NFC Stack Integration in a Complex Environment

Standard Android API for NFC

Christian Lührs

stollmann

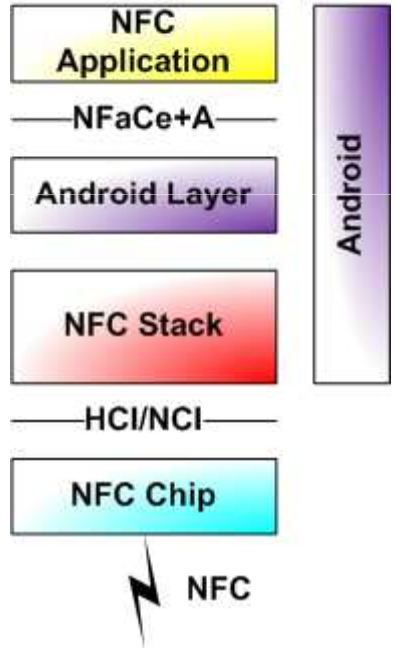
Bluetooth CAPI COMMON ISDN INTERFACE NFC forum Continua

Stollmann

- 20+ years experience of protocol stack development (ISDN, Bluetooth, IADs)
- Leading developer of NFC protocol stacks
- Member of the NFC Forum since 2006
- Various hard- and software NFC platforms
- Reference software for STMicro NFC chips
- Design of standard APIs
  for ISDN (CAPI) and Bluetooth (BECI)

Why an Android API for NFC?

- One API means one application development for all Android devices (phones, PCs, etc.)
- Enables revenues on NFC from “app stores”
- Will make NFC a standard feature for the Android community
- Well defined API is a mandatory requirement from OHA/Google



Requirements

- Abstracted access to NFC technology
- Easy and simple to use API
- Applications developer need to create apps, not to learn a technology
- Must be designed “top down” from application level
- True application interface, not hardware driver
- Independent of NFC chip
- Support for all devices, all use cases
- Non-standard/new use cases (e.g. Secure Simple Pairing)
- Hide new extensions (no changes to the API itself)

Vendor specific APIs

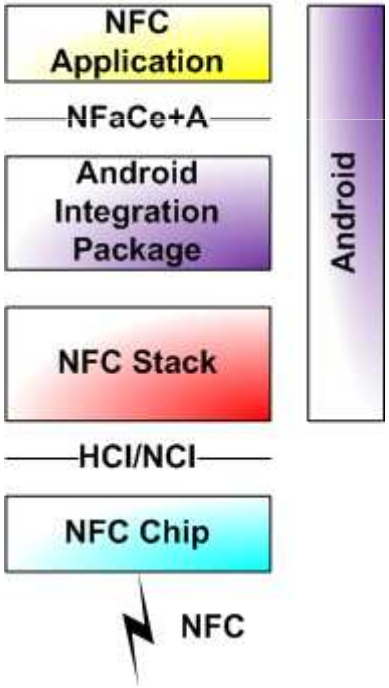
- Used to advertise a product or chip interface on the market as a “standard”
- No abstracted NFC access
- Incomplete feature sets
- Vendor specific features as integral parts of the API
- No proof of concept (e.g. only simulated on Windows)
- No support from independent protocol stacks or platforms
- Specification not reviewed by third parties (standardization process)

Source-Code for free?

- Source-Code is not a standard API
- An API is a well defined application interface, not only a protocol stack in source-code for free
- Providing source-code without API standardization forces the implementer to use the API with just one vendor specific chip set

NFaCe+A Basics

- High level object oriented API,
  easy to understand / use
- Android JAVA API design principles
  (no JSR 257)
- All NFC modes Reader/Writer,
  Card Emulation, Peer-to-Peer supported
- Independent of the underlying
  NFC controller chip
- Platform specifics accessible
  by the application



NFaCe+A Functions

- Multiple NFC applications concurrently supported
- Legacy NFC tag types are modeled as different Java connection objects
- Supports a Secure Element attached to the NFC controller
- Fully integrated into the Android security concept
- Android INTENTs used for control of local NFC adapter

NFaCe+A Class Hierarchy I

- NfcAdapter
- NfcListener
  - NdefListener
- TransactionListener
- P2PListener

- NfcConnection
  - FelicaConnection
  - ISO14443Connection
  - ISO14443BConnection
  - ISO14443BPrimeConnection
  - JewelConnection
  - MifareDesfireConnection
  - MifareStdConnection
  - MifareULConnection
  - NdefConnection
- P2PConnection

NFaCe+A Class Hierarchy II

- NFCCardEmulation
  - FelicaCardEmulation
  - ISO14443ACardEmulation
  - ISO14443BCardEmulation
- NfcCardDelegate
- NfcTargetType
- NfcUid
- NfcCardEmulationTypes

- NdefMessage
- NdefRecord
- NdefRecordType

Application Sample Code I

```
/* obtain instance of main NFC adapter object */
        NfcAdapter adapter = NFCAdapter.getAdapter();
/* enable the NFC adapter */
        adapter.enable();
/* create listener for NDEF tags */
        NdefListener listener =
                adapter.createNdefListener(delegate);
/* activate the listener */
        listener.start();
/* now the NFC field is active. When a NDEF tag is inserted in the field, the
    delegate object method getConnection is called (call back) */
```

Application Sample Code II

```
/* delegate method getConnection */
   delegate.getConnection (NdefConnection connection)

   {
/* create logical connection to tag */
        connection.connect();
/* read NDEF message from tag */
        NdefMessage message = connection.readMessage();
/* logical disconnect from tag */
        connection.disconnect();
   }
```

