3

NFC Use Cases

NFC applications are implemented in three different modes:

– NFC applications using the reader/writer mode to read information (e.g. a URL, an activation code);

– NFC P2P mode to exchange information (e.g. computer files exchange) or to pair two NFC devices (e.g. exchange of connection data);

– NFC application transactions based on the card emulation mode of NFC and communication with services offering interfaces based on smart cards standard (e.g. for m-payment transactions, e-ticketing, identification or access control).

In this chapter, we offer tangible examples of the use of the NFC standard illustrating each of the three operating modes of NFC in a classic use case. However, we must keep in mind that NFC can be applied to any sector for a variety of use cases.

3.1. Usage of the NFC reader/writer mode

The NFC standard reader/writer mode is similar to the use case of QR codes, with the advantage that the user does not need to select a program, nor to center a picture in order to access the content, but simply to bring the NFC mobile device (i.e. acting as a reader) into the

The Art and Science of NFC Programming, First Edition.
Anne-Marie Lesas and Serge Miranda.

close proximity of the "target" object to be read (e.g. NFC tag). This thus automatically starts the mobile application linked to the tag content (see section 2.2.2.1). The target object contains a NDEF message; it can be any material object, for example a building, a door, a poster or a book. The content of the NFC tag (e.g. a URL and a code) acts as a link with information in the digital world (e.g. a work of art story in a museum, a user guide, tourist or place information, media video/music, etc.) or as a trigger for an action "run" when "tapping" (e.g. to open a door, turn a device on or track the visit of a user).

The NFC tag is a unique identifier of the "tagged" object (i.e. because of its UID) and the NFC mobile device is an identifier about the end user (i.e. IMEI and login information) and the context (i.e. "here and now": temporality, location, etc.). This property of precise contextualization of the event allows to create sophisticated use cases (e.g. m-payment and access control).

3.1.1. *Use case: management of equipment loans*

We propose a scenario of equipment loan management using the NFC reader/writer mode. The use case involves a user acting as the manager (here called "admin") and a user borrower (here called "user") holding an NFC smartphone, and something to be borrowed; it can be any kind of object (e.g. book, equipment, car share vehicle or hotel room) having an NFC tag linked to it.

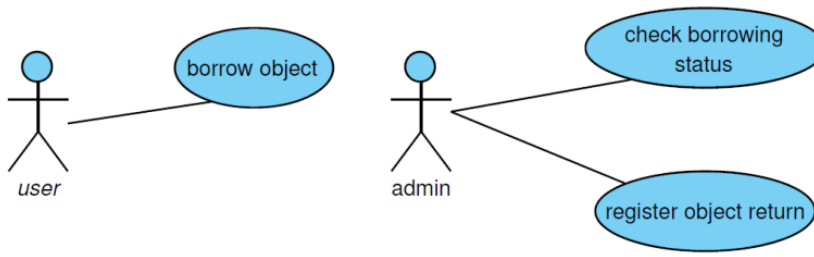

Figure 3.1. *Use case of borrowings management*

The system includes:

– an NFC mobile application for the user intended for the registering of the object borrowing;

– an NFC mobile application for the admin intended for the checking of the borrowing registrations, and for registering the object returns;

– a web server with a paired web services API and a storage unit (database) for information management. We assume that two web user interfaces (UI) will be available: (1) the loans management UI for admin users to control returns and (2) the user's UI to check his borrowing history: As this aspect does not have any impact on the NFC use case, it is not studied here;

– a communicating object having an NFC tag type interface.

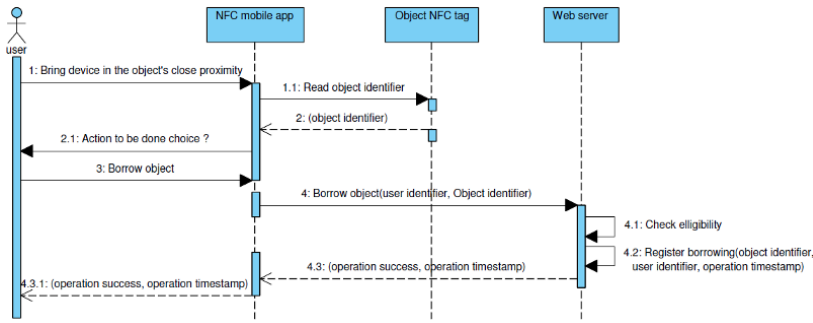

Figure 3.2. *Object borrowing*

Figure 3.2 shows interactions between the system components when the user borrows an object:

