



ARCHITECTURE & DEVELOPMENT OF NFC APPLICATIONS

MOBILE JAVA DEVELOPMENT, JAVA CARD, USIM AND TOUCH-BASED SERVICES

Amosse Edouard

Contexte : Mobile phone as a wallet



NFC Near Field Communication



SERVICES
Payment
Access
Ticketing ...

Receive information

Produce information

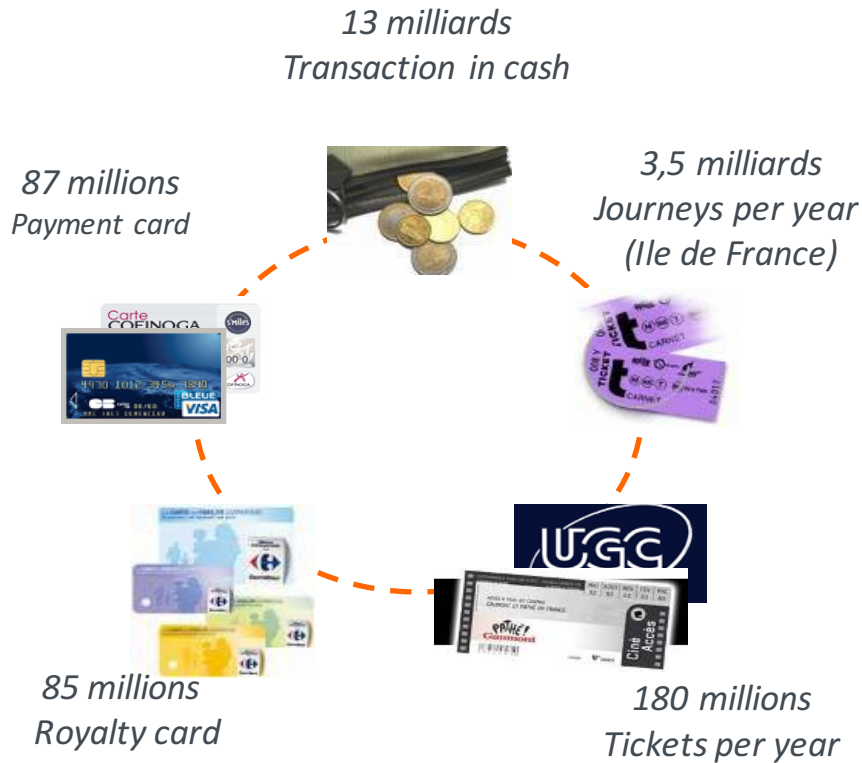


NEW SERVICES
↓
NEW USE CASES

Simplify the daily life

Today : plastic cards, piece of paper

Tomorrow : everything will be in one place



Statistic in France



Ticketing

mobile is digital, targeted and personal

Read and seek
valuable offers



**50% reduction for
girl students at the
star light Dance
Club**



Present

VS.

Receive
personalized
offers 😊



**Come & see us:
Get 10% off ladies
bags until
tomorrow**

Future

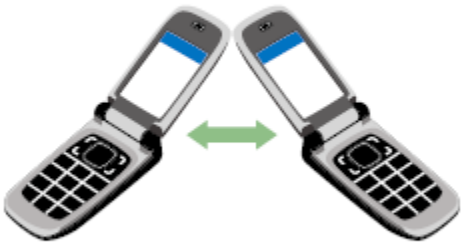
NFC Use cases

Ticketing



Your phone is your travel card

Sharing



Share files between phones

Touch to connect

Service discovery



Get information by touching smart posters

Payment



Your phone is your credit card

Use Case (1)



- **Location based services**
- List of proximity services depending on Points of Interest
- Trailers
- Tickets booking

From SMS push to Smart Poster « pull »

Specifications

NFC Forum releases specification for **NDEF**.

NFC Data Exchange Format which is a way to « format » RFID tags to be compatible with NFC applications.

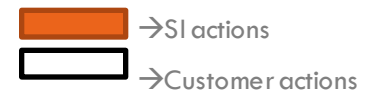
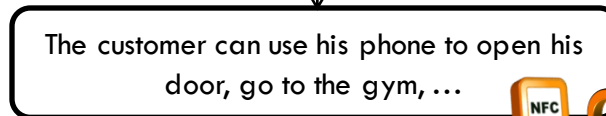
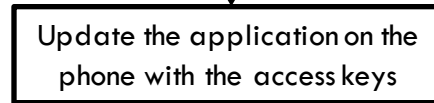
Works with MIME type.