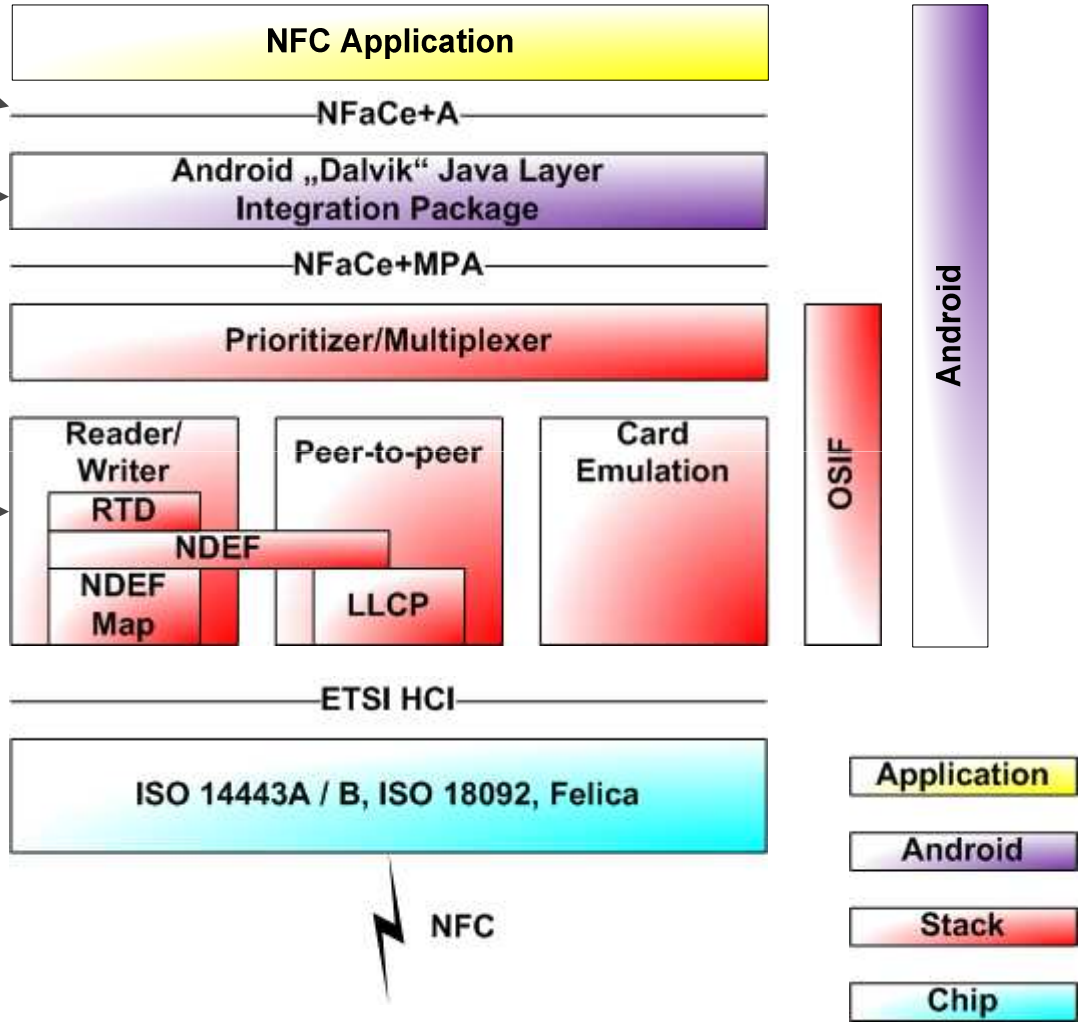
Architecture

NFaCe+ A: **Release Today**

NFC for Android: **Q2/10**

NFaCe+MPA: Available

NFCStack+ for Linux: Available

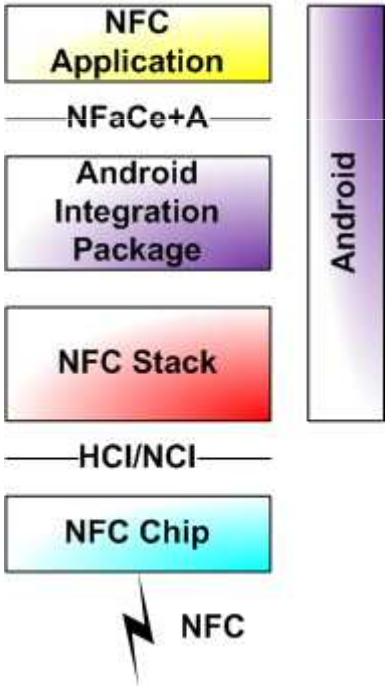


Stollmann API Strategy

- Create a native NFC API for Android
- Respect design rules for standard Android APIs
- Make room to adjust the API / functions
- Hide new extensions under abstract API
  (no changes to the API itself)
- Contribute Android NFC Integration Package to the community as open-source (Google / OHA)
- Stollmann NFCStack+ is not part of the contribution
- The Android Integration Package will be adaptable to other protocol stacks

Android Integration Package

- Native driver support on Linux OS level for various NFC chips (e.g. I2C, UART, etc.)
- Native Android JAVA “Dalvik” layer
- SDK based on common "Beagle" hardware
- All NFC modes (CE, R/W, P2P)
- Legacy tag support
- Host based Card Emulation
- Dynamic content in CE mode
- Multi Secure Element support
- Supports e.g. Felica or ISO14443



Choose a competent partner for communication!

„Bluetooth, IEEE 11073, ISDN, Cable, NFC – driven by Stollmann“