_ TELECOM ESCUELA TÉCNICA VLC SUPERIOR DE UPV INGENIEROS DE TELECOMUNICACIÓN

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

NFC Playground: Estudio de la emulación de una sesión EMV® remota basada en contactos a través de una sesión EMV® basada en HCE

Valencia, julio de 2018

Trabajo de fin de grado de: José Talens Capellino

Dirigido por: Francisco José Martínez Zaldívar

Agradecimientos

A mis padres, Pepe y Sara, porque sois vosotros y no yo quienes merecéis todos los honores por tantos días y noches de desvelo.
A mi hermano David, por aguantarme, ser un luchador nato y haber logrado victorias donde la mayoría fracasaron.
A mis abuelos, Antonio y Francisca. Ojalá os lo pudiera contar en persona.
A Ignacio Melón Andrés. Nachete, estés donde estés, me gustaría hacerte saber que el día que me despedí de ti por última vez puse fecha a este momento. Siempre te llevaré conmigo. Va por ti también, por ser probablemente el humano más sensible e inteligente que jamás he conocido.
A Paco Martínez Zaldívar, por tu diligencia y tu conocimiento, solo comparables a tu cercanía.
A Sergio Domínguez García por el PIN Pad, las tarjetas y alguna que otra consulta intempestiva.

No puedo olvidar a aquellos que de una u otra forma me quisisteis mal. Vuestra falta de aprecio me hizo inasequible al desaliento.

¡IOS, qué buen vassallo,
si oviesse buen señor!
Cantar de Mio Cid, v. 20

Resumen

El acrónimo **EMV®** responde a las siglas de *Europay MasterCard VISA* y hace referencia al conjunto de especificaciones publicadas en 1996 que definen toda la pila de capas del marco de operación de tarjetas de circuito integrado (**ICCs**) en entornos de pago. En 2017, el volumen de transacciones **EMV®** en Europa fue del 98.62 % sobre el total.
Las siglas **NFC** hacen referencia al estándar de conectividad inalámbrica de proximidad entre dispositivos *Near Field Communication* -subconjunto de **RFID** *(Radio Frequency Identification)*- basado en la norma **ISO/IEC 14443**, que limita el rango de actividad a unos pocos centímetros.
Desde que en 2004 se emitiera la primera **ICC** con operatividad sobre **NFC** -*contactless*- el nivel de adopción de esta tecnología en el marco de sistemas **EMV®** -y no **EMV®**- ha crecido exponencialmente.
Por su parte, en octubre de 2013 Google libera la versión 4.4 de **Android** (*KitKat*). Esta versión contempla la posibilidad de emular **ICCs** por software sobre **NFC** mediante **HCE** (*Host-based Card Emulation*).
El presente proyecto pretende conjugar las tecnologías anteriores en un estudio sobre la posibilidad de generar una sesión **EMV®** según recoge la norma **ISO/IEC 7816-4**, en la parte superior de la pila de protocolos de **HCE** partiendo de una sesión **EMV®** remota basada en contactos.
El resultado de la documentación, investigación y trabajo adicional se ha sustanciado en forma de tres desarrollos:

- **NFC Playground**: Aplicación para dispositivos **Android** que sirve como herramienta de *PoC (Proof of concept)* del escenario que se pretende estudiar en particular así como de los diferentes escenarios habituales de un entorno **EMV®** sobre **HCE**.

- **NFC Playground Remote Driver**: Componente remoto que hace de adaptador entre la sesión **EMV®** de contactos original a través de un lector conectado a un servidor, y el desarrollo anterior.

- **NFC Playground PIN Pad Demo Component**: Componente local que habilita el PIN Pad utilizado para las pruebas realizadas en la demostración.

Índice general

Capítulo 1

Introducción

1.1 Descripción del problema

Los últimos años hemos asistido a un auge excepcional en la presencia de tecnologías de comunicación inalámbrica de proximidad, es decir, aquellas cuyo rango de trabajo no excede unos pocos centímetros, en prácticamente todos los ámbitos de nuestra vida: desde aplicaciones de corte más comercial o logístico orientadas a la gestión de stock, optimización de procesos o información, hasta ámbitos tan sensibles como la autenticación, la sanidad o la transferencia electrónica de fondos. En muchos casos el nivel de integración de estas tecnologías en nuestro día a día ha sido de tal magnitud que resulta evidente el cambio impuesto en nuestro modelo funcional y operativo.

El sector de medios de pago no ha sido una excepción. Si bien ya existían aproximaciones anteriores que cubrían la necesidad de un entorno de pago alternativo al uso de efectivo en toda su casuística con un nivel considerable de implantación y de madurez, en estos casos su tecnología dependía en última instancia de la conectividad física entre los actores más sensibles del sistema en un momento dado. La puerta de este nuevo escenario sin dependencia del contacto físico en medios de pago que se abre en 2004 con la publicación del conjunto de estándares de comunicación inalámbrica NFC*(Near Field Communication)* por parte del NFC Forum [1] no tardó en reproducirse en el parque de teléfonos inteligentes de última generación, donde hemos visto cómo estos dispositivos han empezado a llegar al usuario con las modificaciones necesarias tanto a nivel físico como lógico para permitir la implementación de sistemas de pago basados en tecnologías inalámbricas de proximidad.

El presente proyecto pretende cubrir el estudio de un escenario de cambio de paradigma a nivel de aplicación no estudiado anteriormente dentro de este ecosistema de medios de pago así como sus implicaciones de seguridad, contando como interlocutores al sistema operativo Android [2] como anfitrión, NFC como tecnología inalámbrica y EMV® [3] como protocolo de aplicación.

[1] `https://nfc-forum.org`
[2] `https://www.android.com`
[3] `https://www.emvco.com`

1.2 Motivación

Existen diferentes condiciones de contorno que tienen un factor coadyuvante en la elección de este estudio como proyecto final de carrera. En primer lugar la influencia del vínculo laboral con una empresa entre cuyos frentes de trabajo se encuentra desde hace más de 20 años el desarrollo de medios de pago basados, primero en protocolos nacionales implementados sobre la norma ISO/IEC 8583 [1] como PUC *(Protocolo Unificado de Comercios*), PRICE *(Protocolo Integrado de Conexión de Establecimientos)*, PUP *(Protocolo Unificado de Pinpads)* o PUCE *(Protocolo Unificado de Comercio Electrónico)* e internacionales como EMV® y PCI DSS[4] *(Payment Card Industry Data Security Standard)* con posterioridad, que permite una posición privilegiada de cara a entender la lógica de negocio del sector de medios de pago.

En segundo lugar, el nuevo abanico de posibilidades —y riesgos— que añade la capa de comunicación inalámbrica a una sesión del ecosistema de transferencia electrónica de fondos. Aunque especificaciones como NFC requieren un ámbito de proximidad relativamente cerrado, resulta innegable que el paso de la información de sesión a través de un medio vulnerable y compartido como es el espectro electromagnético obliga al sistema, dadas las condiciones adecuadas, a depender fuertemente de las implementaciones de seguridad que descansen sobre el protocolo de aplicación.

Por otra parte, una de las fortalezas de EMV® frente a sistemas anteriores es su seguridad basada en criptografía asimétrica, hasta el punto de que su implementación ha permitido apreciar márgenes de reducción en el fraude de entre un 58 % y un 73 % en función de la zona según Gemalto[5] [2]. Su gran nivel de adopción responde además de a la gran variedad de aplicaciones que soporta, sin duda a la capa de seguridad [3] que ha aportado a los entornos de pago.

El último componente tecnológico de los tres de que forman este trabajo es Android, como actor que hospedará al cliente del sistema anterior. Pese a que la tendencia que se aprecia en las sucesivas versiones liberadas es la de ir acotando la capacidad de interacción sobre -cada vez más- partes sensibles del SO, HCE *(Host Card Emulation)* todavía permite cierta capacidad de maniobra sobre la lógica que subyace en la capa de aplicación. Por su parte y de acuerdo con StatCounter[6] la cuota de mercado en el marco de sistemas operativos (SO) móviles es actualmente del 76.53 % [4].

Por tanto, nos encontramos con los tres líderes de sendas ligas: Android a nivel de SO móvil, NFC en el marco de protocolos de comunicación inalámbrica de proximidad y EMV® en el contexto de protocolos de aplicación de medios de pago. Tres protagonistas que definen, no solo el presente sino el futuro a corto y medio plazo de los escenarios de medios de pago.

[4] `https://www.pcisecuritystandards.org`
[5] `https://en.wikipedia.org/wiki/Gemalto`
[6] `http://statcounter.com`

1.3 Objetivos

Los objetivos de este proyecto son varios pero todos giran alrededor de un objetivo principal: el estudio de la posibilidad de **superar la fase de autenticación de tarjeta** —ver la figura Ejemplo de flujo de una una transacción EMV® [5]— de una sesión EMV® emulada por HCE partiendo de una sesión EMV® remota basada en contactos. Se pretende, por tanto, realizar una aproximación a las implicaciones y efectos que supondría este cambio de paradigma en capa física sobre la capa de aplicación, y por tanto sobre la sesión, y qué repercusión tendría en los escenarios habituales de entornos de pago basados en EMV®.

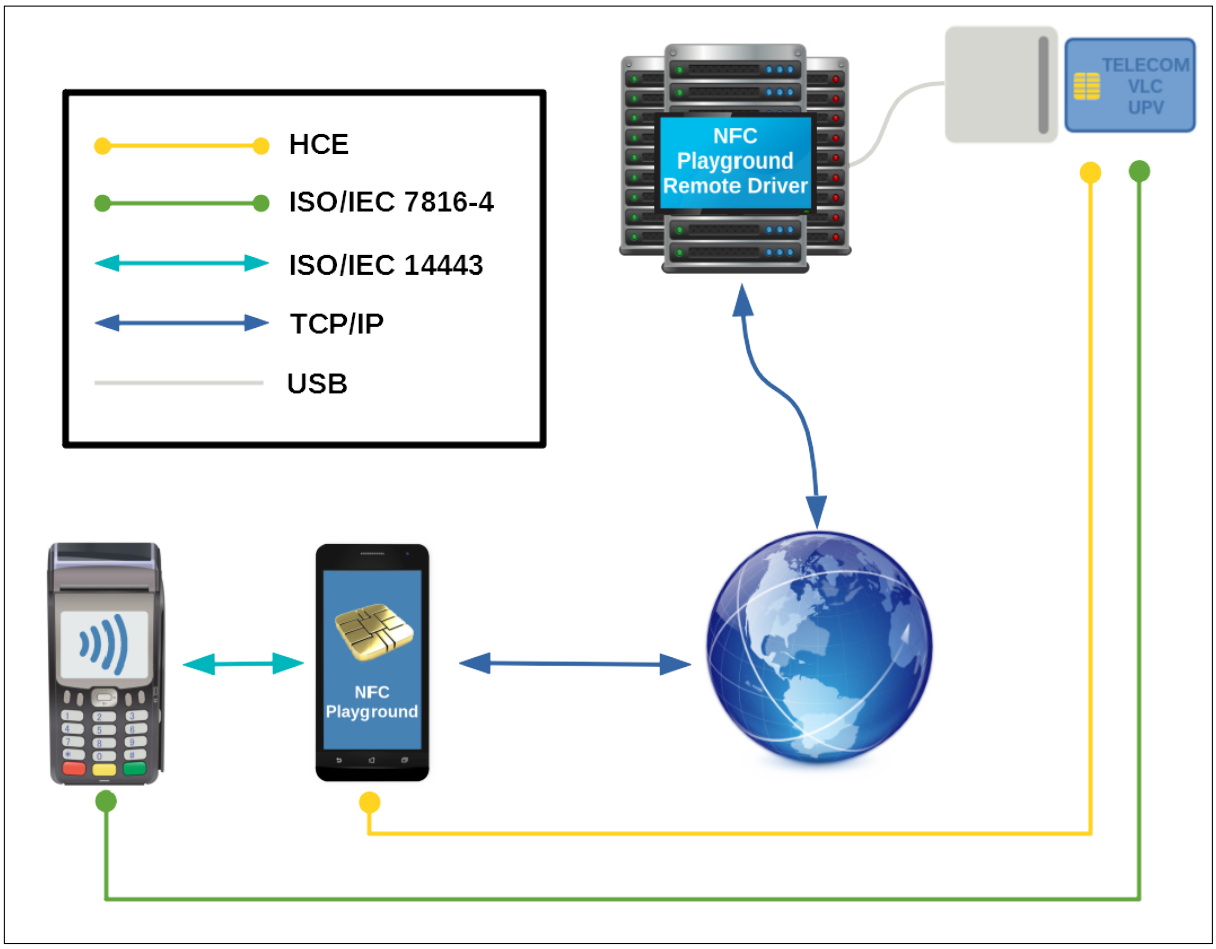


Figura 1.1: Diagrama del escenario inicial

En el diagrama anterior se aprecia cómo se establece —en verde— una sesión EMV® entre el pin pad y la tarjeta mediante la emulación —amarillo— de la misma sobre **TCP/IP** —en azul— a través del entorno desarrollado específicamente para este estudio formado por la App **NFC Playground** y el componente remoto de servidor **NFC Playground Remote Driver**.

Existe un escenario particularmente interesante que se pretende estudiar. Como se verá más adelante en el **Capítulo 3**, de todas las diferencias que existen entre una sesión EMV® basada en contactos y una basada en la norma ISO/IEC 14443 hay una que muy probablemente resulte conocida por cualquier usuario de este tipo de tarjetas con soporte inalámbrico. Éstas cuentan con una funcionalidad especial conocida como **Floor Limit**[7] y hace referencia al importe por

[7] `https://www.creditcards.com/glossary/term-floor-limit.php`

debajo del cual no se requiere autenticación de usuario -aunque sí de tarjeta-. Es una funcionalidad que pretende fidelizar al usuario al facilitarle aún más el pago de pequeñas cantidades que, como se tratará en el capítulo de Conocimientos Previos, son las más frecuentes. Al tratarse de operaciones de pequeños importes el comercio es quien asume el riesgo de fraude en estas operaciones. De ese modo, se consigue además alargar la vida del actual modelo físico de tarjeta bancaria.

En el supuesto de que una sesión EMV® basada en contactos consiguiera superar la fase de autenticación de tarjeta sobre una sesión emulada remotamente por HCE, todo el parque de tarjetas EMV® de contactos dispondría inmediatamente de funcionalidad Floor Limit. Por el hecho de estudiar dicho cambio de paradigma de forma remota, además, una tarjeta de estas características no necesitaría de PIN *(Personal Identification Number)* para realizar un cargo -o varios como se verá más adelante- **en cualquier ubicación del mundo** agrandando el impacto de este tipo de fraude.

Sólo en la zona de Europa, según el último informe [6] de The Smart Payment Association[8] (SPA), en 2016 existían **más de 3 millones de puntos de venta adaptados a pagos contactless**. Las cifras relativas a este medio de pago crecen de forma exponencial [7] lo que pone de manifiesto una disponibilidad geográfica a gran escala de puntos de pago susceptibles de permitir transacciones de Floor Limit.

Teniendo en cuenta todo lo anterior, aunque las cantidades a considerar no serían elevadas —a falta de considerar aquellas implementaciones de Floor Limit sin limitación en el número de transacciones— la puesta en escena de **este tipo de fraude tendría consecuencias desastrosas** en el sector por cuanto en primer lugar se verían afectadas, **con necesidad de contemplar la revisión** que evite este escenario, las siguientes especificaciones:

- La norma **ISO/IEC 7816-4**.
- Las especificaciones de **EMV®** relativas a la capa de aplicación en **entornos inalámbricos**.
- Las especificaciones de **EMV®** relativas a la capa de aplicación en **entornos de contactos**.

En segundo lugar una vez definidas las medidas a implementar, tendría que programarse el **despliegue coordinado en todas las zonas** (África y Oriente Medio, Asia, las dos zonas de Europa, EEUU y Canadá, LATAM y el Caribe) de las mismas **a nivel tanto de puntos de venta, como de pasarelas de pago y/o resolutores**.

En tercer lugar, **si además el correctivo obligara a cambios físicos** —o lógicos en zonas de seguridad sólo accesibles en el momento de construcción— a nivel de *chip*, todas las tarjetas afectadas, es decir **todo el parque de tarjetas mundial que implemente EMV® debería ser retirado**.

A priori, este sería el escenario más desafortunado a efectos de seguridad por cuanto **hasta la fecha este tipo de tarjetas sin soporte inalámbrico requieren de autenticación de usuario para realizar transacciones de cualquier importe.**

Adicionalmente, el siguiente objetivo a cubrir es el de profundizar en desarrollos complejos sobre el SDK de Android y entornos de pago.

[8] `https://www.smartpaymentassociation.com`

1.4 Estructura del documento

El documento se estructura en forma de 7 capítulos que pretenden cubrir todo el horizonte documental del presente trabajo, a los que acompañan varios apéndices.

- El capítulo **Estado del Arte** aborda lo más relevante de la evolución de los distintos actores del proyecto desde su aparición hasta la actualidad sin perder la atención en el objeto de este proyecto. Para ello se presentan tres secciones que corresponden con sendos actores: EMV®, NFC y Android.

- En el capítulo **Conocimientos Previos**, al igual que en el anterior, la información se presenta desglosada por secciones correspondientes a sendos componentes. En este caso, sin embargo, la información que se trata tiene un carácter marcadamente más técnico y es sobre la que se apoya teóricamente el presente proyecto.

- El capítulo **Metodología** documenta las herramientas y material utilizado en la realización del proyecto así como el planteamiento inicial del escenario a considerar.

- El capítulo **NFC Playground Suite** trata de los desarrollos realizados en el marco de este proyecto para servir de prueba de concepto en los escenarios que se desea estudiar. Los detalles relativos a la arquitectura, diseño y guía de uso se tratarán de forma directamente proporcional a la relevancia y envergadura del desarrollo.