Use case (2)

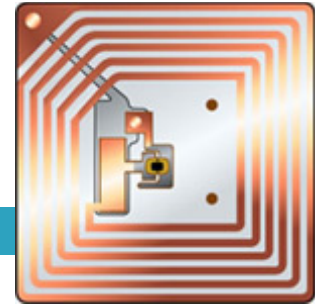


Install and personalize the Mobile Travel Wallet on the client phone

At the hotel

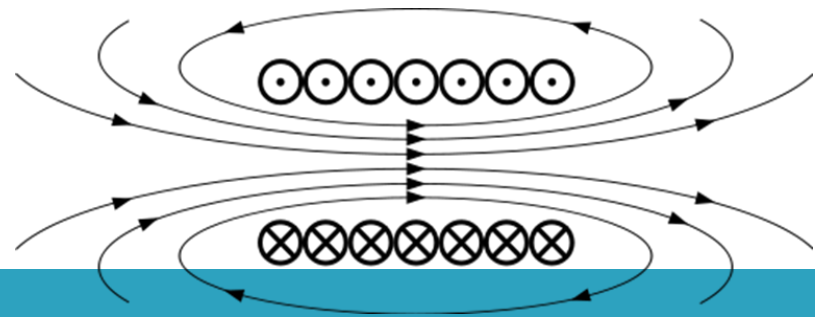


RFID



- RFID : Radio Frequency Identification
- RFID Tags: Store and retrieve data (with a distant reader)
- History : radar technology, cow identification (year 1970).
- Use case examples: road taxes, trace books in libraires, access card, shops (Wall-Mart).
- RFID tags types
 - ▣ Active
 - ▣ Passive (without battery)

From RFID to NFC



- Can communicate with objects
- Magnetic field induction
- Contactless technology based on RFID 13,56MHz
- NFC is standardized ECMA-340 and ISO/IEC 18092
- Backward compatibility with ISO14443 and SmartCard
 - ▣ Millions of readers
 - ▣ Easy to use

Contactless Cards

- FELICA (sony) encryption key generated dynamically at each auth.
- Topaz Tag Innovision
- MIFARE Standard:
 - 512bits UL (no security) used for tickets
 - Other formats : 1K (768 Bytes data), 4K
 - The 16bits random of **MIFARE has been hacked**
 - Solution : MIFAREplus
- MIFARE DESFire preprogrammed card
Example: Oyster Card in London
- Gemalto: Mifare 4 Mobile
- Contactless Java Card



85%+ of the access control / Ticketing
ISO14443 market is
Mifare®

- NFC allows a device to read and write a contactless card, act like a contactless card and even connects to another NFC device to exchange data.

- 3 modes :
 - ▣ Card reading (MIFARE ...)
 - ▣ Peer to peer (initiator & target)
 - ▣ Card emulating

