - tom 2)

VENTRE Daniel

Wojna informacyjna - wydanie 2nd

VITALIS André

Niepewna rewolucja cyfrowa

2015

ARDUIN Pierre-Emmanuel, GRUNDSTEIN Michel,

ROSENTHAL-SABROUX Camille

System informacji i wiedzy

(Advances in Information Systems Set - Volume 2)

BÉRANGER Jérôme

Etyka medycznych systemów informacyjnych

BRONNER Gérald

Asymetria wiary i niewiary w Internecie

IAFRATE Fernando

Od Big Data do Smart Data

(Advances in Information Systems Set - Volume 1)

KRICHEN Saoussen, BEN JOUIDA Sihem

Zarządzanie łańcuchem dostaw i jego zastosowania w informatyce

NEGRE Elsa

*Systemy informacyjne i rekomendujące (Advances
in Information Systems Set - Volume 4)*

POMEROL Jean-Charles, EPELBOIN Yves, THOURY Claire

MOOCs

SALLES Maryse

*Podejmowanie decyzji a system informacyjny (Advances in Information
Systems Set - Volume 3)*

SAMARA Tarek

*ERP i systemy informatyczne: Integracja czy dezintegracja
(Advances in Information Systems Set - Volume 5)*

2014

DINET Jérôme

Wyszukiwanie informacji w środowiskach cyfrowych

HÉNO Raphaële, CHANDELIER Laure

Modelowanie 3D budynków: Outstanding Sites

KEMBELLEC Gérald, CHARTRON Ghislaine, SALEH Imad

Systemy rekomendujące

MATHIAN Hélène, SANDERS Lena

Podejścia przestrzenno-czasowe: Obiekty geograficzne i proces zmian

PLANTIN Jean-Christophe

Mapowanie partyacyjne

VENTRE Daniel

Chińskie bezpieczeństwo cybernetyczne i obrona

2013

BERNIK Igor

Cyberprzestępcość i cyberwojna

CAPET Philippe, DELAVALLADE Thomas

Ocena informacji

LEBRATY Jean-Fabrice, LOBRE-LEBRATY Katia

Crowdsourcing: O krok dalej

SALLABERRY Christian

Wyszukiwanie informacji geograficznych w korpusach tekstowych

2012

BUCHER Bénédicte, LE BER Florence

Innowacyjny rozwój oprogramowania w GIS

GAUSSIER Eric, YVON François

Dostęp do informacji tekstowych

STOCKINGER Peter

Archiwa audiowizualne: Cyfrowa analiza tekstu i dyskursu

VENTRE Daniel

Konflikt cybernetyczny

2011

BANOS Arnaud, THÉVENIN Thomas

Informacje geograficzne i systemy transportu miejskiego

DAUPHINÉ André

Geografia fraktalna

LEMBERGER Pirmin, MOREL Mederic

Zarządzanie złożonością systemów informatycznych

STOCKINGER Peter

Wprowadzenie do archiwów audiowizualnych

STOCKINGER Peter

Cyfrowe archiwa audiowizualne

VENTRE Daniel

Cyberwojna i wojna informacyjna

2010

BONNET Pierre

Zarządzanie danymi przedsiębiorstwa

BRUNET Roger

Zrównoważona geografia

CARREGA Pierre

Informacje geograficzne i klimatologia

CAUVIN Colette, ESCOBAR Francisco, SERRADJ Aziz

Kartografia tematyczna - seria 3 tomów

Kartografia tematyczna i transformacje - tom 1

Kartografia i wpływ rewolucji ilościowej - tom 2 Nowe podejścia w kartografii tematycznej - tom 3

LANGLOIS Patrice

Symulacja złożonych systemów w GIS

MATHIS Philippe

Grafy i sieci - wydanie 2

THERIAULT Marius, DES ROSIERS François

Modelowanie dynamiki miejskiej

2009

BONNET Pierre, DETAVERNIER Jean-Michel, VAUQUIER Dominique

Zrównoważona architektura IT: progresywny sposób modernizacji systemów informatycznych za pomocą SOA

PAPY Fabrice

Nauka o informacji

RIVARD François, ABOU HARB Georges, MERET Philippe

Poprzeczny system informacyjny

ROCHE Stéphane, CARON Claude

Organizacyjne aspekty GIS

2008

BRUGNOT Gérard

Przestrzenne zarządzanie ryzykiem

FINKE Gerd

Badania operacyjne i sieci

GUERMOND Yves

Proces modelowania w geografii

KANEVSKI Michael

Zaawansowane mapowanie danych środowiskowych

MANOUVRIER Bernard, LAURENT Ménard

Integracja aplikacji: EAI, B2B, BPM i SOA

PAPY Fabrice

Biblioteki cyfrowe

2007

DOBESCH Hartwig, DUMOLARD Pierre, DYRAS Izabela

Interpolacja przestrzenna danych klimatycznych

SANDERS Lena

Modele w analizie przestrzennej

2006

CLIQUET Gérard

Geomarketing

CORNIOU Jean-Pierre

Patrząc wstecz i idąc naprzód w IT

DEVILLERS Rodolphe, JEANSOULIN Robert

Podstawy jakości danych przestrzennych