- El capítulo **Resultados experimentales** documenta las sesiones establecidas según el perfil de tarjeta utilizado.

- Con base en los resultados de las pruebas, el capítulo **Conclusiones, limitaciones y líneas futuras** incluye las inferencias de las pruebas realizadas así como una mirada a las siguientes líneas de trabajo de esta investigación.

Capítulo 2

Estado del Arte

2.1 EMV®

Recibe su nombre de las tres organizaciones que crearon estas especificaciones en 1996 —Europay, MasterCard y Visa—. Implementa la norma **ISO/IEC 7816** por lo que define los requisitos de interoperatividad entre aplicaciones de pago basadas en circuitos integrados —en adelante *chip*— y terminales de pago[1].
En la siguiente figura puede comprobarse el momento en el que la tecnología basada en EMV® se incorpora al ciclo evolutivo de los medios de pago y cómo el 2008 marca un punto de inflexión al liberar unas nuevas especificaciones basadas en medios de pago *contactless*[2]:

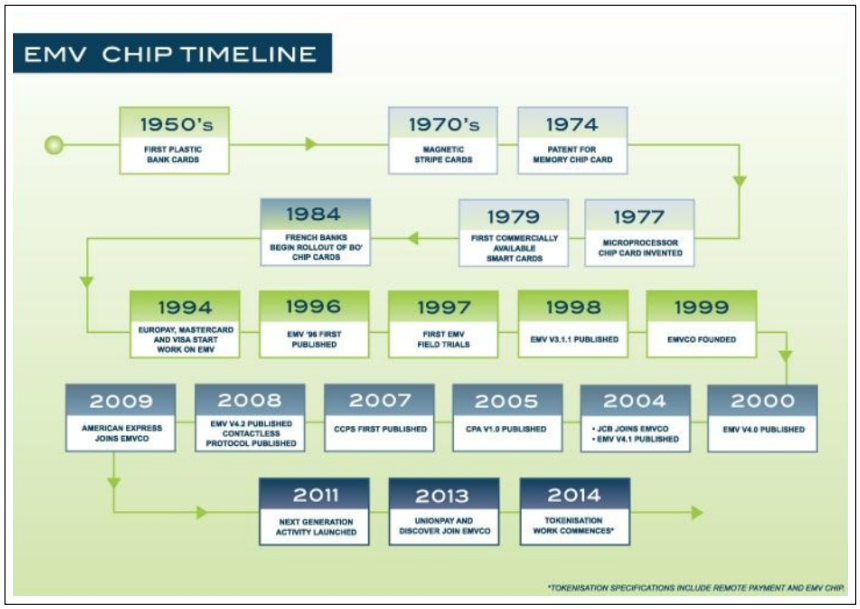


Figura 2.1: Detalle de la evolución de la tecnología asociada a tarjetas bancarias de pago

[1] El término *terminales de pago* hace referencia a terminales atendidos y desatendidos de punto de venta así como cajeros automáticos
[2] A los efectos de este estudio, equivalente a inalámbrico

La aparición de EMV® en el sector tuvo un efecto aglutinador ya que hasta entonces, particularmente en las décadas de los años 1980 y 1990, los despliegues de sistemas electrónicos de pago se basaban en especificaciones domésticas de mercado que en conjunto presentaban problemas de escalabilidad e interoperatividad.

En el capítulo anterior se hacía referencia por primera vez a las ventajas de EMV® en términos de seguridad. Las tarjetas emitidas atendiendo a estas especificaciones fueron las primeras en incorporar *chips*, que no son memorias con zonas de lectura y/o escritura al uso sino microprocesadores, cuyas aportaciones principales a los entornos de pago se podrían resumir en los siguientes tres puntos:

1. Realiza funciones de **procesamiento de información**.
2. Almacena información confidencial de forma **segura**.
3. Realiza operaciones de **aritmética criptográfica**.

Todos los elementos anteriores han posibilitado que el nivel de adopción de EMV® supere ampliamente al resto de alternativas actuales adoptando una posición dominante en el mercado tal y como refleja el siguiente gráfico:

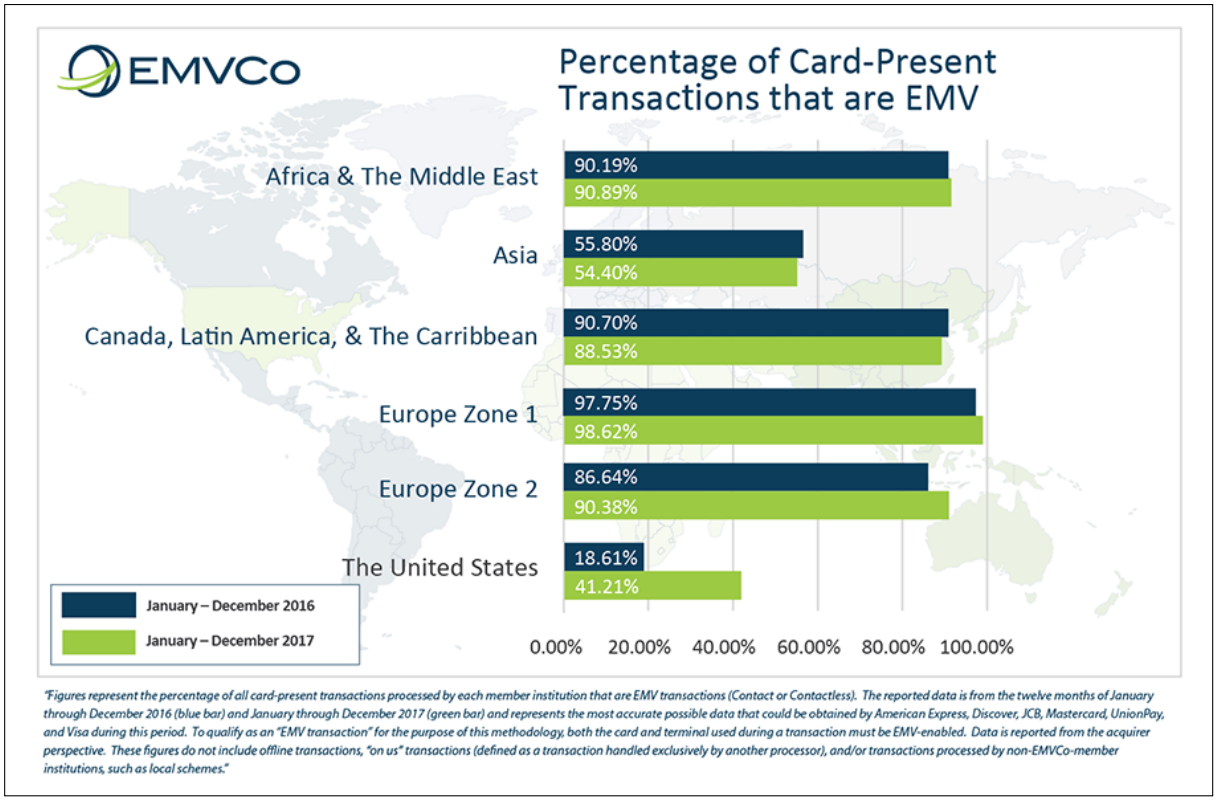


Figura 2.2: Porcentaje de operaciones de tarjeta que fueron EMV® entre 2016 y 2017 [8]

2.2 NFC

Aunque como se ha comentado anteriormente la relación entre NFC y los medios de pago no es causal sino que responde técnicamente a una concomitancia circunstancial, lo cierto es que dentro de los sistemas inalámbricos de proximidad de transferencia de fondos se ha consolidado como el protocolo de comunicación de referencia.

Otras alternativas se han basado por ejemplo en sistemas de *tokenización*[3] en los que una App obtendría un *token* con fecha de caducidad de un sistema externo que una vez convertido a algún código bidimensional e.g. BIDI, QR, DATAMATRIX... y empleado en el pago de la transacción provocaría la *destokenización* del *token* a través de la pasarela de pagos completando así la operación.

Figura 2.3: Distintos códigos bidireccionales empleados como alternativa a NFC[4]

Dentro de los medios de pagos basados en NFC, los tres competidores consolidados son Apple con *Apple Pay*[5], Google con *Google Pay*[6] y Samsung con *Samsung Pay*[7] .
Apple Pay, presentado el 9 de septiembre de 2014, vio su salida a producción demorada debido a que su buque insignia del momento, el IPhone 6, ni siquiera incorporaba la circuitería necesaria para soportar NFC. No obstante, salvadas las limitaciones físicas no tardó en plantear una salida del mercado norteamericano en busca de nuevos mercados de la mano de American Express.

Google Pay fue presentado en septiembre de 2011. Pese a las expectativas y en contra de los pronósticos del sector se ha visto desplazado por *Samsung Pay*, lanzado tres años después, en agosto de 2015. El gigante asiático continuó la línea de trabajo acertada y austera que ha seguido los últimos años consiguiendo una posición más favorable que sus competidores en mercados europeos tras conseguir cerrar acuerdos claves tanto en el sector financiero como en el de *retail* (comercialización masiva de productos y/o servicios a grandes cantidades de clientes). Buena prueba de ello son las empresas DIA, El Corte Inglés o CEPSA que fueron las primeras en integrar el *wallet*[8] de Samsung.

[3] `https://www.paymetric.com/solutions/tokenization/`
[4] `http://www.codigos-qr.com`
[5] `https://www.apple.com/es/apple-pay/`
[6] `https://pay.google.com`
[7] `https://www.samsung.com/es/samsung-pay/`
[8] Se entiende por *wallet* el *software* que permite realizar transacciones electrónicas de fondos

2.3 Android

El móvil es uno de los dispositivos electrónicos al que más horas dedicamos[9], cuenta con conexión a Internet y es el que nos acompaña prácticamente en todo momento del día, tres razones que lo convierten en el anfitrión estratégicamente perfecto para medios de pago. Como se señalaba anteriormente, **la versión 4.4 de Android (KitKat)** -ver **Tabla 2.1**- supone un antes y un después en el horizonte de posibilidades en cuanto a medios de pago.

Nombre	Versión	Kernel	Fecha de Liberación	API	Parches
Petit Four	1.1	2.6.X	09/02/2009	2	No
Cupcake	1.5	2.6.27	27/04/2009	3	No
Donut	1.6	2.6.29	15/09/2009	4	No
Eclair	2.0-2.1	2.6.29	26/10/2009	5-7	No
Froyo	2.0-2.2.3	2.6.32	20/05/2010	8	No
Gingerbread	2.3-2.3.7	2.6.35	06/12/2010	9-10	No
Honeycomb	3.0-3.2.6	2.6.36	22/02/2011	11-13	No
Ice Cream Sandwich	4.0-4.0.4	3.0.1	18/10/2011	14-15	No
Jelly Bean	4.1-4.3.1	3.31-3.4.39	09/07/2012	16-18	No
KitKat	**4.4-4.4.4**	**3.10**	**31/10/2013**	**19-20**	**No**
Lollipop	5.0-5.1.1	3.16.1	12/11/2014	21-22	No
Marshmallow	6.0-6.0.1	3.18.10	05/10/2015	23	Si
Nougat	7.0-7.1.2	4.1	22/08/2016	24-25	Si
Oreo	8.0-8.1	4.10	21/08/2017	26-27	Si
Android P	9	4.14?	06/06/2018 (beta 2)	28	En beta
Android Q	10?	4.X	Preview de desarrolladores	?	No?

Tabla 2.1: Historial de versiones de Android

Atendiendo al reparto de las versiones anteriores sobre el parque de móviles Android puede comprobarse cómo el 90 % de terminales opera con una versión superior o igual a la **4.4**:

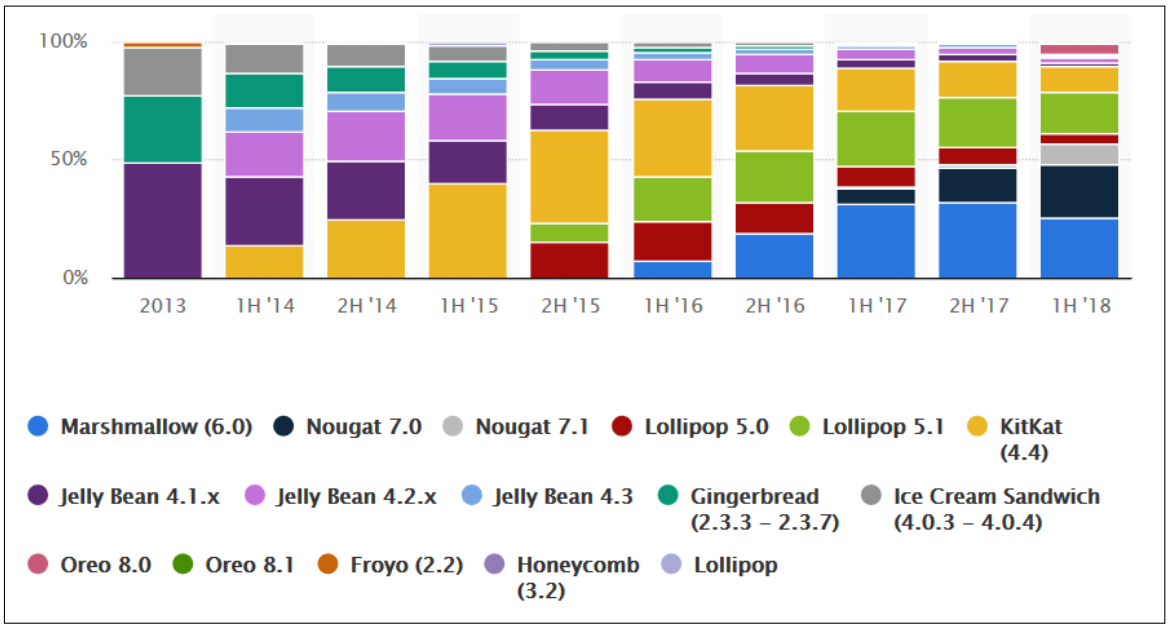


Figura 2.4: Distribución de versiones Android del parque de terminales según año [9]

[9] http://www.nielsen.com/us/en/insights/reports/2017/the-nielsen-total-audience-q2-2017.html

En el siguiente gráfico puede apreciarse el detalle del volumen de teléfonos inteligentes que llegaron al mercado mundial por año en millones de unidades según sistema operativo. Se aprecia una **una posición claramente dominante del parque móvil Android** frente a su mayor rival, IOS, que se traduce en la figura **Figura 2.5**

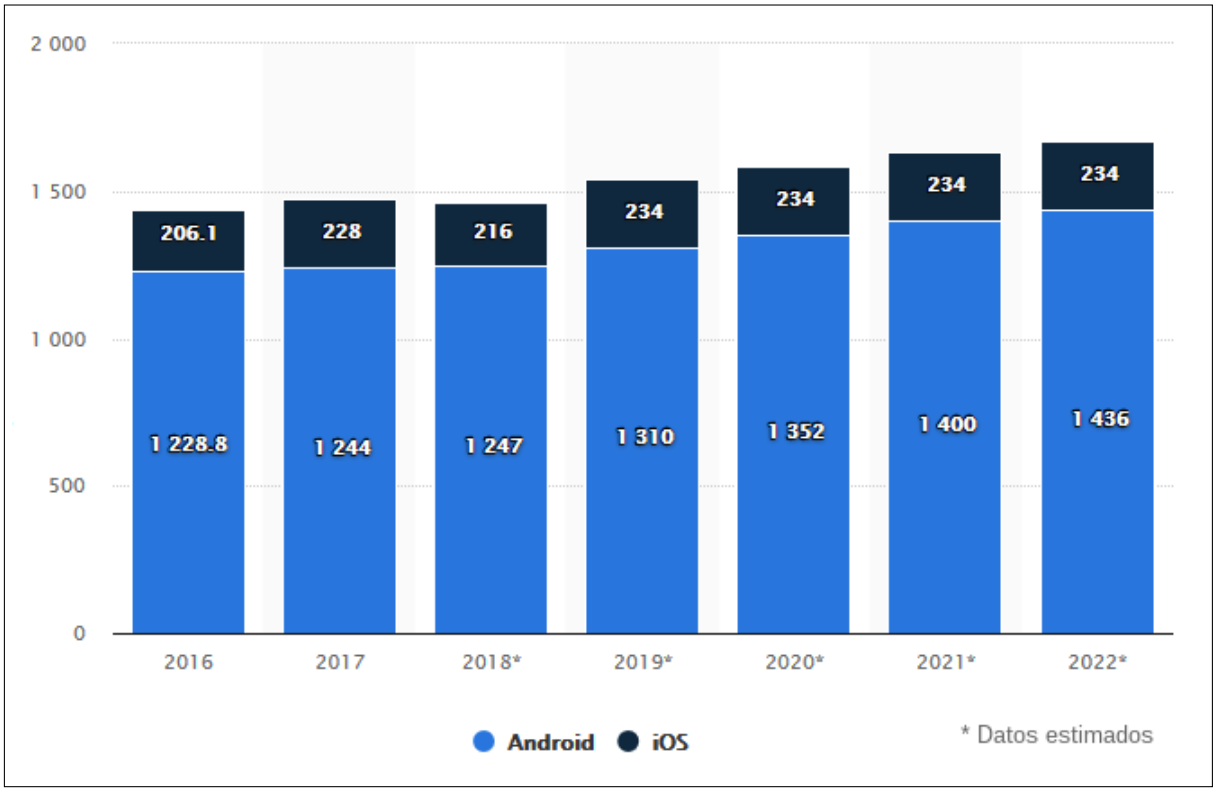


Figura 2.5: Distribución de sistemas operativos por móvil en millones de unidades y por año [10]

Antes de 2013, como se verá en el siguiente capítulo, no existía la posibilidad de plantear un diálogo entre un punto de venta y un móvil sin contar con la presencia de un **elemento seguro -SIM** o similiar- en este último que validara la sesión. HCE supone un cambio de paradigma en la manera de gestionar la interlocución entre cliente y terminal de pago en entornos móviles.