- Distance : 0 - 10 centimeters

- Bandwidth to 424 kbits/s

- NFC Forum : NDEF specs

- N-Mark: <http://www.nfc-forum.org/resources/N-Mark>

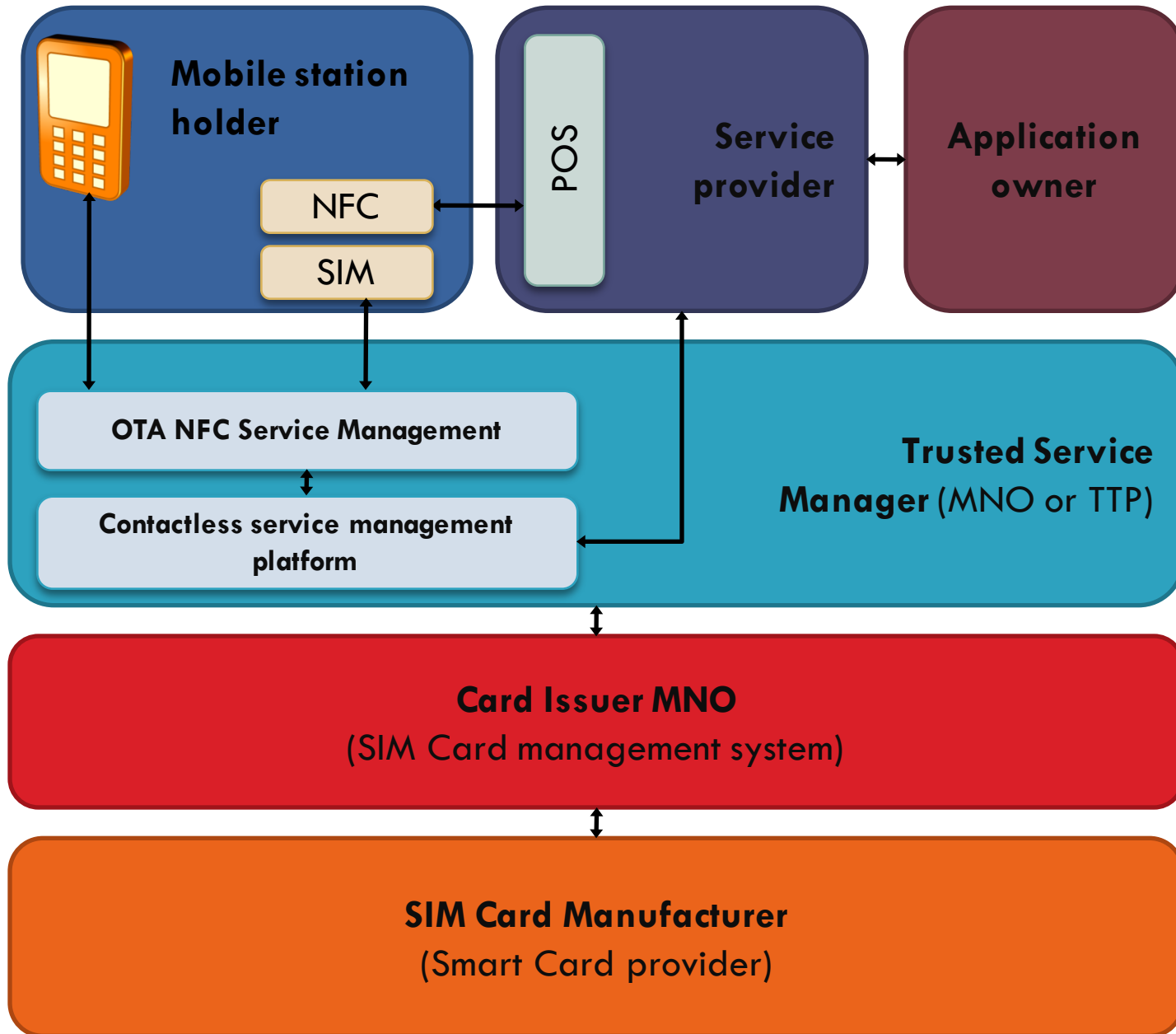


NFC on a Mobile Phone

one thing among all

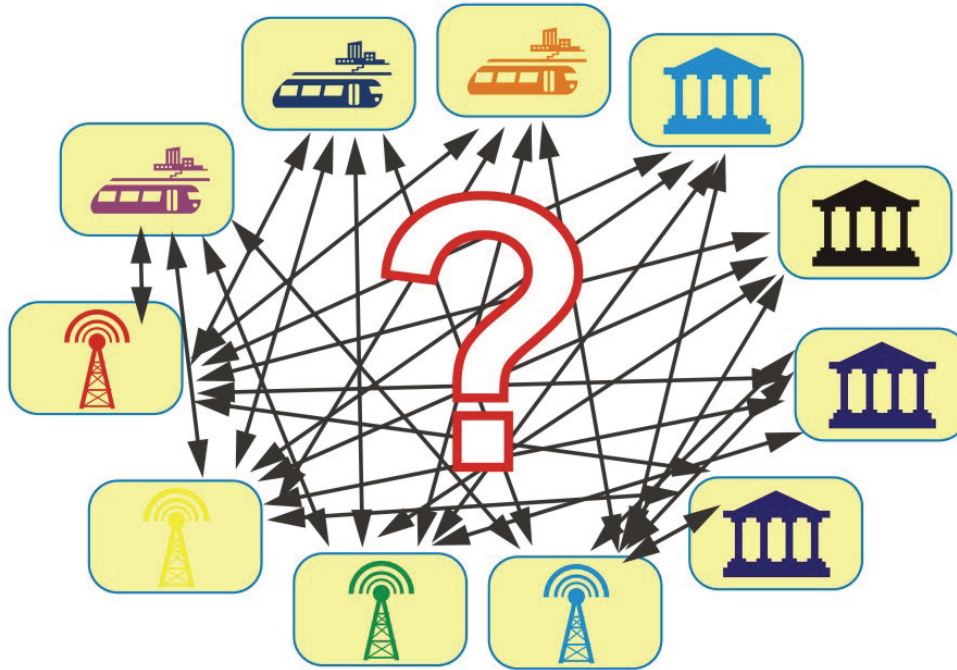


NFC Roles and actors



NFC without TSM

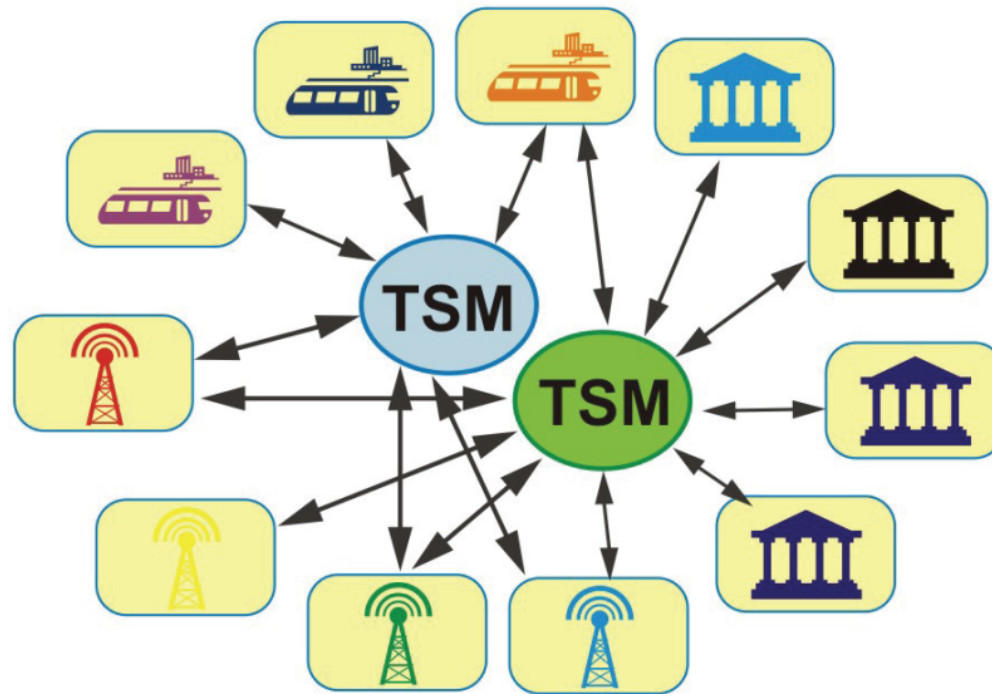
14



No regulations between MNO and services providers! It's complicated to manage services deployment on the mobile phone