(1) the user taps the NFC interface of the object he/she wants to borrow with his/her NFC mobile phone;

(1.1) the application reads the object ID (it can be the UID or a code encoded on the tag earlier);

(2.1) the application asks for the action to perform (indeed, the application can offer several options such as, for example, display information on the object or the history of the user's borrowings);

(3) the user selects the option to borrow the object;

(4) the application connects to the server to register the borrowing;

(4.1) the server checks the eligibility of the request (indeed, the system checks that the user is allowed to borrow the object and that the object has not already been borrowed somewhere else);

(4.2) the borrowing is registered in the database;

(4.3) the server informs the application that the borrowing was properly registered;

(4.3.1) the application informs the user that the object borrowing was successfully registered.

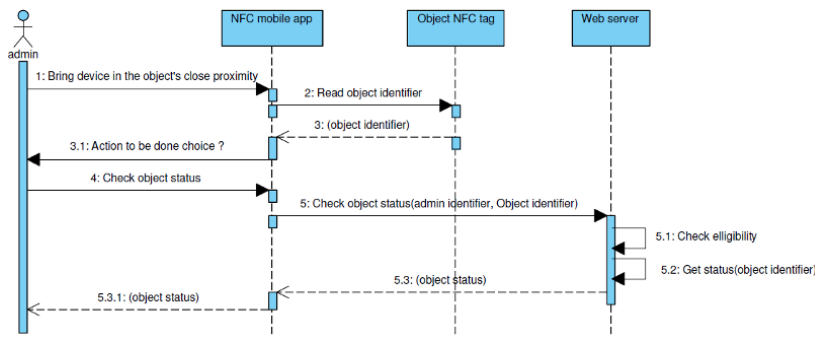

Figure 3.3. *Checking an object status*

An object (borrowed or not) can be checked at any time by a loans admin (for example to check that a user has properly registered the borrowing), as illustrated in Figure 3.3:

(1) the admin taps the NFC interface of the object to control with his/her NFC mobile device;

(2) the application reads the object identifier;

(3.1) the application asks for the action to perform;

(4) the admin selects the check object status option;

(5) the application connects to the server to collect data on the object;

(5.1) the server checks the eligibility of the request;

(5.2) the server collects data on the object status (or its history);

(5.3) the server informs the application that the borrowing was successfully registered;

(5.3.1) the application displays object information.

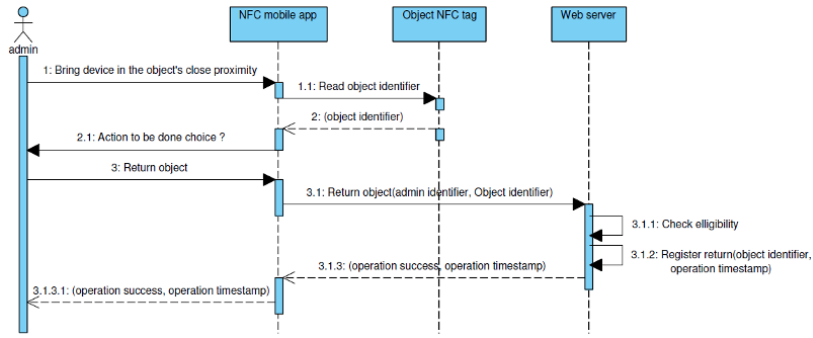

Figure 3.4. *Object return*

Returning a borrowed object is performed by the loans admin once the object has been physically returned by the user (borrower) (see Figure 3.4):

(1) the admin taps the NFC interface of the object to control with his/her NFC mobile;

(1.1) the application reads the identifier of the object;

(2) the application asks for the action to perform;

(3) the manager selects the return option of the object;

(3.1) the application connects to the server to register the object return;

(3.1.1) the server checks the request eligibility;

(3.1.2) the server registers the object return;

(3.1.3) the server informs the application that the return was successfully registered;

(3.1.3.1) the application informs the user that the return was successfully registered.

This example shows how simple it is to perform transactions with NFC technology, even without a costly professional device. The power of such a system is strengthened by incorporating geolocation, provided by the GPS of mobile phones during transaction requests; in this way, inference rules can be added to the eligibility check of transactions and/or additional functions around traceability can be implemented.