Por tanto disponer de la posibilidad de emular tarjetas de crédito o débito por software en un sistema Android con una versión igual o superior a ***KitKat*** significa a día de hoy llegar a miles de millones de compradores en potencia y esto, en términos de capacidad de penetración y adopción de una tecnología puede llegar a marcar la diferencia del futuro de *Apple Pay* o *Samsung Pay* en favor de *Google Pay*.

Capítulo 3

Conocimientos Previos

3.1 ISO/IEC 7816

La norma ISO/IEC 7816 consta de distintas partes que bajo el título genérico de *Identification cards - Integrated circuit cards* cubren todos los aspectos de interoperatividad desde la capa física a la de aplicación en el marco de entornos de pago con tarjeta.
Aunque se publicaron inicialmente en 1987 [11], han tenido lugar revisiones hasta las más recientes con fecha de 2017:

Parte	Nombre	Referencia
1	*Tarjetas con contactos: Características físicas*	[12]
2	*Tarjetas con contactos: Dimensión y localización de los contactos*	[13]
3	*Tarjetas con contactos: Interfaz eléctrica y protocolos de transmisión*	[14]
4	***Organización, seguridad y comandos para el intercambio***	**[15]**
5	*Registro de proveedores de aplicación*	[16]
6	*Elementos de datos interindustria para el intercambio*	[17]
7	*Comandos interindustria para el Structured Card Query Language* (SCQL)	[18]
8	*Comandos para operaciones de seguridad*	[19]
9	*Comandos para la gestión de tarjetas*	[20]
10	*Tarjetas con contactos: Señales electrónicas y ATR para tarjetas síncronas*	[21]
11	*Verificación personal mediante métodos biométricos*	[22]
12	*Tarjetas con contactos: Interfaz eléctrica USB y procedimientos operativos*	[23]
13	*Commandos para la gestión de aplicación en un entorno multiaplicación*	[24]
15	*Aplicación de información criptográfica*	[25]

Tabla 3.1: Partes de la norma ISO/IEC 7816

Pese a que es evidente que todas las partes de la norma tienen relevancia en el ecosistema de medios de pago y que definen el contexto original que presidirá este trabajo, para este estudio resulta de especial interés la ***Parte 4*** **[15]** — de ahí que en este documento se haga referencia a la norma ISO/IEC 7816-4 — ya que es la que recoge el conjunto de reglas de diálogo en la comunicación con el integrado de la tarjeta y que serán comunes a escenarios inalámbricos y de

contactos. De su interpretación se infieren las especificaciones del ***Libro 3: Especificaciones de Aplicación [5]*** de EMV®, que como se vio en **Figura 2.2** es la implementación comercial de la norma ISO/IEC 7816 que domina el mercado actual de medios de pago con tarjeta y el protocolo de aplicación sobre el que se centra el estudio.

En la **Figura 1.1** se describía el esquema del escenario inicial, en el que en la parte superior derecha se puede apreciar una tarjeta EMV® y su lector de tarjetas inteligentes —ver capítulo 4—. En esta parte del sistema, por su naturaleza, la pila de especificaciones de la sesión contemplará la parte correspondiente a una sesión de contactos según queda recogido en el ***Libro 1: Application Independent ICC to Terminal Interface Requirements [26]*** de las especificaciones de EMV®.

Como se señalaba anteriormente, el circuito integrado de la ICC no es una memoria al uso. Si además la tarjeta presenta funcionalidad *contactless* como puede apreciarse en el siguiente diagrama, el nivel de tecnología que incorpora la pequeña superficie de plástico se incrementa al añadir un circuito electromagnético que hará de antena para la comunicación sobre NFC, y lo que resulta más interesante aunque no a efectos del presente estudio: el propio circuito integrado incorpora su propia antena para comunicarse por radio con la antena principal.

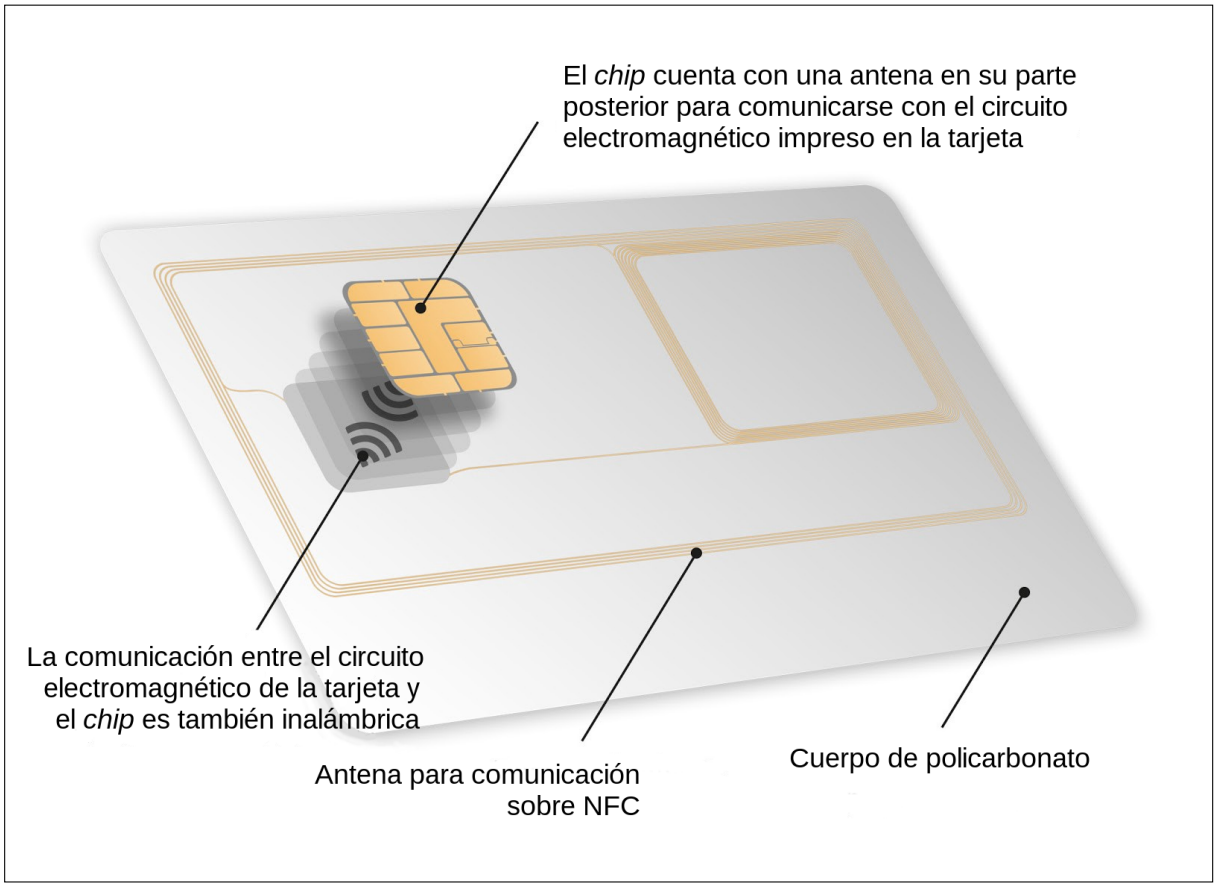


Figura 3.1: Vista de la tecnología de una tarjeta EMV® con soporte NFC

3.1.1 Fases de una sesión EMV®

1. Inserción de la ICC (Integrated Circuit Card) en el IFD (Interface Device) y conexión y activación de los contactos.
2. Inicialización de la ICC y establecimiento de la comunicación entre la ICC y el terminal.
3. Ejecución de las transacciones.
4. Desactivación de los contactos y extracción de la ICC.

La primera de las fases tiene lugar a nivel de capa física y excede del ámbito de este documento por razones de concisión, aunque puede consultarse en el punto 6.1.2 del ***Libro 1: Application Independent ICC to Terminal Interface Requirements [26]***. Se presenta sin embargo el diagrama de diseño de una ICC para, al menos, entender la relación entre el juego de señales y los contactos físicos de la ICC. Como se verá a continuación, esta comunicación es síncrona y atiende a una señal de reloj ofrecida por el terminal de una frecuencia comprendida entre 1 MHz y 5 MHz [26].

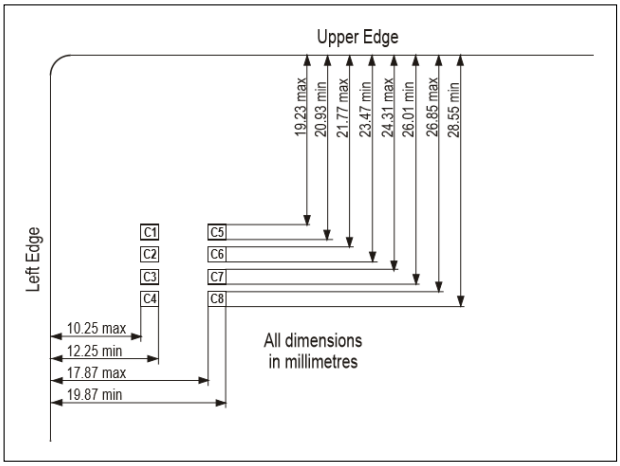


Figura 3.2: Distribución de los contactos de una ICC [26]

C1	Alimentación (VCC)	**C5**	Tierra (GND)
C2	Reset (RST)	**C6**	Reservado Futuro
C3	Reloj (CLK)	**C7**	Entrada/Salida (I/O)
C4	Sin uso	**C8**	Sin uso

Tabla 3.2: Relación de contactos de una ICC [26]

Cuando se ha satisfecho esta primera fase de la sesión, se entiende que la ICC está presente y receptiva a la comunicación. A partir de aquí el terminal iniciará la segunda fase, primero intentando lo que se conoce como *Cold Reset*, un proceso en el que mediante distintas combinaciones de las entradas *RST*, *CLK*, *VCC* y *I/O* buscará obtener de la ICC una respuesta llamada **Answer To Reset (ATR)** que por una parte sirva de notificación de operatividad y por otra parte defina los parámetros básicos de la comunicación a nivel de aplicación tales como la convención —si la lógica que se seguirá es directa o inversa—, la orientación —orientado a bloque o a byte—... En caso de que la respuesta al *Cold Reset* no sea satisfactoria se volverá a intentar obtener el ATR, modificando ligeramente el juego de señales enviadas por el terminal, en un modo que se conoce como *Warm Reset*. Si el problema persiste, el terminal dará por abortada la comunicación y se procederá a la desactivación de los contactos.

3.1.2 Estructura de los mensajes

Para poder abordar la tercera fase es necesario entender la codificación de los mensajes intercambiados. Según se recoge en [5], la comunicación en esta fase tiene lugar en forma de comandos y respuestas llamados **APDU (*Application Protocol Data Unit*)**. El formato del comando es el siguiente, en el que los primeros 4 bytes son obligatorios y conformarían la cabecera y el resto son condicionales:

CLA	INS	P1	P2	Lc	Data	Le

Tabla 3.3: Formato de un comando APDU [5]

Nombre	Descripción	Bytes
CLA	Clase de instrucción	1
INS	Código de instrucción	1
P1	Parámetro 1 de instrucción	1
P2	Parámetro 2 de instrucción	1
Lc	Nº de bytes del campo Data del comando	0 o 1
Data	Cadena de bytes de longitud Lc	variable
Le	Nº máximo de bytes del campo Data de la respuesta	0 o 1

Tabla 3.4: Campos de un comando APDU [5]

Por su parte, el formato y detalle del APDU de respuesta sería el que se define en las siguientes tablas. En este caso el campo Data formaría la cabecera y SW1 y SW2 la cola:

Data	SW1	SW2

Tabla 3.5: Formato de una respuesta APDU [5]

Nombre	Descripción	Bytes
Data	Bytes de datos recibidos como respuesta	variable (Lr)
SW1	Estado del procesamiento del comando	1
SW2	Calificador del procesamiento del comando	1

Tabla 3.6: Campos de una respuesta APDU [5]

Esta secuencia de comandos-respuesta puede ocurrir únicamente dentro de cuatro escenarios que se resumen en la siguiente tabla:

Tipo	Escenario
1	Comando: *Cabecera* Respuesta: *Cola*
2	Comando: *Cabecera* + *Le* Respuesta: *Data* + *Cola*
3	Comando: *Cabecera* + *Data* Respuesta: *Cola*
4	Comando: *Cabecera* + *Data* + *Le* Respuesta: *Data* + *Cola*

Tabla 3.7: Tipos de escenario en un intercambio de APDUs

3.1.3 Estructura de datos

La manera de estructurar los comandos de la sesión entre la ICC y el terminal se ajusta al lenguaje de notación formal **ASN.1**[1]. La información ASN.1 se codifica siguiendo la estructura **BER-TLV** (*Basic Encoding Rules - Tag Length Value*) —en adelante TLV—, según la cual cada objeto de datos consiste en un *tag*, un byte de longitud y la información a comunicar en cuestión.

Supongamos por ejemplo que quisiéramos estructurar en formato TLV la siguiente información ASN.1:

Secuencia
{
Nombre::= "jotaca1"
Patrimonio::= "0"
Vegetariano::= "false"
Fumador::= "true"
}

Siguiendo la leyenda `Tag Longitud Valor`, imaginemos que disponemos de la siguiente representación en hexadecimal de los tags:

Tag	Significado
30	Secuencia
0C	String UTF-8
02	Entero
01	Booleano

Tabla 3.8: Tags TLV para el ejemplo

Con todo lo anterior ya sería posible estructurar la información ASN.1 del ejemplo según TLV. La información quedaría de la siguiente manera:

```
30 12 0C 07 6A 6F 74 61 63 61 31 02 01 00 01 01 00 01 01 01
```

donde puede apreciarse la recursividad de la propia estructura, *e.g.* el `Valor` del primer objeto a su vez tiene una *metainterpretación* TLV que se desglosa en otros objetos TLV, que pueden a su vez pueden seguir contemplando recursividad o convivir al mismo nivel:

Tag Longitud Valor	Interpretación
30 12	Secuencia de 18 bytes de longitud
0C 07 6A6F7461636131	String UTF-8 de 7 bytes de longitud con el valor *jotaca1*
02 01 00	Entero de 1 byte con el valor 0
01 01 00	Booleano de 1 byte con valor *false*
01 01 01	Booleano de 1 byte con valor *true*

Tabla 3.9: Información estructurada según TLV

[1] https://www.itu.int/en/ITU-T/asn1/Pages/introduction.aspx

3.1.4 Acceso a la información

Puesto que ya se ha descrito tanto el modelo de comunicación como el modelo de datos en el que se estructura dicha comunicación, se retomará el ciclo de vida de una sesión EMV® en su fase 3. El siguiente paso es el de la **Selección de la Aplicación**. No solo la información asociada de la comunicación está estructurada. La norma ISO/IEC 7816 [15] exige que la información contenida en el circuito integrado de la tarjeta también tenga una estructura particular. Como puede apreciarse en el siguiente diagrama, la información se estructura en forma de árbol sobre dos objetos de datos: *Archivos Dedicados* **(Dedicated Files - DF**) y *Archivos Elementales* (**Elementary Files - EF**). Según las especificaciones [15] debe existir al menos un DF raíz llamado *Archivo Maestro* (**Master File - MF**):

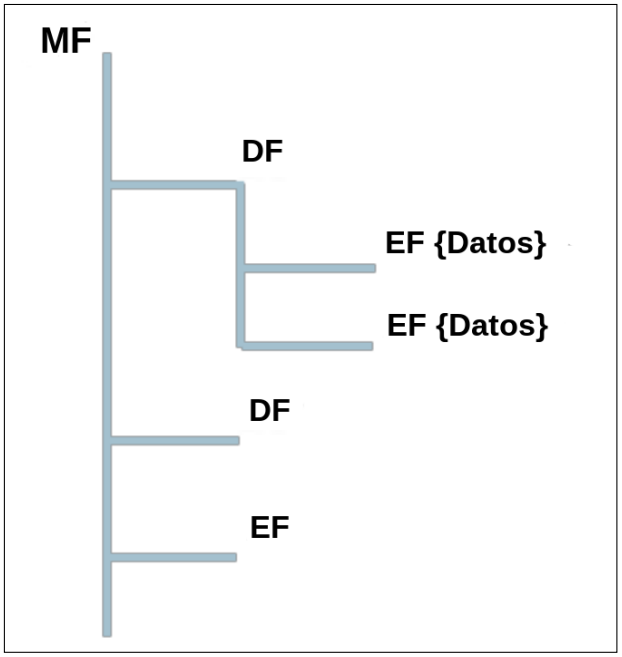


Figura 3.3: Estructura de la jerarquía de archivos según la norma ISO/IEC 7816-4 [15]

Es en estas ubicaciones donde el circuito integrado guarda, entre otras, la información relativa a las aplicaciones que soporta la tarjeta —si esta es multiaplicación, como suele ocurrir habitualmente— que dependerá del emisor en primera instancia —*VISA*, *MasterCard*, *American Express*...— y del banco en segundo lugar. Las aplicaciones se registran en el circuito integrado mediante un identificador único llamado ***Application Identifier*** **(AID)**. En esta tercera fase el proceso de selección de aplicación lo que pretende es seleccionar el ***Application Definition File*** (**ADF**) en particular de nuestra aplicación para poder operar con ella.