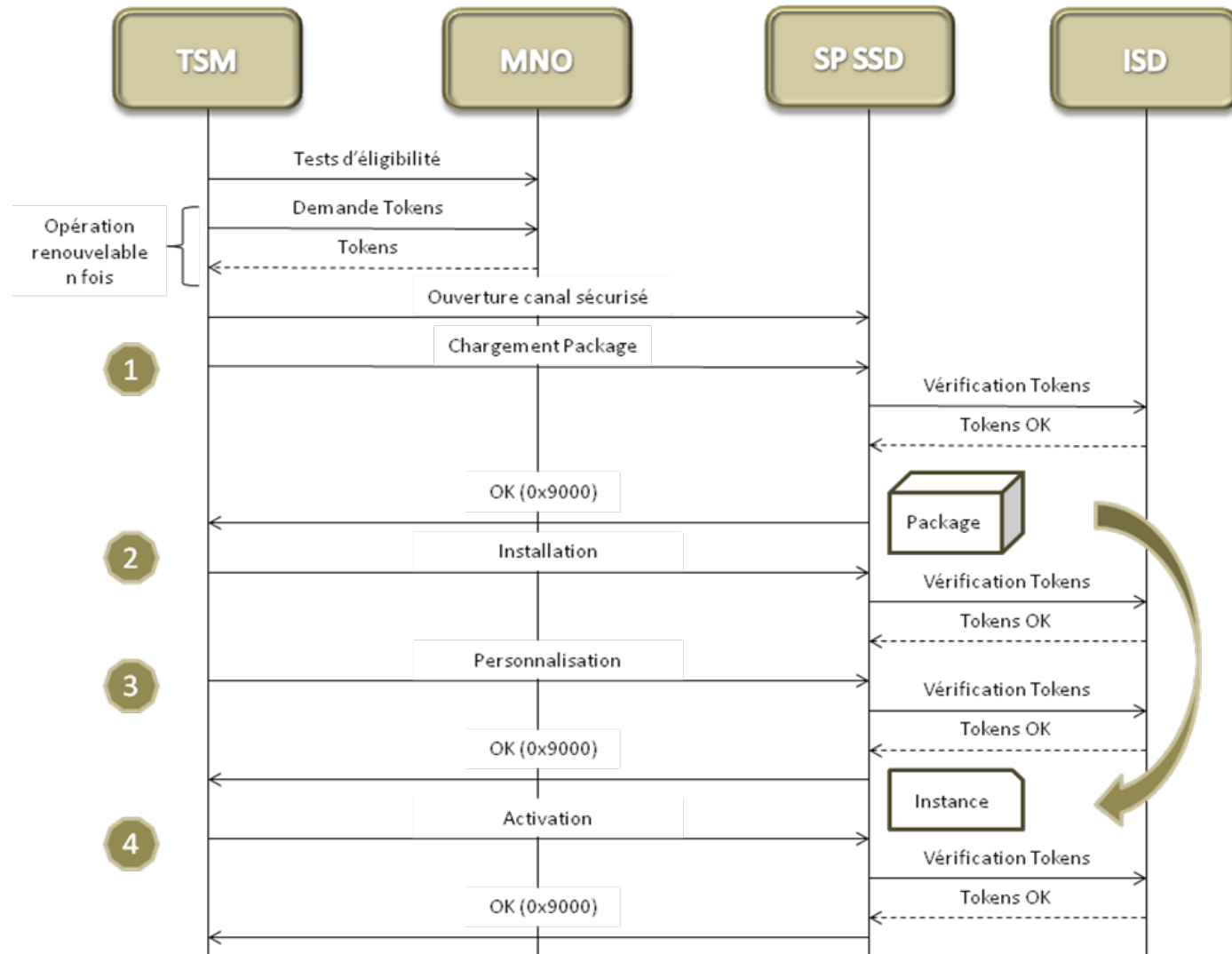
TSM as a regulator

15



TSM controls are trusted both by the services providers and MNO; they control deployment and updates of services (CARDLETS) on the smart phones