3.2. Usage of the NFC P2P mode

The simplest use case of the P2P mode of the NFC standard consists of exchanging data between two NFC peripheral devices. This can simply be media sharing between two users, i.e. where a first user starts the exchange by choosing beforehand the document he/she wishes to share with the target device of the other user. The two users will then bring their two devices into close proximity and the NFC transfer will occur from the initiator device to the target device.

3.2.1. *Use case: NFC pairing*

The use case of NFC P2P mode addressed in this section involves a user having two NFC-enabled devices with another high-speed connectivity (e.g. Bluetooth® and Wi-Fi):

– the “master” device has a user interface (for example a smartphone);

– the “slave” device does not need to have a user interface (for example a speaker, but it could be any object: a coffee machine, a car, a house, etc.).

Here, the “master” device (the smartphone) acts as a “universal remote control” allowing the user to connect and control the “objects”, NFC technology being the “universal connector” to pair devices with “zero manual configuration”.

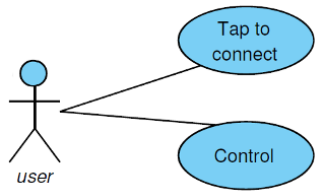

Figure 3.5. *Use case of pairing*

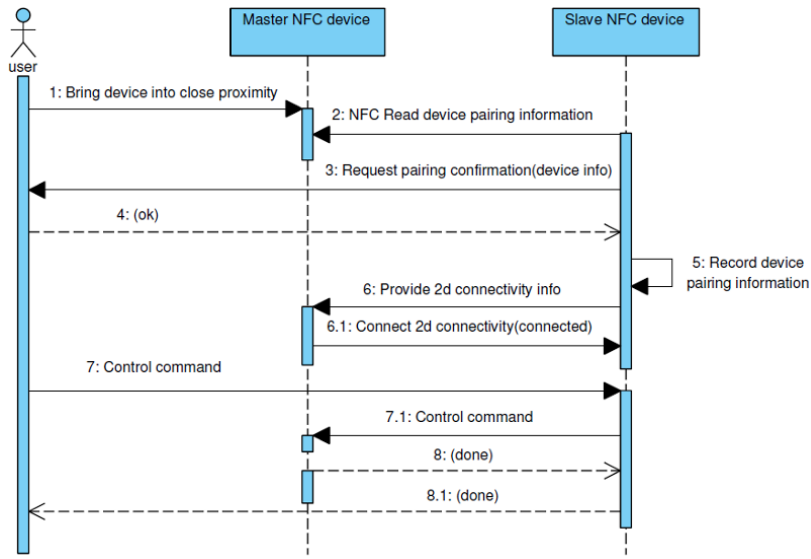

Figure 3.6. *Interactions of pairing scenario*

The interactions of our pairing scenario (see Figure 3.6.) show how a second connectivity can be set up with NFC:

(1) the user brings the slave peripheral device to pair into close proximity with the control device (master);

(2) the master device (e.g. smartphone) reads the pairing information of the device to pair thanks to NFC connectivity;

(3) since this is the first contact, the master device requests a confirmation from the user who wishes to pair with the slave device (after the first pairing, it will connect automatically, with no need for confirmation);

(4) the user confirms the pairing;

(5) the master device registers the parameters of the slave device in order to use it again later (e.g. authentication and control data);

(6) depending on pairing information, the master device sends information to allow the slave device to use the second connectivity (this part is to the developer's judgment with the help of the NFC standard forum mentioned in section 1.3.4 and depends on the API available from the development platform);

(6.1) the slave device connects to the master device through the second connectivity;

(7) the user can then send control commands to the slave device through the user interface of the master device;

(8) the master device sends commands to the slave device according to the pattern provided by the collected data when initializing the pairing (with NFC), through the second connectivity;

(8.1) the slave device notifies the master device that the command was run successfully; the user is notified by the master device.