Existen dos formas de dirigir las peticiones en la conversación con el circuito integrado para obtener el **ADF**: haciéndolo hacia a un entorno de sistema de pago —***Payment System Environment*** (**PSE**) para contactos y ***Proximity Payment System Environment*** (**PPSE**) en entornos inalámbricos— o dirigiéndolas a una **AID** en particular.

Así, dentro del **MF** existen uno o varios **Application Definition Files (ADF)** que a su vez contienen uno o mas **AEF**s (*Application Elementary Files*), que a su vez contienen diferentes registros entre los que eventualmente se encuentra la información asociada al **AID**.

Una vez seleccionado el **ADF**, lo que sigue es el proceso de Inicio de Aplicación (***Initiate Application***), que inicia el grueso del flujo de una transacción en EMV®. Por comodidad se representa el diagrama de flujo de transacción en la figura de la siguiente página.

3.1.5 Floor Limit

En la sección **1.3 Objetivos** se ponía el foco por primera vez en la funcionalidad **Floor Limit** y las implicaciones que podía tener para el presente estudio. La toma de decisiones en lo tocante a esta funcionalidad tiene lugar en el proceso coloreado en azul —**Terminal Risk Management**— en la siguiente figura:

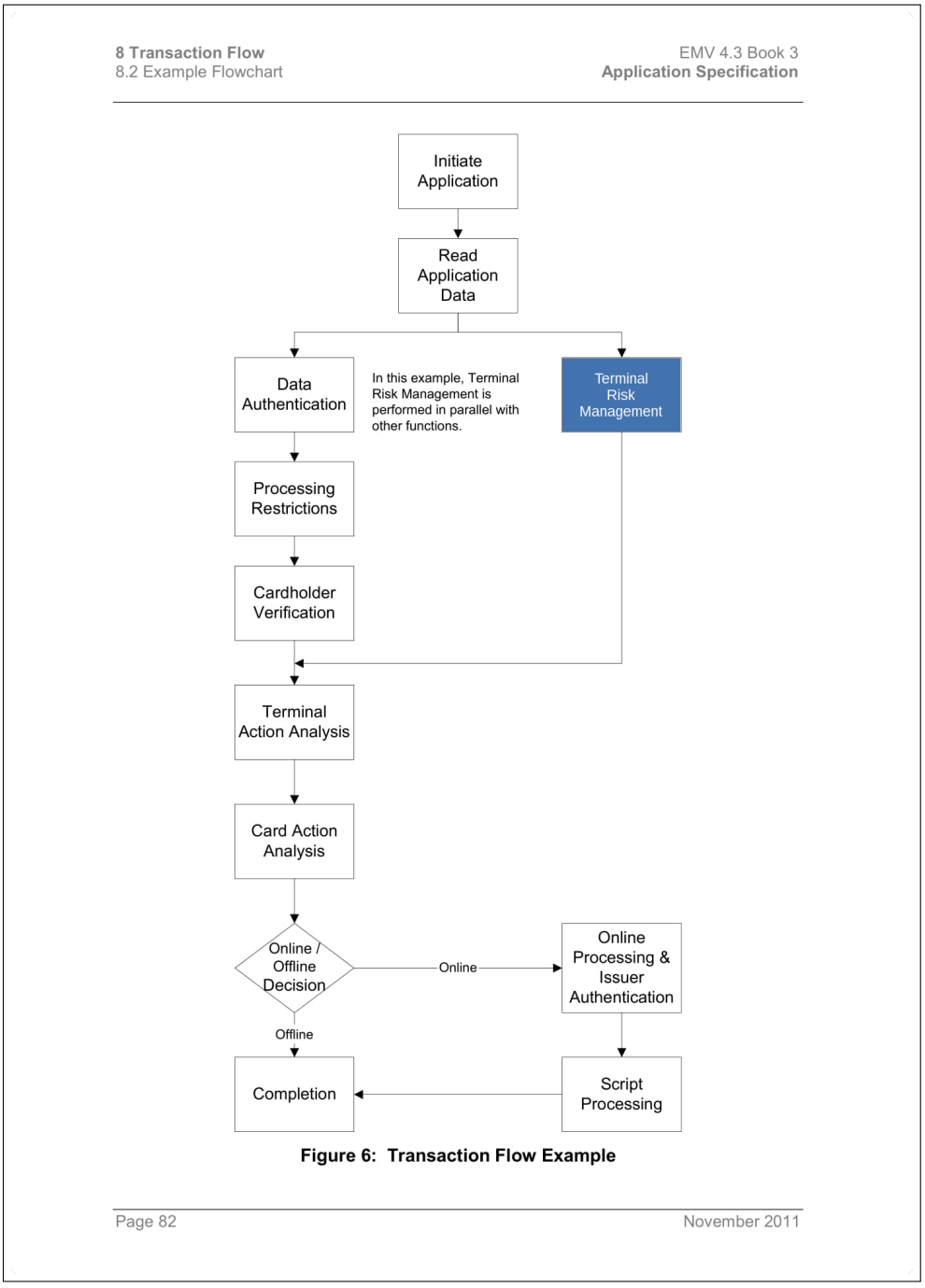


Figura 3.4: Ejemplo de flujo de una una transacción EMV® [5]

Este proceso tiene como cometido la gestión de *Floor Limit*, la selección aleatoria de transacción[2] y la comprobación de la velocidad[3]. Resulta fácil comprobar cómo en caso satisfactorio el flujo es transferido al terminal —*evitando* dadas las circunstancias la verificación de titular— para que implemente el análisis de acción, esto es, si la operación se aprobará *offline*, se denegará *offline* o si retransmitirá *online*. Hay que tener presente que de acuerdo con el informe de 2016 de Visa Europa[4] [27] el importe medio de operación fue de 13.83 € tras verse incrementado en un 12 % respecto al año anterior.

3.1.6 Análisis temporal

Una de las dudas más razonables en cuanto a condiciones limitantes en el presente estudio tiene que ver con la gestión de tiempos. EMV® define tiempos de transmisión, recepción y bit de cualquiera de las fases de la sesión. Aunque como quedará demostrado en el siguiente capítulo estas ventanas son más restrictivas bajo la implementación de la norma ISO/IEC 14443, se documentará para EMV® a efectos de haberse tenido en consideración como factor limitante en potencia.

En EMV® el tiempo de *bit* empleado en la linea de entrada/salida (E/S) se define como ***elementary time unit (etu)*** [12]:

$$etu = \frac{372}{f} \tag{3.1}$$

donde f es la frecuencia en hercios (Hz) y dado que este estudio trabaja bajo lo dispuesto en la norma ISO/IEC 14443,

$$f = 13.56 \text{ MHz}$$

resultando $\boldsymbol{etu = 27.43\ \mu s}$

De acuerdo con [12] el tiempo más crítico es el tiempo inicial de recepción del *Answer to Reset* (ATR), teniendo que completarse (bien mediante *cold reset* o *warm reset*) en un **máximo de 19 200 *etus* (526 725 μs o 0.52 s aproximadamente)**.

La intención de este estudio se centrará en evaluar si en el escenario elegido —**Figura 1.1**— es posible completar un camino alternativo a la verificación de usuario a través de la gestión de la funcionalidad *Floor Limit* durante el proceso **Terminal Risk Management** hasta llegar a la fase de completar la operación sobre una sesión EMV® basada en contactos.

A título informativo, en el **Apéndice A.2** se presentan dos tablas con una relación del importe de *Floor Limit* a nivel mundial.

[2]Según se recoge en el apartado 10.6.2 de [5], toda transacción que reúna ciertas condiciones relacionadas con el importe y dos parámetros —*Target Percentage to be Used for Random Selection* y *Maximum Target Percentage to be Used for Random Selection*— pueda ser elegida para resolverse *online* con independencia del resultado de la gestión de riesgo asociada a *Floor Limit*.

[3]Según se recoge en el apartado 10.6.3 de [5], esta comprobación permite al emisor solicitar la resolución *online* de la transacción dada la concurrencia de cierto número de transacciones.

[4]`https://www.visaeurope.com`

3.2 ISO/IEC 14443

El conjunto de estándares NFC [28], como subconjunto de tecnologías **RFID**[5] (*Radio Frequency Identification*), abarca un campo vasto de tecnología inalámbrica asociada a ICCs, por lo que son varias las bases normativas sobre las que despliega su ámbito de alcance. Para este estudio, únicamente resulta de interés su herencia por parte de la norma ISO/IEC 14443 que entre otros es estándar en Europa y EE. UU., aunque es preciso señalar que parte de la familia NFC se ve influida por otras normas o estándares como **ISO/IEC 18092** [29] o **FeliCa**[6].

La norma ISO/IEC 14443 se divide en 4 partes que definen todos los requisitos y reglas en el marco de la comunicación inalámbrica de proximidad según puede apreciarse en la siguiente tabla:

Parte	Nombre	Referencia
1	*Características físicas*	[30]
2	*Alimentación por radiofrefuencia (RF) e interfaz de señal*	[31]
3	*Inicialización y anticolisión*	[32]
4	*Protocolo de transmisión*	[33]

Tabla 3.10: Partes de la norma ISO/IEC 14443

Aunque define dos tecnologías —A y B— que difieren en la estructuración de sus comandos operativos, modulación o modelo de inicialización entre otros, ambas trabajan a la misma frecuencia (**13.56 MHz**). La razón de esta distinción hay que buscarla en el ámbito de trabajo para el que originalmente fueron concebidos el estándar A —memorias— y el B —circuitos integrados con capacidad de computación— , aunque la distinción es actualmente irrelevante a efectos de prestaciones físicas[7]. Resultan de especial interés para este estudio las **Partes 3 [32]** y **4 [33]** de la norma en la medida en que ayudan a comprender otra de las fortalezas de NFC-EMV en entornos de pago.

Como ya se ha señalado con anterioridad, pese a que durante un período determinado parte del sistema invade un medio vulnerable y compartido, el rango de proximidad efectivo con el que se trabaja bajo esta norma está por debajo de los 10 cm y en función del escenario incluso por debajo de los 2 cm, aunque como se verá en breve no es el único mecanismo para garantizar la unicidad de la comunicación. Uno de los detalles operativos más relevantes de este estándar es la posibilidad de que uno de los dos interlocutores sea un dispositivo pasivo, esto es, que sea alimentado por inducción eletromagnética por el otro interlocutor —dispositivo activo—.

3.2.1 Inicialización

Los actores de esta comunicación son por una parte el **PCD** (*Proximity Coupling Device*) que es el dispositivo activo[8] y por otra el **PICC** (*Proximity Integrated Circuit Card*). El modo de operación durante la fase de vinculación que se recoge en la Parte 3A de la norma responde a un sondeo por parte del PCD, quien mediante la difusión de comandos **`REQA`** (*Request Type-A*) esperará respuestas de PICCs en su rango de alcance en forma de comando **`ATQA`** (*Answer To Request*).

[5] `http://www.ieee-rfid.org/the-ieee-journal-of-rfid`
[6] `http://www.felicatech.org/en/index.html`
[7] `https://www.secureidnews.com/news-item/is-the-debate-still-relevant-an-in-depth-look-at-iso-14443`
[8] En entornos EMV® nos referiremos a él como *terminal*, rol que interpretará el *PIN Pad* o *datáfono* —ver **Sección 4.1**—.

En el caso de encontrar varios contendientes activará un **protocolo anticolisión** basado en el filtrado gradual del identificador (*UID*) del PICC hasta que únicamente responda uno, momento en el que el PICC elegido enviará un comando **`SAK`** (*Select Acknowledge*) incluyendo una presentación y una relación de parámetros y protocolos soportados por su parte.
Para poder dar por establecido el canal de comunicación entre ambos interlocutores debe concluir el proceso que se ilustra en la siguiente figura, en el que ante la recepción del **`SAK`** en cuestión el PCD enviará el comando **`RATS`** (*Request for ATS*) solicitando más información al PICC, que en caso satisfactorio será respondido con el comando **`ATS`** (*Answer To Select*) donde se incluirá información relativa al dispositivo. Una vez finalizado el proceso, habrá quedado establecido

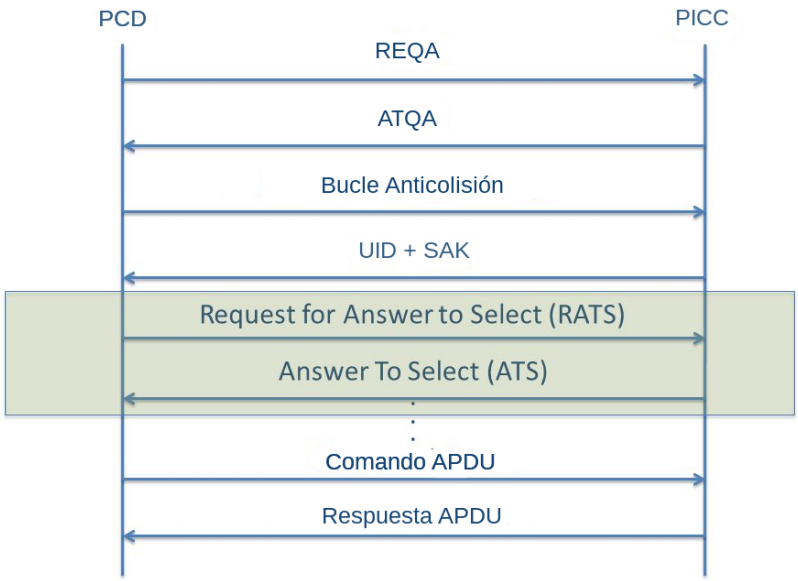


Figura 3.5: Inicialización del canal según la norma ISO/IEC 14443-3 [32]

un canal de comunicación ***half-duplex***[9] sobre el que se desarrollará la sesión EMV® con sus APDUs en forma de comandos y respuestas. Por tanto, **la implementación de este protocolo obliga a quedar fuera de la comunicación a cualquier tercer actor no legítimo en la conversación**.

3.2.2 Modelo operativo

Inspirado en la norma ISO/IEC 14443, NFC contempla tres modos de trabajo:

1. **Modo lectura/escritura**: También conocido como iniciador, en el que el dispositivo adopta el rol activo con capacidad de leer o escribir *tags*[10] [29].

2. **Modo Peer-to-Peer**: Ambos dispositivos adoptan un rol activo permitiendo una comunicación bidireccional.

3. **Modo Emulación**: El dispositivo adopta un rol pasivo —*tag*— permitiendo la lectura pero no la escritura. Este es el modo sobre el que se trabajará en HCE y por tanto en este estudio.

[9] Por tanto los interlocutores de este estudio cuya comunicación se base en NFC solo podrán estar en modo transmisión o recepción, pero nunca a la vez.
[10] Dispositivos que implementan NFC en modo pasivo y/o son compatibles con la norma ISO/IEC 18092

Según el NFC Forum, se definen 4 tipos de *tag* NFC en función de sus prestaciones. Se presenta una tabla resumen con las principales diferencias, aunque existen otras diferencias basadas en capacidad de almacenamiento, persistencia del almacenamiento o la velocidad de comunicación:

Características	**Tipo 1**	**Tipo 2**	**Tipo 3**	**Tipo 4**
Protocolo Anticolisión	No	Sí	Sí	Sí
Disposición en modo *Solo Lectura*	Sí	Sí	Sí	Sí
Identificador (UID)	Sí	Sí	No	Sí
Precio	Bajo	Bajo	Moderado	Elevado

Tabla 3.11: Tipos de *tags* definidos en NFC

La información a transportar por el *tag* se guarda atendiendo a un formato específico. Así, podremos encontrarnos ante aplicaciones **NDEF** (***NFC Data Exchange Format***) o **NO-NDEF** en función de si estas aplicaciones NFC respetan el formato definido en [34] o si por el contrario nos encontramos ante formatos desarrollados en ámbitos propietarios. Para situar ambos paradigmas en la pila interna de NFC se presenta el siguiente diagrama:

Figura 3.6: Ubicación de NDEF en el modelo operativo de NFC

Con todo lo anterior queda definida toda la parte del ecosistema inalámbrico sobre el que descansa el marco de operación de medios de pago contactless que nos ocupa. EMV® ocuparía el lugar de la capa de Aplicación sobre la pila completa de protocolos en una comunicación sobre NFC —**Figura 3.7**— donde además se puede destacar el uso de **LLC (*Logical Link Control*)** como protocolo de la capa de Enlace de Datos por encima de *Digital Protocol*, que es el protocolo de bajo nivel definido en los estándares NFC *NFCIP1* [29] y *NFCIP2* [35].

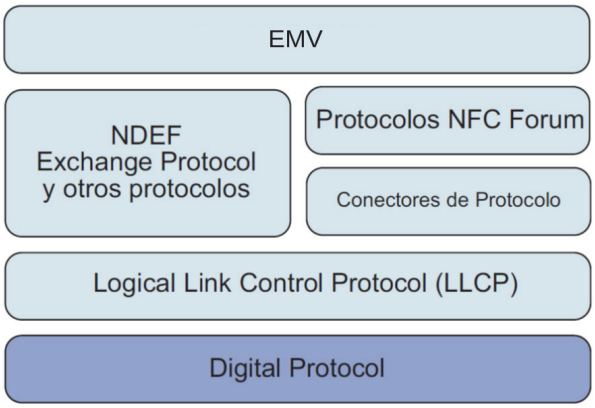


Figura 3.7: Pila de protocolos de una sesión EMV® en un modelo de comunicación basado en NFC

3.2.3 Análisis temporal

De forma análoga a la consideración de las restricciones temporales que se hizo en EMV® a continuación se estudiarán las limitaciones que pueden impactar en este estudio derivadas de la implementación de una comunicación sobre NFC y TCP/IP —ver **Figura 1.1**— en la que distintos componentes introducirán un retardo efectivo. Para el caso de NFC, son dos los tiempos a tener en especial consideración: El tiempo entre frames en las fases de inicialización y de anticolisión o ***Frame Delay Time*** **(FDT)** y el tiempo de espera de respuesta del PCD a partir de ese momento o ***Frame Waiting Time*** **(FWT)**.