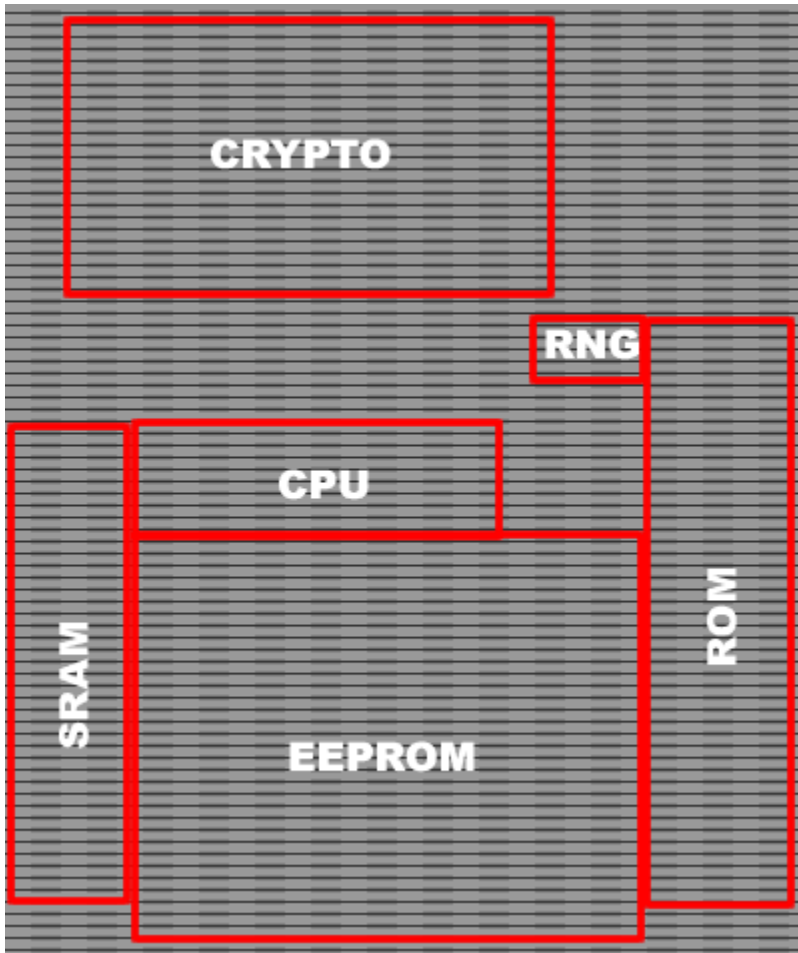
OTA Management



Smart Card

- Piece of plastic the size of a credit card hosting an electronic circuit that can **store** and **process** information.
- The integrated circuit (chip) may contain a microprocessor capable of processing this information, or it can only contain non-volatile memory with a security component (memory card).
- Smart cards are mainly used as means of personal identification (identity card, access badge to buildings, health insurance card, SIM card) or payment (credit card, electronic purse) or proof of subscription to prepaid services (calling card, ticket).
- Contact or Contactless smart card readers are used as a communications medium between the smart card and a host (point of sale).

Smart Card anatomy



- CPU: Control Processing Unit
- SRAM: Static Random Access Memory
- ROM: Read Only Memory
 - Static
 - Store the Operating System
- EEPROM: Electrically Erasable and Programmable Read Only Memory
 - Persistent
- CRYPTO: Cryptographic processor
- RNG: Random Number Generator
 - Used to generate keys



Smart Card used in France for healthcare refunds
(Carte Vitale)

Smart card applications

- Payment
 - ▣ Credit card, SIM card, TV Channel card, Access card
 - ▣ Transports
 - ▣ Electronic purse (coffee machine)

- Identification
 - ▣ PKI
 - ▣ Digital signature
 - ▣ Can store biometric data
 - ▣ 2009 in Spain and Belgium: eID card
 - 2 certificates: one used to authenticate and one to apply the digital signature (real legal value)

NFC Application categories

21

□ “**OPEN (non-secure)**”

1. **Read/Write mode** (similar to QR Code)
2. **P2P mode**

⇒ Ex: Tourism, Marketing 2.0, initiate Bluetooth® pairing...

□ « **SECURE** »

3. **Emulation card mode** (EMV, AMEX)

⇒ Ex: M-paiement, Ticketing (transport), Access control...

➔ **SECURE ELEMENT (SE), NFC MOBILE WALLET**

➔ **TSM (& OTA)**

SECURE ELEMENT (SE)