3.3. Usage of NFC card emulation mode

NFC card emulation mode is characterized by the APDU layer (smart cards protocol and JavaCard™ platforms). NFC technology is thus a contactless transport layer for services embedded in a chip called "secure element": the SE (see section 1.3.2.2). When the SE is integrated into the mobile device (i.e. in a smartphone or a tablet), an API accessible from the OS of the mobile phone is necessary to communicate with the SE. This API is not always available for developers. On the Android platform (see section 2.2.4), OMAPI

(see section 1.3.6) enables communication with the SE in its three hardware form factos (SIM, eSE and microSD), regardless of the configuration, in the way of an internal reader (see section 2.2.4.3.2). Theoretically, all existing services based on a smart card (e.g. credit cards, transportation cards, identification cards, subscription cards and health insurance cards) can be hosted in an SE interfaced internally with an application benefiting from the advanced features of the mobile device.

An application in NFC card emulation mode is made of an on-card service (with restricted access) running in the SE, and of an application run in the host mobile device OS.

For the use case given as an example in section 3.3.1, we address a digital wallet scenario.

3.3.1. *Use case: digital wallet in the SE*

We present the original use case[1] of a digital wallet as an example for the NFC card emulation mode. The digital wallet can act as alternative virtual currency for community trades; it can thus be a barter currency between two individuals within the community, or a dedicated currency virtualizing service vouchers, or loyalty points (e.g. meal vouchers, holiday vouchers, gift vouchers), which are only valuable in a specific context.

The system involves two users equipped with NFC devices acting, respectively, as initiator and target for a debit/credit transaction (see Figure 3.7), where the target user authorizes the transaction by bringing into proximity and "tapping" the initiator mobile. The digital wallet of the payer user will then be debited and the wallet of the debit user will receive the same amount. We assume an opening balance of zero and the possibility of negative balance (i.e. in a balanced economic system, scales must balance with transactions). The system includes an application acting as a user interface to manage

1 All intellectual property rights reserved to the author.

debit/credit transactions interfacing with a digital wallet service hosted in the SE.

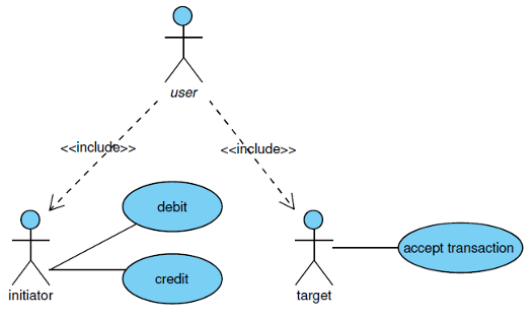

Figure 3.7. *Use case of a digital wallet*

NOTE.– In this example, we forget the more complex aspects of the lifecycle management of the on-card SE service, which is standardized and well documented by GlobalPlatform (see section 1.3.5).

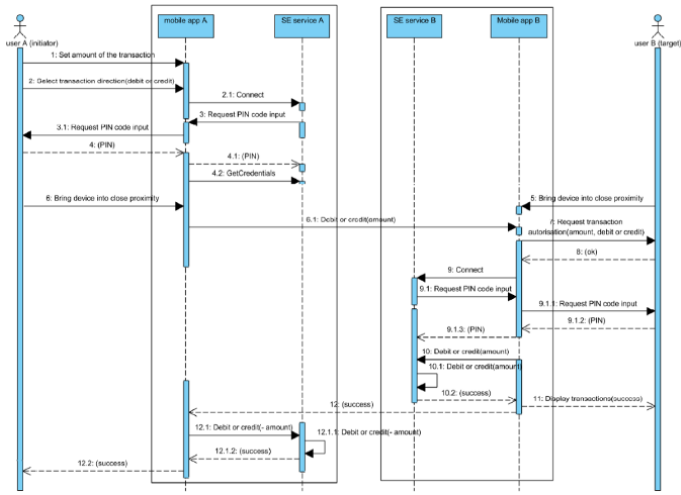

Figure 3.8. *Scenario of the digital wallet interactions with an SE*

Figure 3.8 shows the interactions between two NFC mobile devices during a debit/credit transaction:

(1) user A (initiator) enters the transaction amount;

(2) user A enters the direction of the transaction (debit or credit);

(2.1) the mobile app of user A connects in NFC card emulation mode with the digital wallet service hosted in the SE of user A's mobile phone through an internal reader (i.e. through the SE communication API);

(3) SE A sends an authentication request via PIN code;

(3.1) mobile app A asks user A to enter his/her PIN code;

(4), (4.1) user A enters his/her PIN code, which is transmitted to the SE service A;