El *Frame Delay Time* más restrictivo tiene lugar cuando el último bit del frame del PCD durante la inicialización vale 1. En ese escenario, el FDT se calcularía atendiendo a la siguiente relación:

$$FDT = \frac{n \cdot 128 + 20}{f_c} \tag{3.2}$$

donde f_c es la frecuencia en hercios (Hz) y bajo lo dispuesto en la norma ISO/IEC 14443, $f_c = 13.56$ MHz resultando $\mathbf{FDT_{max} = 86.43\ \mu s}$ y $n \geq 9$ para tasas de transmisión de 106 kb/s[11] [36]

Por su parte, el *Frame Waiting Time* seguiría la siguiente relación:

$$FWT = \left(256 \cdot \frac{16}{f_c}\right) \cdot 2^{FWI} \tag{3.3}$$

donde FWI es el *Frame Waiting Time Integer* y hace referencia a un valor entero utilizado para definir el FWT tal que $0 \leq FWI \leq 14$. A pesar de que según [28] se contempla además el uso de un bloque específico —*Waiting Time eXtension* (WTX)— para solicitar un tiempo de adicional respuesta al PCD, la ventana FWT que nos permite la norma iría desde $\mathbf{FWT_{min} = 302\ \mu s}$ a $\mathbf{FWT_{max} = 4989\ ms}$ que se entiende como suficiente para cubrir los retardos sobre TCP/IP de ida y vuelta, aunque habrá que tener en consideración el retardo total en el que se incluyen los retardos de aplicación —cuantificables de forma experimental— de cada componente a nivel de aplicación.

[11] `https://nfc-forum.org/resources/what-are-the-data-transmission-rates`

3.3 Android

En la sección 2.3 del capítulo *Estado del Arte* se hacía referencia por primera vez al tercer actor de este estudio. En esa sección se ponía de manifiesto por qué la elección de este sistema operativo no era arbitraria sino que respondía a su puesta al día con los entornos actuales de medios de pago al implementar la tecnología HCE (*Host-based Card Emulation*) desde su versión 4.4 —ver **Tabla 2.1**— y por su posición marcadamente dominante en el parque móvil mundial. Decisiones como la de poner a disposición y mantener de forma ejemplar un SDK (*Software Development Kit*), APIs (*Application Programming Interface*) y entornos de desarrollo —ver sección **Herramientas**—a disposición de los usuarios han acabado por consolidar esa posición. Por todo lo anterior, el popular sistema operativo de Google para arquitecturas x64, x86, ARM[12] (32 y 64 bits), MIPS[13] y MIPS64 basado en el *kernel*[14] de GNU-Linux[15] será el sistema antifitrión del desarrollo *NFC Playground* que se documenta en la sección **NFC Playground**.

Por continuidad con la sección anterior, se abordará en primer lugar lo concerniente a la implementación de HCE en Android, y posteriormente dada la envergadura del desarrollo *NFC Playground* se incidirá en aquellos aspectos teóricos más relevantes para el desarrollo de sus funcionalidades como son el concepto de *multithreading* (programación multihilo) —consultar el problema clásico de **La Cena de los Filósofos**— y *fragments* en Android.

3.3.1 HCE (Host-based Card Emulation)

En primer lugar y con carácter previo a entrar en mayor detalle, cabe puntualizar que Android también concibe la comunicación sobre NFC fuera de HCE. Para ello se requiere de un *elemento seguro* —físico tipo SIM [37]— como se ilustra en la **Figura 3.8**. Fue el modelo original de comunicación sobre NFC implementado en Android y el adoptado por algunos *wallets* propietarios como el de Vodafone. Sin embargo, como se verá a continuación, la posibilidad de trabajar sobre NFC en modo HCE lo ha relegado en favor de este último.

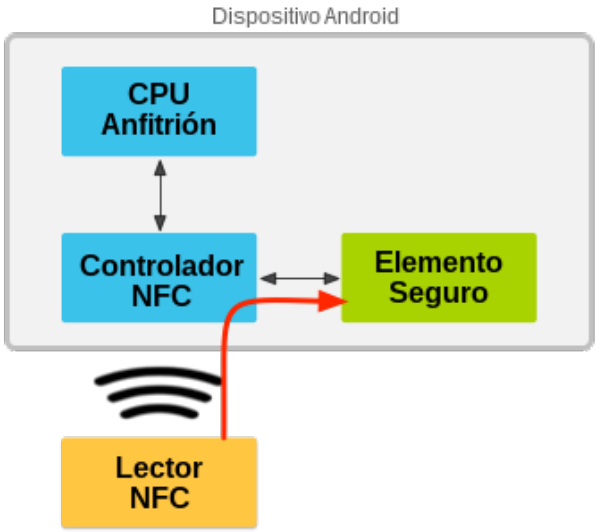


Figura 3.8: Modo de operación sobre NFC basado en *elemento seguro*

[12] https://www.arm.com
[13] https://www.mips.com
[14] https://www.kernel.org
[15] https://www.gnu.org/gnu/linux-and-gnu.es.html

El modo de operación que ocupa a este estudio y por extensión el modelo por el se está decantando el sector gracias a su flexibilidad —no todos los *wallets* pueden ofrecer o integrar un elemento seguro bajo su control en el dispositivo del usuario— es el que llega de la mano de ***KitKat*** —version 4.4 del sistema operativo Android—, es decir, el modo de operación en el que se prescinde de elemento seguro de forma que todo la capacidad de interlocución durante el proceso de emulación recae sobre la CPU[16] del dispositivo según puede apreciarse en la **Figura 3.9**. La pila de protocolos HCE se muestra en la **Figura 3.10**: A pesar de que esta implementación

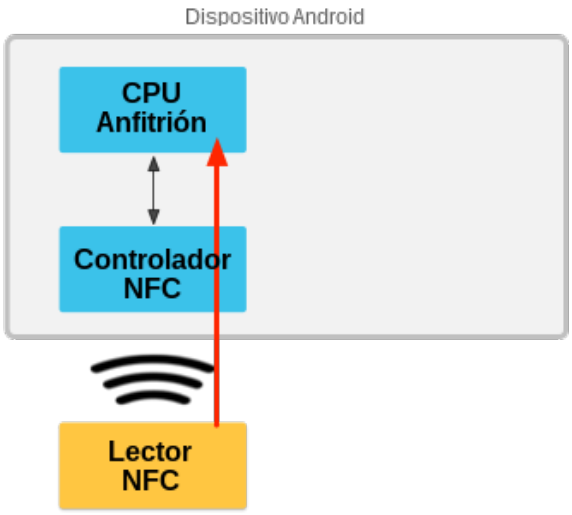


Figura 3.9: Modo de operación sobre NFC basado en HCE

Figura 3.10: Detalle de la pila de protocolos en HCE

de NFC en ***KitKat*** y posteriores soporta desarrollos sobre aplicaciones *NO-NDEF* [34], una vez más por razones de concisión y sobre todo por la propia naturaleza del entorno NFC-EMV®, el presente estudio se centrará en el tratamiento de información *NDEF* dentro del universo Android.

Para entender el funcionamiento de HCE es necesario entender cómo gestiona Android el flujo de información asociada a aplicaciones *NDEF*, razón por la que resulta fundamental conocer los principios de funcionamiento del *subsistema de envío de tags* o `tag dispatch system`[17]. Este subsistema funciona como un demonio[18] que se encuentra permanentemente[19] sondeando *tags* NFC en su radio de comunicación de acuerdo al protocolo de inicialización visto en la sección anterior —ver **Figura 3.11**— hasta descubrir algún *tag*. Debido al corto rango de acción de la comunicación sobre NFC, el subsistema intentará además entregar de forma automática, es decir, sin interacción por parte del usuario, la información correspondiente a la actividad responsable de gestionar este tipo de datos. Lo hará así con la idea de evitar que el movimiento del terminal asociado a la interacción del usuario -aun siendo de poca magnitud- provoque la ruptura del canal de comunicación sobre NFC. En caso de que ninguna actividad responda a la llamada, Android sí solicitará interacción del usuario para que en última instancia tome una decisión.

[16] *Central Processing Unit*
[17] `https://developer.android.com/guide/topics/connectivity/nfc/nfc#tag-dispatch`
[18] `http://www.enderunix.org/docs/eng/daemon.php`
[19] Siempre que la funcionalidad NFC esté activada y la interfaz no esté bloqueada.

Para ello, el `tag dispatch system` analizará el *payload*[20] del *tag* intentando obtener el *URI* [21]o el tipo *MIME* [22] que contiene, y lo encapsulará de acuerdo con [40] en un *intent*[23] con el objeto último de iniciar una actividad a la que ceder dicho *intent*.

Existen 3 tipos de *intent* en función de su nivel de prioridad que se detallan en la siguiente tabla ordenados de mayor a menor prioridad:

Intent	Contexto
`ACTION_NDEF_DISCOVERED`	*Tag* escaneado y *payload* reconocido.
`ACTION_TECH_DISCOVERED`	Anterior no atendido o *tag NO-NDEF* o *NDEF* no mapeable.
`ACTION_TAG_DISCOVERED`	Ninguno de los anteriores atendido.

Tabla 3.12: *Intents* relativos a *tags* NFC en Android

Con objeto de reducir el nivel de abstracción, el ciclo de vida de un *intent* cualquiera de los anteriores podría resumirse visualmente en el siguiente diagrama:

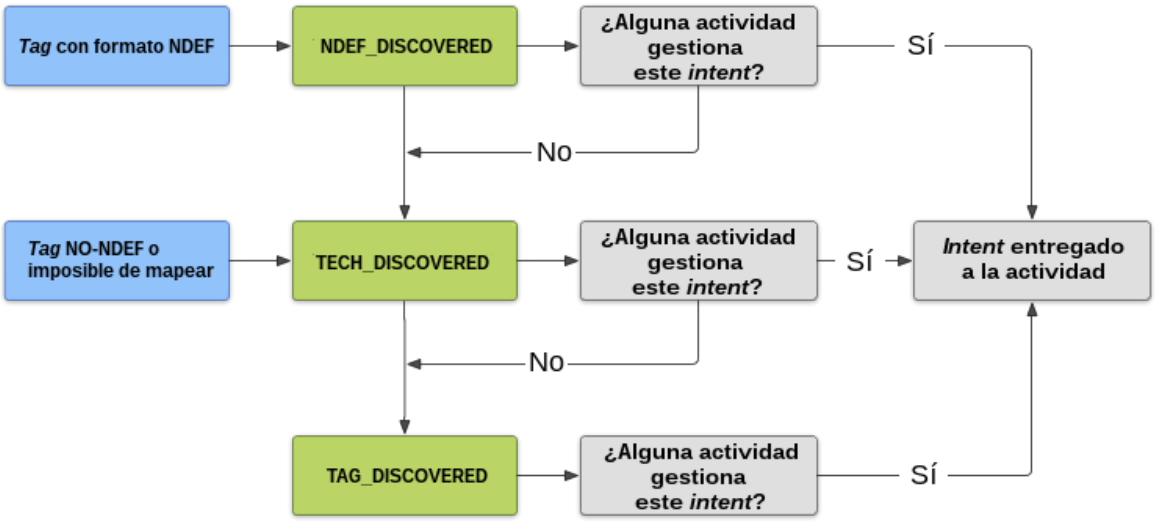


Figura 3.11: Ciclo de vida de un *intent* asociado a un *tag* NFC

Para entender la gestión de la información que viaja en los *intents* —y sus objetos de la clase `Tag` asociados— a nivel de aplicación es esencial la mención a la clase **IsoDep**[24]. Esta clase recibe el nombre del modo de operación acorde a la norma ISO/IEC 14443-4 [33] y facilita las operaciones de entrada/salida (*I/O*) y el acceso a las propiedades del *tag*.

[20]Carga útil que supone la enjundia del mensaje.
[21]*Uniform Resource Identifier* [38]
[22]*Multipurpose Internet Mail Extensions* [39]
[23]Un *intent* es una abstracción del modelo de comunicación entre procesos asociada a una tarea a realizar. Es por ello un componente fundamental en la relación entre actividades en el ecosistema Android. [41]
[24]`https://developer.android.com/reference/android/nfc/tech/IsoDep`

De todos los métodos de la clase, 6 resultan de especial relevancia para cualquier desarrollo que pretenda implementar HCE:

```
public static IsoDep get(Tag tag)
```

Cuando, siguiendo el procedimiento descrito, el `tag dispatch system` encapsule y envíe la información del *tag* a nuestra aplicación, ésta deberá en primer lugar obtener una instancia de la clase IsoDep para poder interactuar con el dispositivo. Esto se conseguirá a través de este método y el objeto de la clase `Tag` recibido del `tag dispatch system`.

```
public void connect()
```

A través de la llamada a este método sobre la instancia de la clase IsoDep obtenida se establecerá el canal lógico de comunicación entre nuestra aplicación y el dispositivo. Esta llamada **puede ser bloqueante**.

```
public byte[] getHiLayerResponse()
```

Método diseñado específicamente para la comunicación con dispositivos NFC que implementen tecnología ISO/IEC 14443 A.

```
public byte[] getHistoricalBytes()
```

De forma análoga al anterior, este es el método a emplear para la comunicación con dispositivos NFC que implementen tecnología ISO/IEC 14443 B.

```
public byte[] transceive(byte[] data)
```

A partir del momento en que ha quedado establecido el canal lógico de comunicación, este método es con diferencia la punta de lanza de la interacción durante el proceso de identificación con el interlocutor de nuestro dispositivo Android, pues su implementación permitirá definir nuestra propia pila del protocolo de aplicación que intercambiará información IsoDep en crudo. Por su propia naturaleza de operar a nivel de I/O **no solo puede sino que bloqueará** al proceso hasta su finalización.

```
public abstract byte[] processCommandApdu(byte[] commandApdu, Bundle extras)
```

Si `transceive` era la piedra angular para la comunicación con el *tag* durante la identificación, `processCommandApdu` es su homólogo en la fase de emulación. Este método permitirá responder al diálogo iniciado por el PCD según la información recabada durante el proceso de identificación hasta completar la sesión.

3.3.2 Multithreading

Con objeto de que la App no sea una mera aplicación para probar el escenario, se ha desarrollado además una serie de funcionalidades que mejoran la QoE[25] (*Quality Of Experience*) que se documentarán en la **Sección NFC Playground**. No obstante, para poder abordar estos desarrollos se requieren ciertos conocimientos adicionales tanto del paradigma de la OOP[26] (*Object Oriented Programming*) como de desarrollos en entornos que además implican UI[27] (*User Interface*) como es el caso de Android.

Una App puede estar formada de una o varias actividades. Sin embargo, el ciclo de vida es común a todas y resulta fundamental para el correcto funcionamiento de la aplicación que el ciclo de vida de una actividad guarde una relación lo más armónica posible con el del resto de actividades así como con su propio cometido. Para entenderlo de forma más gráfica se presenta un diagrama con el ciclo de vida de una actividad en Android:

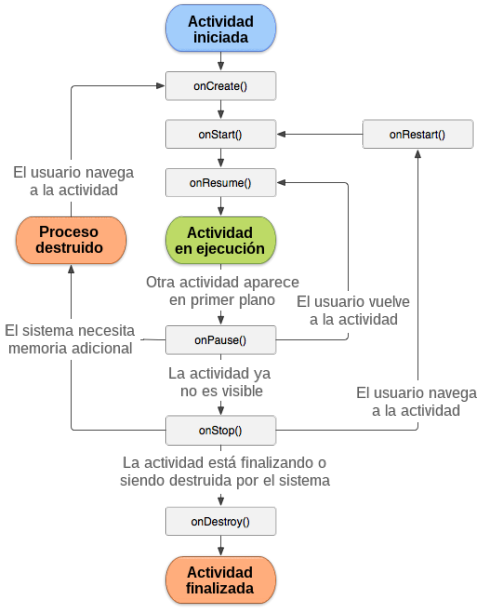


Figura 3.12: Ciclo de vida de una actividad en Android

El propio paradigma de programación en Android requiere no interferir —bloquear— al proceso principal, esto es, el que ve y sobre el que interactúa directamente el propio usuario. Por tanto, para poder implementar HCE en cualquier desarrollo *debe existir una rigurosa gestión de los procesos de las actividades y sus respectivos hilos* —subprocesos— toda vez que en muchos casos estos procesos, como se ha visto anteriormente, son bloqueantes. Las soluciones implementadas en la App para garantizar que dichos procesos bloqueantes coexistan sin afectar a la *QoE* o al correcto funcionamiento de la App son varias pero podrían resumirse en dos escenarios en función de la envergadura del proceso a gestionar en el hilo:

1. **Procesos que por su calidad de bloqueantes o por comodidad en el acceso al hilo principal de la actividad no requieren mucha capacidad ni tiempo de procesamiento:**

   En este tipo de escenario se prefiere adoptar un enfoque más circunstancial a través de

[25] `https://www.itu.int/md/T05-FG.IPTV-IL-0050/es`
[26] Programación Orientada a Objetos
[27] Interfaz de Usuario

una interfaz de la clase `Runnable` y el método **`runOnUiThread`** para acceder eventualmente al hilo principal desde un hilo en segundo plano (*background*), realizar la acción que se requiera y volver a un segundo plano o finalizar.

2. **Procesos bloqueantes que soportan una carga de proceso mucho mayor:**

   Estos escenarios, por su complejidad, requieren mayor flexibilidad a la hora de gestionar su relación con el hilo principal. Técnicamente, la solución anterior no sería bastante flexible para permitir una relación suficientemente dinámica con el hilo principal. Por ello, la forma más eficiente de realizar esta gestión es a través de la clase **`AsyncTask`** [42].