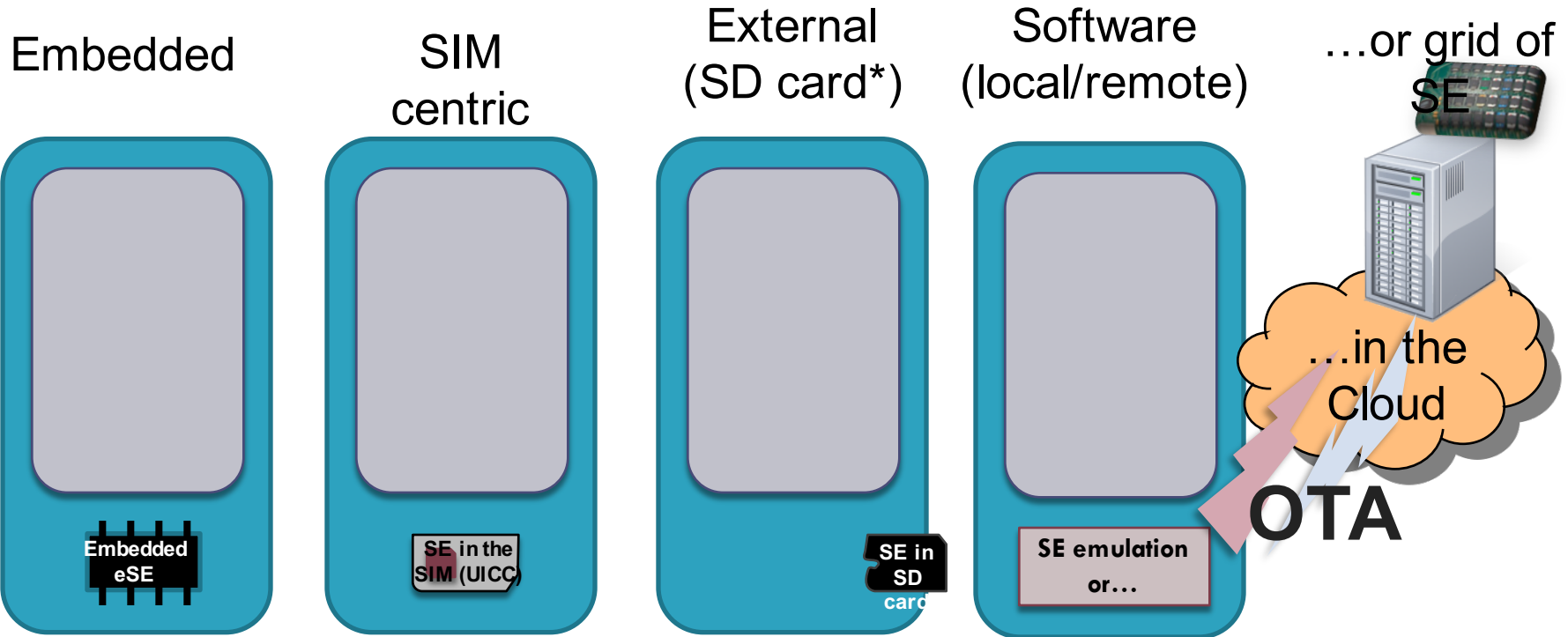
22

- Hardware
 - ▣ SIM Card (UICC) → MNO
 - ▣ In the terminal (embedded SE) → Manufacturer (Google, Samsung, Nokia, ..)
 - ▣ SD Card (removable SE) → Service Provider (Bank, Transport,...)
 - With or without NFC chipset
- Software
 - ▣ In the mobile (soft-SE)
 - ▣ In the Cloud (Cloud-SE)

Secure Element

Several architectures (may be combined)

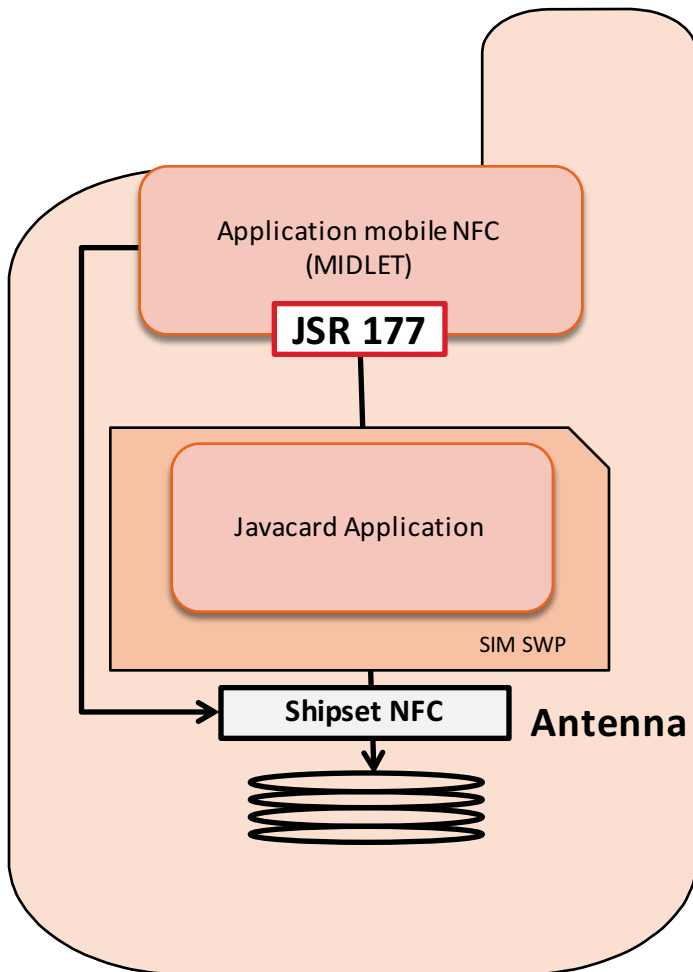
23



** with or without
NFC controller*

several OS : Java, Multos, proprietary

NFC application



- **MIDLET** (Java or Smartphone Application Android/W8,..)
- **CARDLET** (Javacard application)

Midlet & Cardlet

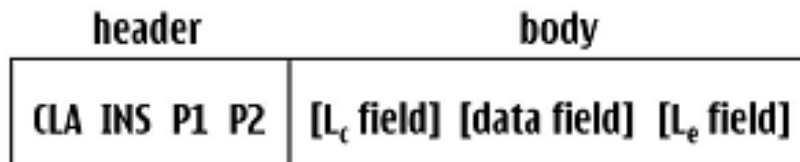
- Communication defined by the JSR 177 standard
- Information are exchanged using APDU command

APDU Command

- APlication Data Unit
- Standard for exchanging information with a Smartcard
- Two parts
 - APDU Command
 - APDU Response

APDU Command

- APDU Command (**C-APDU**), sent by reader to the card
- Hexadecimal string
- Header, 4 Bytes
 - Class instruction (**CLA**) : Class of the application (App id)
 - Code instruction (**INS**) : Type of the instruction (Read the balance)
 - Parameters : **P1** et **P2** : (**reserved bit**)
- Optional body (random size) → Extra data
 - **Lc** = length of body (data) in Bytes
 - **Le** = length of response to the command (Bytes)
 - The **data field** contains data to be sent to the card, to process instructions specified in the header.