(5), (6) users A and B bring their mobile devices into close proximity;

(6.1) mobile app A sends the debit/credit request to mobile app B via the NFC P2P mode;

(7) mobile app B asks user B for credit/debit transaction authorization;

(8) user B gives authorization for the transaction;

(9) mobile app B connects in NFC card emulation mode to the digital wallet service hosted in SE B;

(9.1) SE B sends back an identification request via PIN code;

(9.1.1) mobile app B asks user B to enter his PIN code;

(9.1.2), (9.1.3) user B enters his/her PIN code, which is sent to the service of SE B;

(10) mobile app B sends back a debit/credit command to the service of SE B;

(10.1) the digital wallet service of SE B registers the debit/credit transaction;

(10.2), (11) the SE service B notifies that the debit/credit transaction was processed successfully, which is notified by the mobile app to user B;

(12) mobile app B notifies mobile app A that the transaction succeeded by using NFC P2P mode (at this point, the two devices must touch);

(12.1), (12.1.1) mobile app A sends a payment reversal request (debit/credit) of the same amount (i.e. of reverse sign) to the digital wallet service of SE A that registers the transaction;

(12.1.2), (12.2) the service of SE A notifies mobile app A the transaction succeeded, which is notified to user A.

NOTE.– In our original example of NFC card emulation mode usage, the SE service is not directly accessed externally. However, the configuration we propose is fully adapted to transactions processed with a terminal connected to a contactless NFC reader.

3.4. Usage of the HCE mode

The Host-based card emulation (HCE) mode is a special case of NFC card emulation mode: the on-card SE service is replaced by a background service running in the OS of the mobile device that has an interface able to manage APDUs (see section 2.2.5). The HCE mode does not require a special communication API to interface the SE (as there is no SE involved) and benefits from the access to advanced features of the mobile device (contrary to SE services running in the restricted environment of the smart card's microchip). Moreover, the management of the lifecycle of an HCE service is the same as the management of an ordinary mobile application, which makes its implementation much simpler and does not require interfacing with third parties (i.e. TSM/SEI).

The use case of the NFC card emulation mode implementing an SE suggested in the previous section (see section 3.3.1) could also be implemented in the same way with an HCE service instead of the on-card SE service. In the use case example of HCE proposed in section 3.4.1, the HCE service acts as a gateway for an SE accessed remotely in the Cloud.

3.4.1. *Use case: SE in the Cloud with HCE*

Our scenario of the usage of HCE mode describes the interactions of a use case where the mobile HCE service receives APDU instructions from a terminal NFC reader and sends them to a remote SE service hosted in the Cloud:

(1) the user brings his/her NFC mobile device into close proximity with the NFC reader;

(2) the terminal application connected to the NFC reader sends APDU instructions to the mobile app (received by the HCE service);

(2.1), (3) the mobile app connects to a remote web server;

(3.1) the mobile app forwards the APDU instruction (or the sequence of instructions) to the web server;

(4) the web server sends the instruction (C-APDU) to the hosted SE service;

(4.1), (4.2) the SE service processes the instruction(s) and returns the response (R-APDU);

(4.2.1) the web server returns the APDU response to the mobile app;

(5) the mobile app returns the APDU response to the terminal application.

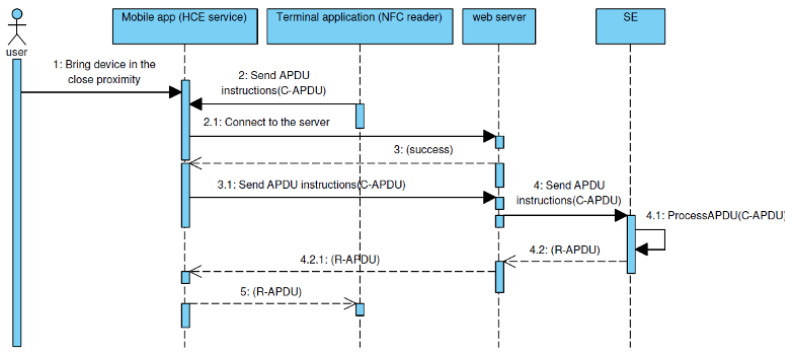

Figure 3.9. *Interactions scenario SE in the Cloud with HCE*

NOTE.– This use case solution of SE in the Cloud using HCE requires Internet connectivity.