Las posibilidades en la relación de las instancias de la clase anterior con el hilo principal atenderían a la siguiente figura:

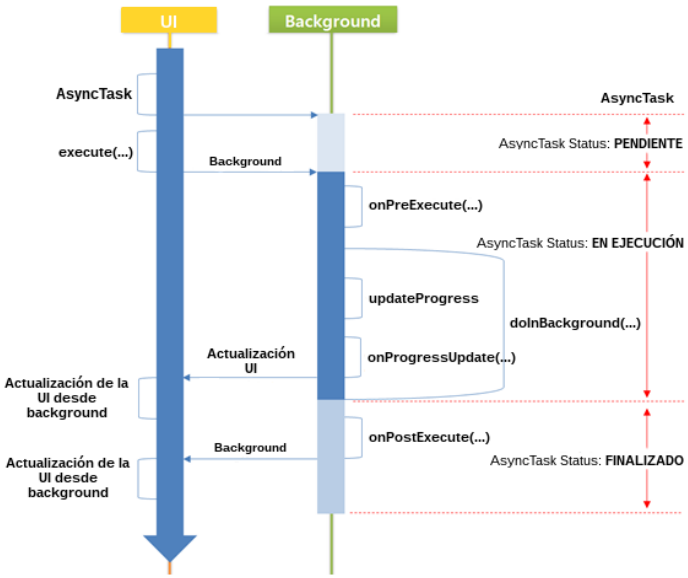


Figura 3.13: Relación de la clase Asynctask con el hilo UI

Este modelo, basado en un *pool* de 128 hilos *reutilizables*, optimiza la comunicación con el hilo de UI —o con cualquier otro hilo—, lo que redundará en una mayor flexibilidad para el desarrollador, mayor versatilidad en la aplicación y mejor experiencia para el usuario.

3.3.3 Fragments

Para poder presentar la información de una forma más clara, ordenada y que además permita al usuario realizar el mayor número de acciones con el menor esfuerzo, el desarrollo realizado en el marco de este estudio implementa también un tipo de actividad especial llamada `fragment`[28]. Este tipo de actividad debe ser vista como una modularización del concepto clásico de actividad[29]. Su ciclo de vida, propio y distinto del de una actividad al uso, se desarrolla al amparo de una actividad principal que gestiona su comportamiento, la comunicación entre fragmentos o incluso, como es también el caso del desarrollo de este trabajo, entre actividades y fragmentos.

[28] `https://developer.android.com/guide/components/fragments?hl=es-419`
[29] `https://developer.android.com/reference/android/app/Activity`

Capítulo 4

Metodología

4.1 Material utilizado

4.1.1 Tarjetas EMV® de certificación

Son tarjetas con soporte EMV® emitidas por ***American Express Co.*** —en adelante **AMEX**— y ***VISA Inc.*** —en adelante **VISA**— con el fin de ser empleadas para realizar certificaciones de nuevas versiones de *software* en circuitos de prueba con carácter previo a su salida a producción. Algunas de ellas disponen de soporte *contactless* y otras no.

Figura 4.1: Detalle del anverso de una de las tarjetas de certificación VISA utilizadas. Sin soporte *contactless*.

Figura 4.2: Detalle del reverso de una de las tarjetas de certificación VISA utilizadas. Sin soporte *contactless*.

Figura 4.3: Detalle del anverso de una de las tarjetas de certificación AMEX utilizadas. Con soporte *contactless*.

Figura 4.4: Detalle del reverso de una de las tarjetas de certificación AMEX utilizadas. Con soporte *contactless*.

Figura 4.5: Detalle del anverso de una de las tarjetas de certificación AMEX utilizadas. Sin soporte *contactless*.

Figura 4.6: Detalle del reverso de una de las tarjetas de certificación AMEX utilizadas. Sin soporte *contactless*.

4.1.2 PIN Pad Ingenico IPP 320

PIN Pad[1] con soporte *contactless* y EMV® de gran implantación en el sector desarrollado por *Ingenico*[2] . Pese a que ya se está reemplazando por versiones de última generación todavía suponen un pequeño porcentaje frente al **IPP 320** empleado en este estudio, que puede encontrarse en la mayoría de comercios.
El *firmware*[3] con el que trabaja el **IPP 320** empleado en este estudio—PUP 1.6V1.60-0010— es una de las versiones que se encuentra trabajando en producción en uno de los grandes clientes del sector. Cabe señalar adicionalmente que el juego de claves públicas y privadas del dispositivo es un juego de claves de certificación no apto para entornos de pago reales. Por ello, está preparado únicamente para evaluar la fase de autenticación de tarjeta, que por otra parte es un marco válido para este estudio.

Figura 4.7: PIN Pad usado en el estudio

[1] https://www.tefpay.com/tefpay/preguntas-frecuentes-faq.php#5
[2] https://ingenico.us/smart-terminals/telium2/payment-terminals/ipp-series/ipp-320.html
[3] Software de bajo nivel para interactuar con el dispositivo

4.1.3 Smart Card Reader LTC31

Dispositivo con capacidad de lectura de tarjetas inteligentes (*Smart Card*) de contactos y comunicación **USB** creado por el fabricante desaparecido C3PO S.L.[4].

Figura 4.8: Detalle de la planta del lector LTC31

Figura 4.9: Detalle de la parte inferior del lector LTC31

[4] https://www.einforma.com/informacion-empresa/c3po

4.1.4 Samsung Galaxy J5

El móvil en el que se instalará y ejecutará la App —ir a **NFC Playground**—. Es un teléfono inteligente fabricado por *Samsung* en 2016, modelo *SM-J510FN*[5]. Dispone de soporte NFC y su sistema operativo es Android *MarshMallow* (6.0.1) —ver **Tabla 2.1**—

Figura 4.10: Móvil utilizado en el estudio

Figura 4.11: En azul, ubicación de la antena NFC

[5] https://www.samsung.com/es/support/model/SM-J510FZDNAMO

4.2 Herramientas

4.2.1 Android Studio

Es un entorno de desarrollo integrado (IDE) basado en **IntelliJ IDEA**[6]. Se utiliza para la programación de aplicaciones de sistemas operativos Android. En este estudio se ha empleado para la creación de la App NFC Playground —ir a **NFC Playground**—

Versión empleada[7]:
Android Studio 3.0.1
Build #AI-171.4443003, built on November 9, 2017
JRE: 1.8.0_152-release-915-b01 amd64
JVM: OpenJDK 64-Bit Server VM by JetBrains s.r.o

4.2.2 Eclipse

IDE para desarrollos principalmente basados en Java[8], con *Apache Maven*[9] como software de gestión de proyectos. En este estudio se ha empleado para la creación del componente NFC Playground Remote Driver —ir a **NFC Playground Remote Driver**—

Versión empleada[10]:
eclipse Oxygen.3 Release (4.7.3)
Build: 20180308-1800
Apache Maven 3.5.3

4.2.3 Sublime Text

Editor de texto con funcionalidades avanzadas. Empleado en este trabajo para la creación del componente NFC Playground Remote Driver —ir a **NFC Playground PIN Pad Demo Component**—

Versión empleada[11]:
Sublime Text 3.0
Build:3170

[6] https://www.jetbrains.com/idea
[7] https://developer.android.com/studio/releases/
[8] https://www.java.com/es/about/
[9] https://maven.apache.org
[10] http://www.eclipse.org/downloads/
[11] https://www.sublimetext.com/3

4.3 Planteamiento

Por convenio, en la composición del **Diagrama del escenario inicial** se definirá como ubicación *local* a la ubicación en la que coincidan el **PIN Pad Ingenico IPP 320** y el móvil **Samsung Galaxy J5** con la App **NFC Playground** instalada y en ejecución. Por el contrario, se entenderá como ubicación *remota* aquel sistema en una ubicación geográfica diferente al anterior en el que se encuentre instalado y en ejecución el componente **NFC Playground Remote Driver** y que además tenga conexión con el **Smart Card Reader LTC31**.

Desplegado el entorno anterior, se establacerá la correspondiente conexión entre los componentes y se implementará una batería de pruebas en función del tipo de tarjeta utilizada en el estudio —ver **Tarjetas EMV® de certificación**— en la que la App NFC Playground intentará establecer una sesión EMV® hasta la fase de autenticación de tarjeta según lo visto en la sección **ISO/IEC 7816**. Solo en ese momento se podrá evaluar el impacto, por ejemplo en el retardo, de todas aquellas circunstancias por cuyo volumen de variables solo es posible cuantificar de forma experimental.

Con base en los primeros resultados, y siempre que sea posible, se tomará las consideraciones oportunas a nivel de programación con la finalidad de alcanzar el objetivo establecido. Una vez finalizada la batería de pruebas y con los resultados definitivos se estudiará las conclusiones y las líneas futuras de trabajo, si procede.

Capítulo 5

NFC Playground Suite

5.1 NFC Playground

5.1.1 Diseño

Debido a que la implementación de HCE en Android no está diseñada para que la información realice el circuito del **Diagrama del escenario inicial** ha sido necesario crear desde la base un andamiaje lógico que soporte ese flujo. Por ello, la comunicación entre actividades, servicios e hilos asíncronos atiende al patrón de diseño *Observer* [43] —clave de la arquitectura *Modelo Vista Controlador* (MVC) [44]— según el cual un objeto, ante un cambio en el estado de otro objeto, notifica este cambio al resto de objetos con los que mantiene una relación de dependencia.

Dentro de las consideraciones iniciales de diseño, la primera pasa por registrar en el momento de desarrollo la solicitud del uso de recursos como el acceso al estado del teléfono, salida a Internet o los permisos lectura y escritura asociados al **SAF**[1] (**Storage Access Framework**) pero especialmente los asociados a NFC. Del mismo modo es requisito imprescindible el registro de los **AIDs** de las aplicaciones con las que trabajará la App.

NFC Playground se desglosa en las siguientes actividades:

- ***Presentación***: Primera toma de contacto con la UI de la App.
- ***Menú***: Carrusel interactivo para la selección de actividades.
- ***Identificación***: Toma de datos de la PICC.
- ***Emulación***: Emulación de la PICC identificada.
- ***Ubicuidad***: *PoC* del escenario objeto del presente estudio.
- ***Sesiones***: Registro de las sesiones generadas.
- ***Ajustes***: Preferencias de la App.
- ***Acerca de***: Breve introducción a la App.

[1] `https://developer.android.com/guide/topics/providers/document-provider?hl=es-419`

Presentación

Lo primero que verá un usuario al arrancar la aplicación es una pantalla de presentación:

Figura 5.1: Pantalla de inicio de la aplicación

Realizada la presentación, el usuario accederá automáticamente a una pantalla en la que se mostrará un menú animado en 3D desde el que podrá navegar a cada una de las 6 actividades de la App según se representa en la siguiente figura:

Figura 5.2: Aspecto inicial del Menú Principal

Figura 5.3: Detalle del Menú Principal desde otra perspectiva

Identificación

Por orden de aparición, la primera actividad accesible al usuario es la actividad *Identificación*. Esta actividad, que implementa HCE, tiene como misión en primer lugar evaluar que la configuración de NFC es la correcta, esto es, si el dispositivo soporta NFC y los servicios asociados a NFC están operativos, y en caso contrario interactuar con el usuario para corregir el problema. Esta actividad gestiona además el comportamiento y la comunicación de dos `fragments`, *FragmentoLectura* y *FragmentoMonitor*. Pese a que su fin último es el mismo, es decir, habilitar la disposición del terminal en un modo en el que pueda atender al `tag dispatch system` e interactuar con el *tag* —en este caso una **PICC** con soporte EMV® de las vistas en **Tarjetas EMV® de certificación**— y generar una sesión en la que obtenga información asociada a la tarjeta —variable en función del emisor y del operador bancario tal y como se trató en la sección **ISO/IEC 7816**—, tanto su manera de trabajar como el resultado final ofrecido al usuario son sensiblemente diferentes.
Al entrar en la actividad, por defecto se habilita el *FragmentoLectura*. En la siguiente figura se presenta el aspecto de la UI de esta actividad:

Figura 5.4: Imagen de la UI de la actividad con *FragmentoLectura* activo

En función del resultado —o del problema—, la actividad ofrecerá al usuario la posibilidad de pasar al modo de emulación o una relación de alternativas en caso de alguna contingencia en sentido contrario.

Como puede comprobarse en la figura anterior, esta actividad incorpora en la parte inferior derecha un menú animado —ir a **Manual de Usuario**— desde el que es posible acceder a la configuración NFC del dispositivo y a un modo especial de identificación desplegado por *FragmentoMonitor*. Este modo de identificación permite, además de poder establecer la sesión NFC/EMV® tal y como lo hacía *FragmentoLectura*, ofrecer una interpretación comprensible de la conversación completa entre el móvil y la PICC, acorde a la norma ISO/IEC 7816-4. Además, **siempre que el usuario así lo defina en sus preferencias** —ver **Manual de Usuario**— se guardará a través del SAF una versión interpretada completa de la conversación. En la siguiente imagen puede apreciarse el aspecto de esta UI:

Figura 5.5: Detalle del Terminal de *FragmentoMonitor*

Emulación

La siguiente actividad tanto por posición en el menú principal como por lógica de funcionamiento es *Emulación*. Esta actividad, además de los controles anteriores sobre el estado del dispositivo, tiene como misión la generación de una representación visual de la PICC identificada, la disposición del terminal en modo pasivo —que como se discutió en **ISO/IEC 14443** es el necesario para actuar como una PICC (tarjeta *contacless*) ante un PCD (*PIN Pad*)— y eventualmente, la conmutación entre el modo de identificación de *tags* y el de emulación de PICCs. La siguiente figura representa el aspecto del estado inicial de la UI de esta actividad:

Figura 5.6: Aspecto inicial de la UI de la actividad *Emulación*

Ubicuidad

Una vez implementados los dos escenarios —toma de datos y la emulación con arreglo a ellos— en las dos primeras actividades que cubrirían la principal funcionalidad de un *wallet* comercial, la siguiente actividad —*Ubicuidad*— pretende implementar la funcionalidad descrita en el **Diagrama del escenario inicial** en el que la App establece comunicación con el componente remoto NFC Playground Remote Driver sobre TCP/IP por una parte y con el PCD sobre NFC por otra, para intentar gestionar la sesión EMV® objeto de este estudio.

Figura 5.7: Detalle del inicio de la actividad *Emulación*

Figura 5.8: Detalle del panel de eventos durante la conexión

La actividad *Sesiones* permite, si así lo ha configurado el usuario, acceder a los registros de la sesión, visualizarlos, exportarlos y eliminarlos.

Figura 5.9: UI de la actividad *Sesiones* tras 5 sesiones EMV®

La siguiente actividad, *Ajustes*, permite configurar las preferencias relativas a la seguridad, las sesiones o su persistencia.

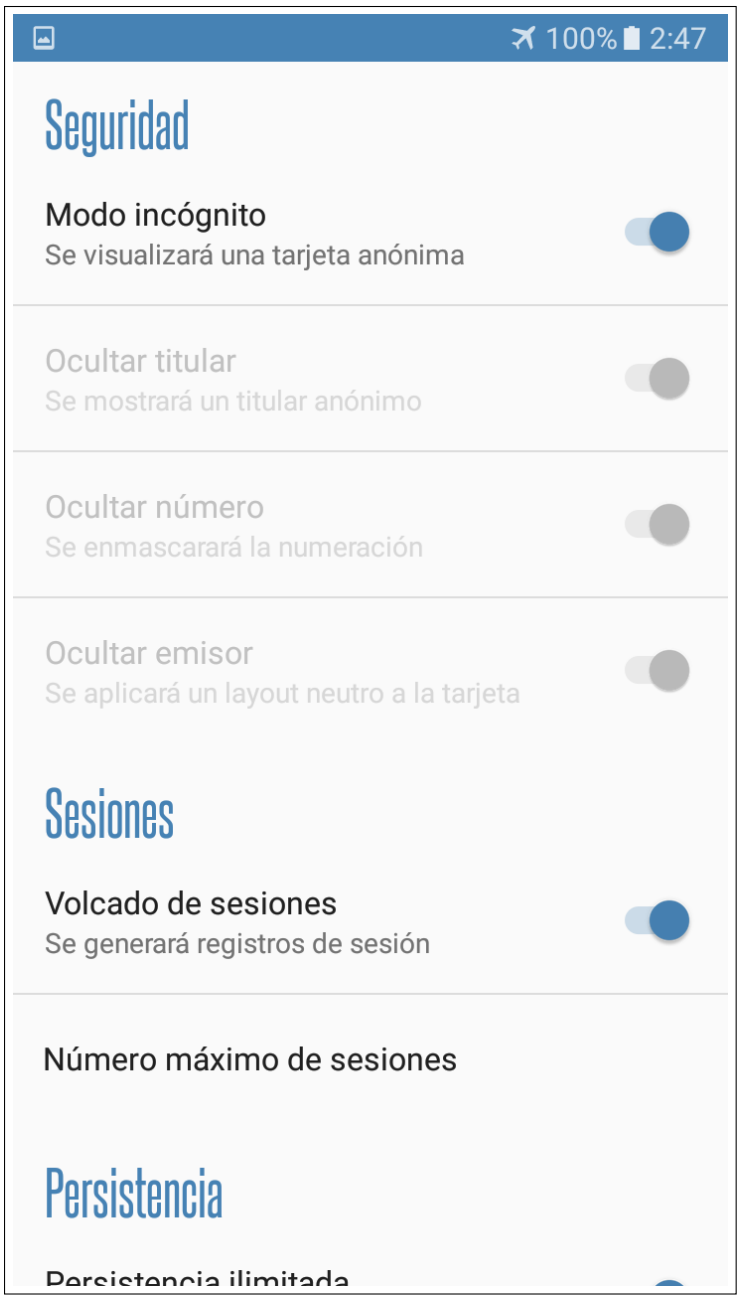


Figura 5.10: UI de la actividad *Ajustes*

Acerca de es la última actividad que implementa la actual versión de *NFC Playground*. En ella se presenta un pequeño desplegable que introduce al usuario la App, resumiendo en unas pocas líneas los detalles más singulares tanto de la aplicación como de la experiencia asociada a su desarrollo.