APDU command types

- 4 APDUs commands are possible depending on whether it expects a response back or if it contains data.
 - ▣ **No data, no required answer**
 - CLA INS P1 P2
 - ▣ **Data, no required answer**
 - CLA INS P1 P2 Lc Data
 - ▣ **No data, required answer**
 - CLA INS P1 P2 Le
 - ▣ **Data, required answer**
 - CLA INS P1 P2 Lc Data Le

APDU Response

- An APDU response is the response to an APDU command
- Two parts
 - ▣ Response Data (size defined by the le bit in the APDU command)
 - ▣ Response status (SW1, SW2)
- Successful answer must be ended by 9000

Response APDU		
Response	SW1	SW2

Applet – An example

- A wallet that stores the account of a user in his phone
- The user can :
 - ▣ Retrieve
 - ▣ Recharge
 - ▣ Debit

Applet CLA

- We choose the hexadecimal value 0xB0 to identify our Wallet.
- This value identifies all APDU commands that are processed by the applet.
- It means that the APDU commands debit and credit all start with the byte CLA 0xB0.

```
Wallet_CLA =(byte)0xB0;
```

Applet INS

- The 2nd byte of an APDU command identifies the action to be processed

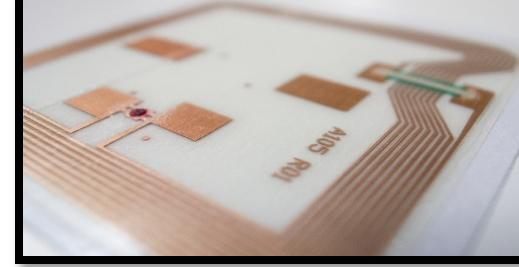
final static byte VERIFY = (byte) 0x20;
final static byte CREDIT = (byte) 0x30;
final static byte DEBIT = (byte) 0x40;
final static byte GET_BALANCE = (byte) 0x50

Examples of APDU command

Suppose that the account has 5 euros

- Get the balance
 - ▣ Response :
- Recharge of 20 euros
 - ▣ Response :
- Debit of 10 euros
 - ▣ Response :
- Get the balance
 - ▣ Response :

Tag Types



Interoperability between tag providers and NFC device manufacturers

- **Type 1**, based on ISO14443A. Tags are read and re-write capable; users can configure the tag to become read-only. Memory availability is 96 bytes and expandable to 2 Kbytes. Communication speed is 106 Kbit/s.
- **Type 2**, same as Type 1 except that memory availability is 48 bytes and expandable to 2 Kbytes.
- **Type 3** is based on FeliCa. Tags are pre-configured at manufacture to be either read and re-writable, or read-only. Memory limit is 1Mbyte per service. Communication speed is 212 Kbit/s or 424 Kbit/s.
- **Type 4**, fully compatible with ISO14443A and B standards. Tags are pre-configured. Up to 32 Kbytes per service. Communication speed is up to 424 Kbit/s.

NDEF : NFC Data Exchange Format

- Standard data format defined by the NFC Forum use to described how a set of actions can be encoded into a NFC tag or to exchanged between two active NFC device.
- Two basics parts :
 - ▣ Record Type
 - ▣ Payload Data

Record Type

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**
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
Sig	Signature	urn:nfc:wkt:Sig	NFC Forum Signature RTD

Record Type

37

- **Some URIs prefix (Record type U)**

0x00 : pas de préfixe

0x01 : http://www.

0x02 : https://www.

0x03 : http://

0x04 : https://

0x05 : tel:

0x06 : mailto:

0x1D : file://



NFC Application on Android

- Android supports the three NFC mode
 - ▣ Card reading/writing
 - ▣ Pear to Pear
 - ▣ Card Emulation
- In this session we will experience the Card reading/writing mode

NFC Permissions & features

- The following permissions are required in order to use the NFC chipset in an Android application
 - ▣ Test if the devices is nfc enabled

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

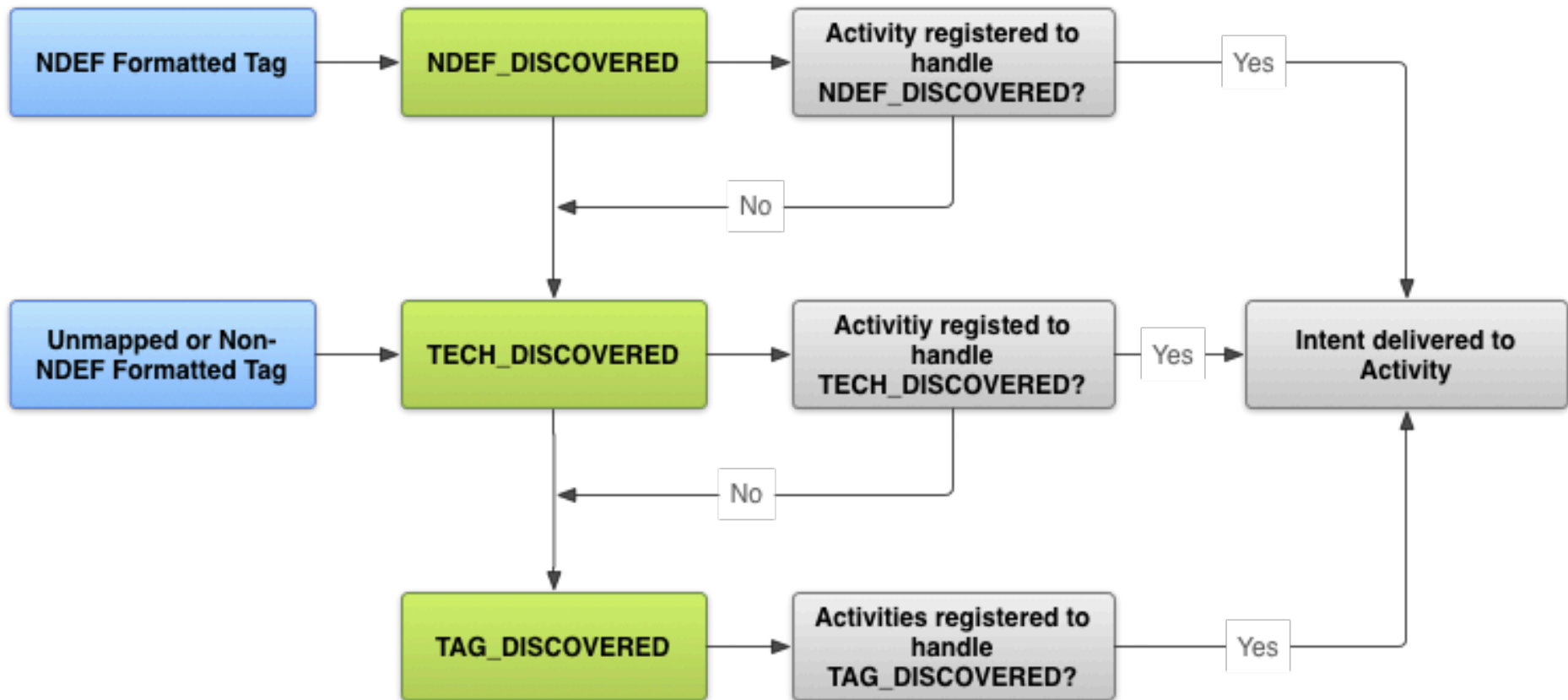
- ▣ Ask for permission

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

Intent filter

- There are three different filter for NFC tags (sorted from highest to lowest priority)
 - ▣ ACTION_NDEF_DISCOVERED
 - ▣ ACTION_TECH_DISCOVERED
 - ▣ ACTION_TAG_DISCOVERED

Android filter Intent resolution



Tech discovered intent

□ Enable tag discovering by the activity

```
<intent-filter>
```

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

```
<category android:name="android.intent.category.DEFAULT" />
```

```
</intent-filter>
```

□ List of NFC tags to be discovered

```
<meta-data android:name="android.nfc.action.TECH_DISCOVERED"
```

```
    android:resource="@xml/nfc_tech_filter" />
```

List of NFC Tags

□ Add the tags to be discovered in an xml file

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

NDEF Discovered Intent

```
<intent-filter>  
  <action android:name="android.nfc.action.NDEF_DISCOVERED" />  
  <category android:name="android.intent.category.DEFAULT" />  
  <data android:mimeType="text/plain" />  
</intent-filter>
```

Filter on a specific domain

- Read the tags that contain URLs of type (<http://mbds-fr.org/application>)

```
<intent-filter>
  <action
    android:name="android.nfc.action.NDEF_DISCOVERED"/>
  <category android:name="android.intent.category.DEFAULT" />
  <data
    android:host="www.mbds-fr.org"
    android:pathPrefix="/application/"
    android:scheme="http" />
</intent-filter>
```

Resources

- ❑ <http://discussion.forum.nokia.com/forum/forumdisplay.php?f=144>
- ❑ <http://wiki.forum.nokia.com/index.php/NFC>
- ❑ <http://forum.java.sun.com/forum.jspa?forumID=23>
- ❑ <http://www.nearfieldcommunicationsworld.com>
- ❑ <http://www.talknfc.com>
- ❑ <http://www.blognfc.com>
- ❑ <http://www.nfcnews.com>
- ❑ **Writing a Java Card Applet**
<http://developers.sun.com/mobility/javacard/articles/intro/index.html>



Resources

- Contactless Smart Cards and NFC
Peter Harrop, Ning Xiao & Raghu Das
- <http://www.nxp.com>, thanks for pictures
- <http://www.nearfield.org>
- <http://www.nfc-forum.org>
- <http://www.gsmworld.com/documents/>
- <http://www.rfidjournal.com>
RFID Information
- <http://mobilepayment.typepad.com>
Mobile payment blog
- <http://0x9000.blogspot.com>
Great blog on Java Card development