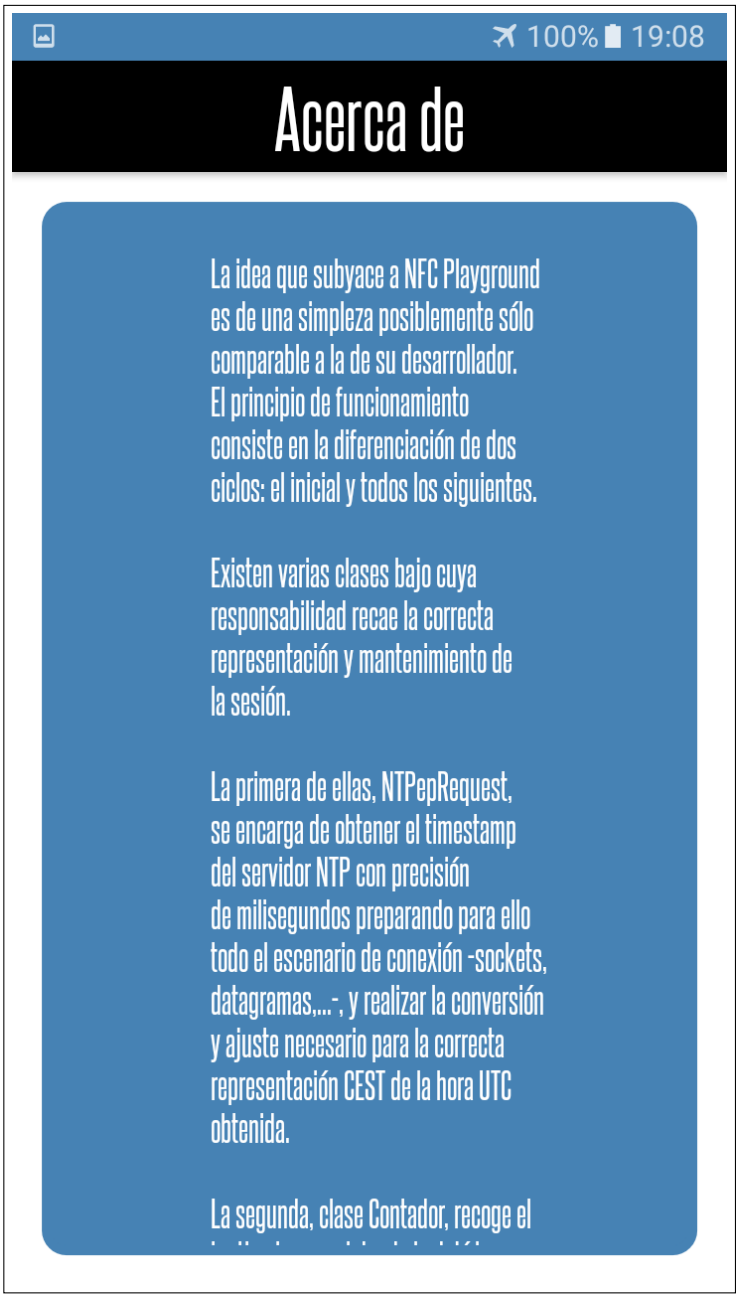


Figura 5.11: Imagen de la interfaz de la actividad *Acerca de*

5.1.2 Arquitectura

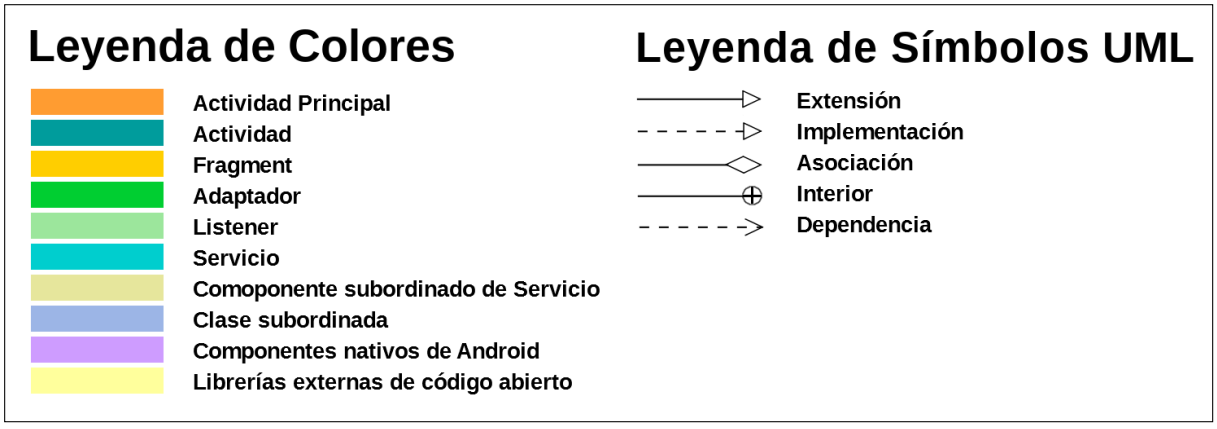
La arquitectura de NFC Playground responde al modelo de *separación de intereses* [45]. Se intenta que cada una de las piezas de código tenga un número limitado de responsabilidades con el fin de que cada una de ellas mantenga la máxima autonomía. Este planteamiento no es casual, toda vez que cabe recordar que en desarrollos Android los procesos derivados de la instanciación de las clases *no* son propiedad de la App, sino que representan la materialización del contrato entre la App y el substrato del sistema operativo. Por ello, en cualquier momento —y sin atender a ningún tipo de diplomacia informática— el sistema puede destruirlos en base a interacciones del usuario o a situaciones sobrevenidas como la falta de memoria o similares. Este enfoque tiene su máximo exponente en aquellas tareas que tienen un componente visual por ser la capa del programa más inmediata al usuario.

Como se describe en la sección **Diseño**, la App está dividida en 7 actividades. Aunque tiene una orientación muy marcada en la línea de la investigación, esto no está reñido con una interfaz más actractiva y cómoda para el usuario final por lo que como aportación adicional incorpora varias de las funcionalidades de un *wallet* estándar enmarcado en el segundo de los objetivos del presente trabajo, que es el de la profundización en el desarrollo de aplicaciones para el universo Android.

Por comodidad en la perspectiva del presente desarrollo en la siguiente página se incluye un diagrama de clases que representa los principales componentes de la aplicación y la relación que existe entre ellos descrita según el lenguaje unificado de modelado UML[2] (*Unified Modeling Language*) y la leyenda tanto para los componentes como para su relación.

[2] `http://www.uml.org`

Figura 5.12: Diagrama de Clases de la App *NFC Playground*



De los componentes del diagrama anterior son especialmente relevantes las actividades ***Identificación***, ***Emulación*** y ***Ubicuidad*** a los efectos de este estudio. El resto de actividades enmarcadas en el contexto de mejorar la QoE se omitirán por razones una vez más de concisión.

Aunque la mejor manera de entender una aplicación es leyendo su código fuente, por relajar la complejidad en la representación de cada una de las partes y su relación, se presenta a continuación sendos diagramas de secuencia para los principales escenarios de actuación.

En primer lugar, el diagrama de secuencia de la actividad ***Identificación*** con el `fragment` *FragmentoLectura* activo:

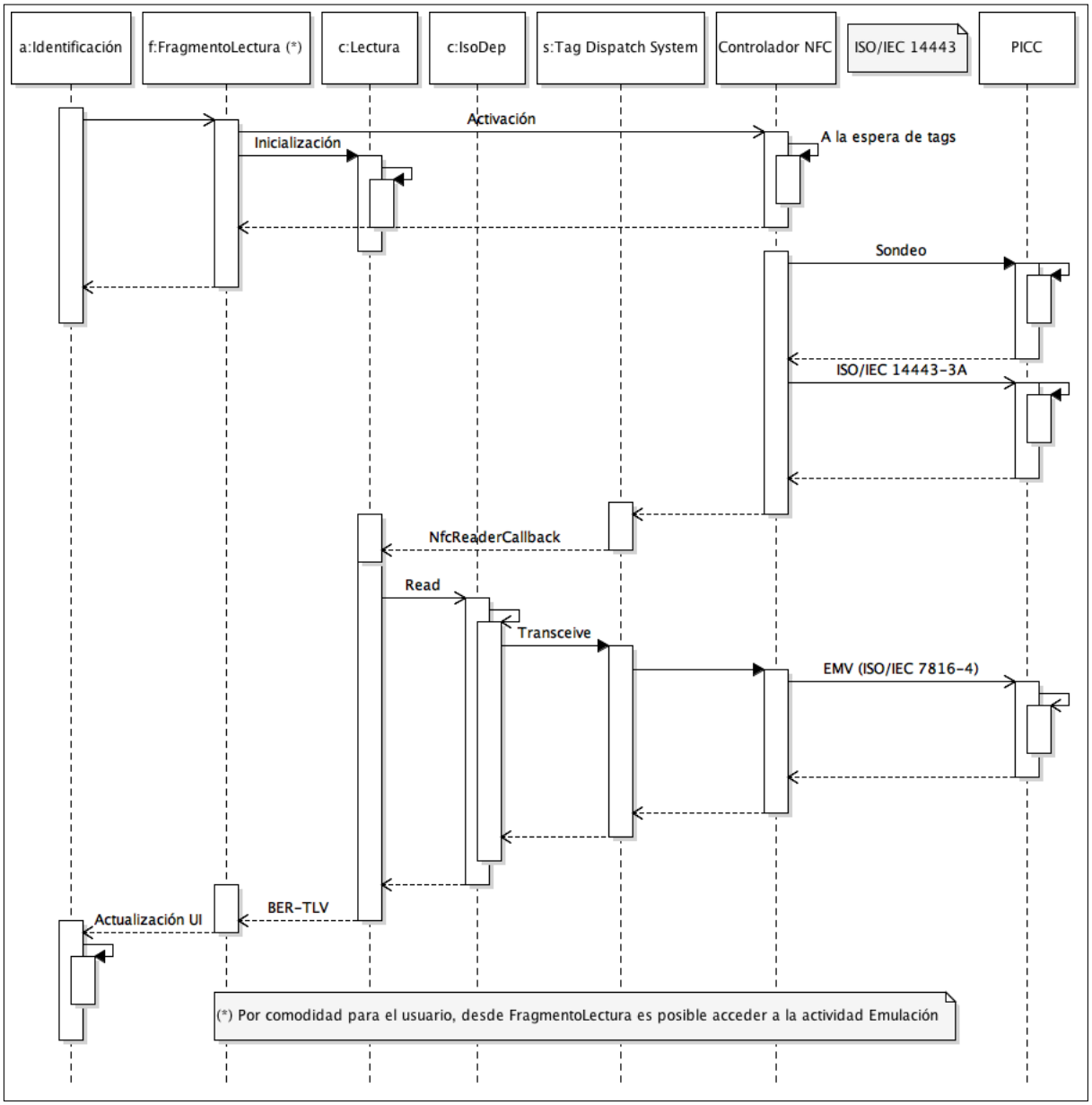


Figura 5.13: Diagrama de secuencia de la actividad *Identificación* desde el `fragment` *FragmentoLectura*

A continuación el diagrama de secuencia de la actividad ***Identificación*** con el `fragment` *FragmentoMonitor* activo:

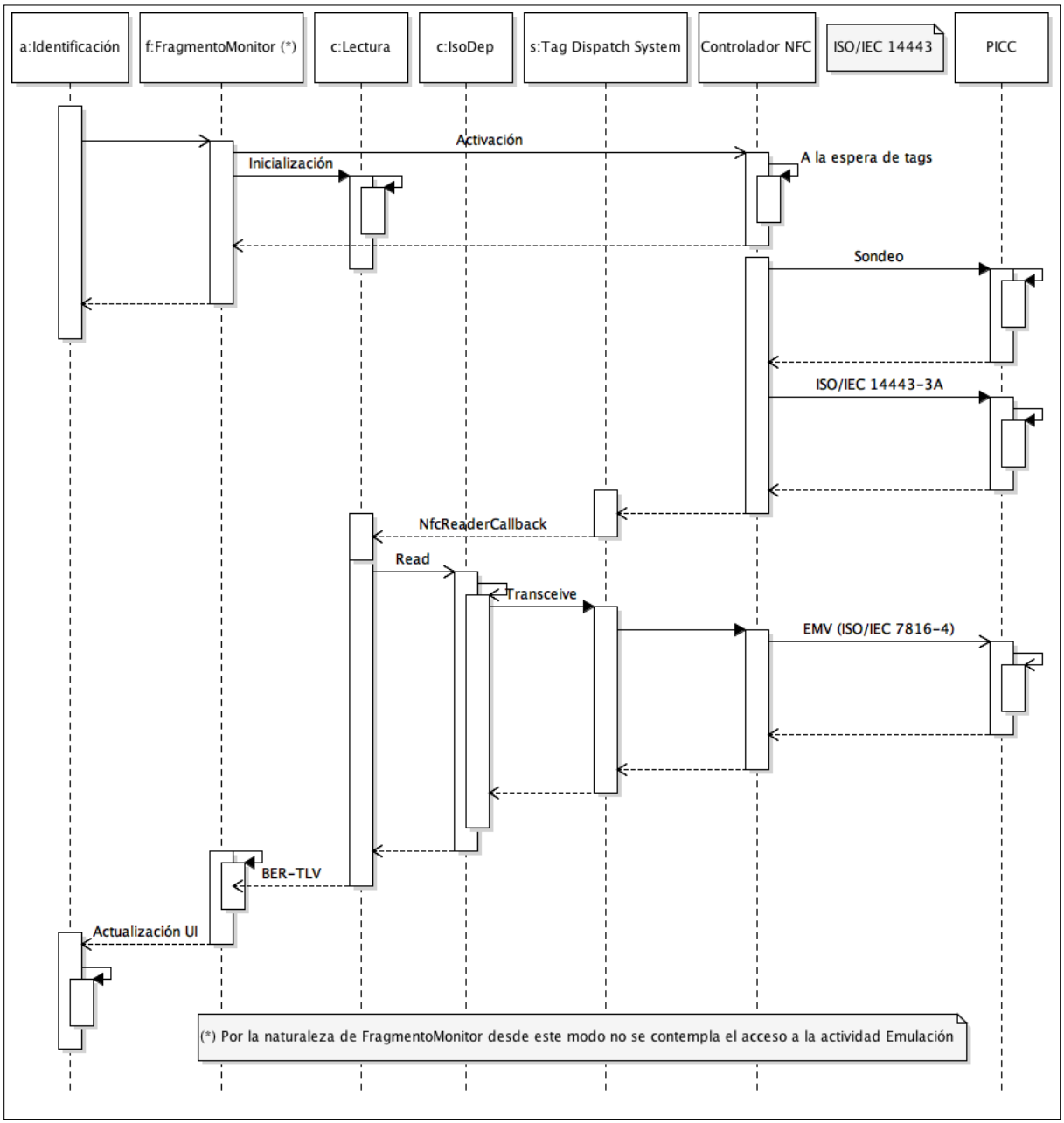


Figura 5.14: Diagrama de secuencia de la actividad *Identificación* desde el `fragment` *FragmentoMonitor*

Para mayor detalle acerca de la nota de la parte inferior de este diagrama y sucesivos, el lector puede remitirse a la sección **Diseño**.

El siguiente diagrama de secuencia que se presenta es el correspondiente a la actividad ***Emulación***:

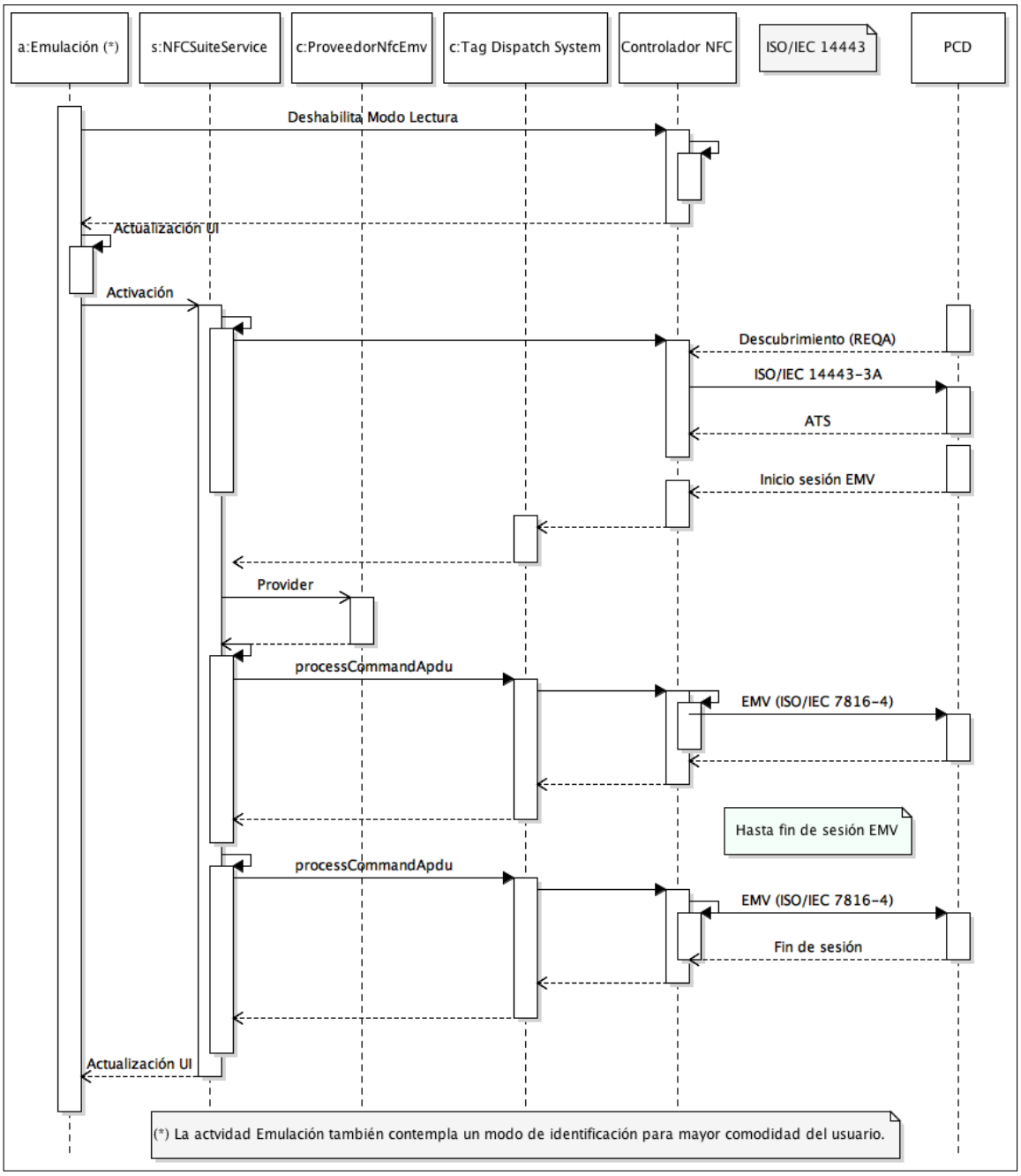


Figura 5.15: Diagrama de secuencia de la actividad *Emulación*

Por su parte, la actividad ***Ubicuidad*** podría representarse según la siguiente figura en la que como se indica en la anotación, se omite todas las fases de la comunicación entre el momento en que queda establecida la sesión EMV® y el momento inmediatamente anterior al de la finalización de la fase de autenticación de tarjeta:

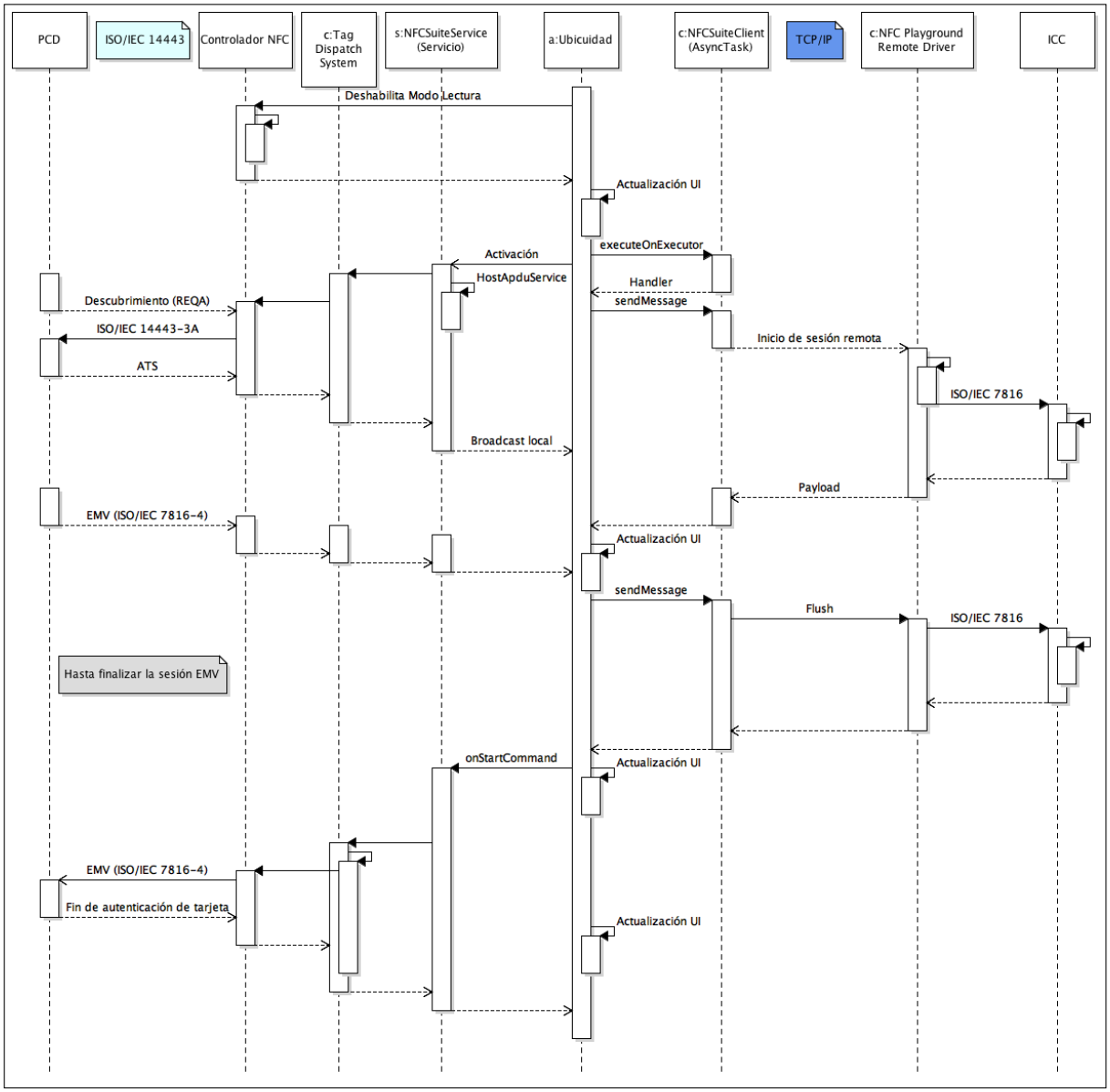


Figura 5.16: Diagrama de secuencia de la actividad *Ubicuidad*

5.1.3 Manual de Usuario

Puesto que algunas actividades como *Presentación*, *Menu* o *Acerca de* no revisten de mayor complejidad, en unos casos por no requerirse acción por parte del usuario y en otros por resultar su uso muy intuitivo, se detallará los aspectos más relevantes del resto de actividades a nivel de usuario en el mismo orden seguido en la sección **Diseño**. No obstante, cabe puntualizar que NFC Playground requiere ser configurado como el servicio predeterminado de transacciones de pago sobre NFC para poder operar con normalidad. Por ello, en su arranque solicitará ser configurado como el método predeterminado.

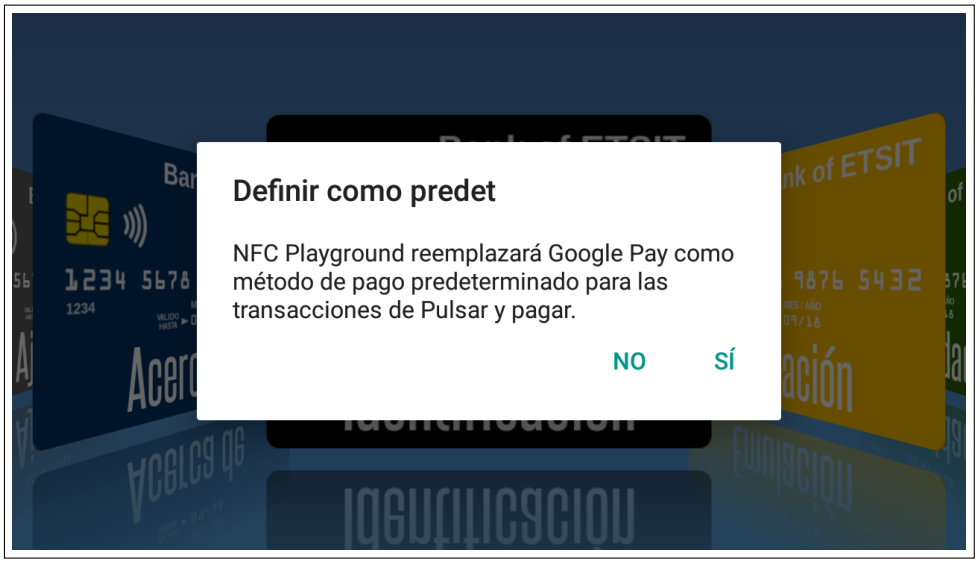


Figura 5.17: Solicitud de configuración del servicio NFC Playground como aplicación de pago por defecto

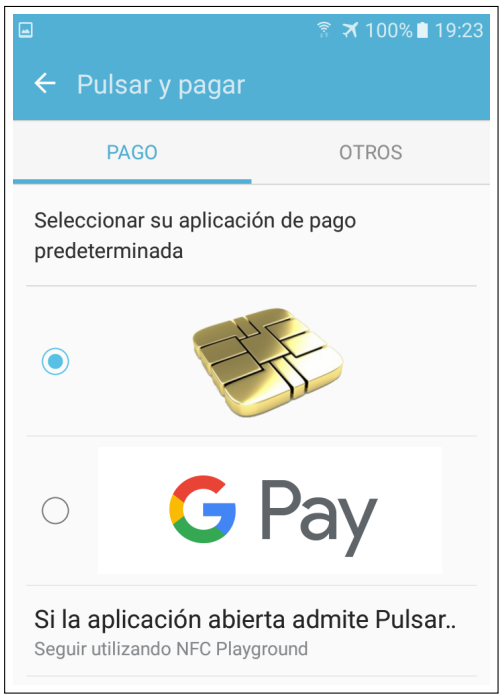


Figura 5.18: NFC Playground configurado como aplicación de pago por defecto

Identificación

En esta actividad en su modo por defecto, el usuario lo único que necesita es una PICC ya que la actividad le informará de cuantas acciones se requieran por su parte. Opcionalmente, éste tiene la posibilidad de cambiar al modo Monitor en el supuesto de que quiera estudiar la conversación entre el móvil y la PICC. Para ello, en la parte inferior derecha existe un icono que al contacto despliega un pequeño menú vertical animado —**Figura 5.19**— con un botón para acceder a las opciones de configuración NFC del terminal y otro para activar el modo Monitor. Este menú se repliega en tres casos: al elegir una opción, al pulsar el botón principal del menú, o al pulsar *Atrás* en el terminal.

Figura 5.19: Detalle del menú en su configuración por defecto

Desde el modo Monitor el menú funciona de forma ligeramente diferente —**Figura 5.20**—. En primer lugar y con motivo de la nueva disposición de la UI por el espacio reservado a la ventana de *NFC Playground Terminal*, el menú que antes era vertical pasará a desplegarse en horizontal. Además, al desplegarlo aparecen dos opciones diferentes. El icono de la pantalla vacía borra el contenido del terminal —útil cuando se encadenan varias sesiones—, mientras que por su parte el icono de la pantalla con una flecha apuntando hacia abajo esconderá el *NFC Playground Terminal* configurando la actividad nuevamente en su modo *Lectura* por defecto.

Figura 5.20: Detalle del menú en su configuración en modo Monitor

Emulación

La actividad *Emulación* resulta también bastante intuitiva a nivel de usuario. Para el caso base de que el usuario acceda a ella por primera vez, el terminal quedará dispuesto en modo de captura de datos a la espera de disponer de una tarjeta que emular. Así, de completarse satisfactoriamente el proceso de captación de datos se generará automáticamente una representación visual del modelo de tarjeta identificada personalizada con los datos recabados.
En caso de llegar a esta actividad de la mano de la actividad *Identificación* en su modo de operación por defecto, generará la vista de la tarjeta sin necesidad de volver a identificar la tarjeta quedando como muestra la **Figura 5.21**:

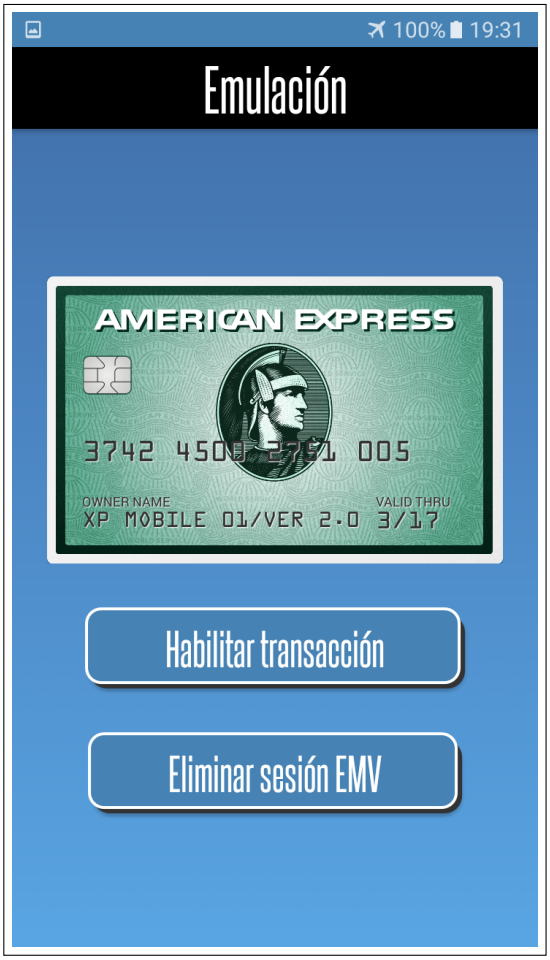


Figura 5.21: UI de la actividad *Emulación* tras identificar una PICC

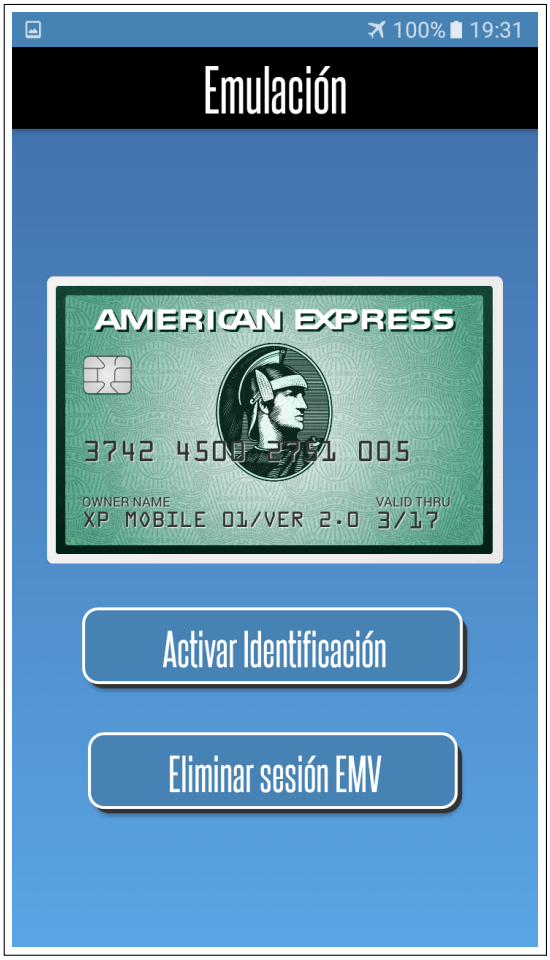


Figura 5.22: Detalle de los botones tras pulsar *Habilitar transacción*

Adicionalmente se materializarán dos botones. El primero de ellos —***Habilitar transacción***— solicita confirmación al usuario de que está conforme con el *tag* identificado —no quiere identificar y usar otro— de cara a disponer el terminal en modo de emulación de tarjeta para realizar una transacción en un *PIN Pad*.

En caso de pulsar ***Habilitar transacción***, este botón cambiará su mensaje a ***Activar Identificación*** permitiendo al usuario volver al modo de identificación según conveniencia. Este botón, por tanto permite al usuario conmutar fácilmente entre los dos roles en HCE. El otro botón, ***Eliminar sesión EMV*** elimina la información EMV® de la tarjeta obtenida durante la identificación.

Ubicuidad

La actividad *Ubicuidad* presenta dos botones y un panel de eventos. El botón —***Iniciar Servicio***— establece comunicación con el componente remoto **NFC Playground Remote Driver**. Su texto cambiará a ***Detener Servicio*** en caso de que la comunicación sea satisfactoria —para ofrecer al usuario la posibilidad de interrumpir la sesión a su discreción— al mismo tiempo que el panel de eventos reflejará los acontecimientos más relevantes de la sesión hasta el punto de iniciar una transacción en el *PIN Pad*

Figura 5.23: Detalle del registro del fallo de conexión a Internet

Figura 5.24: Detalle del registro de la desactivación del servicio de NFC

La cabecera del panel de eventos es táctil, de forma que además de informar de los cambios de estado y adecuar su color a los mismos, al pulsarlo comprobará el estado de las conexiones.

Ajustes

La actividad *Ajustes* está dividida en tres secciones: **Seguridad**, **Sesiones** y **Persistencia**. Permite al usuario configurar algunos parámetros del comportamiento de la aplicación en los tres ámbitos. Por defecto —**Figura 5.25**— está desactivada toda personalización para las vistas generadas de las tarjetas identificadas.

Cada opción es representada por un literal sobre el parámetro en cuestión, *e.g. Modo Incógnito* y debajo, dentro del mismo objeto, una descripción en gris del estado en función de la opción configurada en ese momento. Por ejemplo, en la **Figura Configuración por defecto de la App tras su primera instalación** al estar activada la opción —botón en azul— la lógica positiva sobre el *Modo Incógnito* obliga a visualizar una tarjeta anónima, de ahí que la descripción en este caso sea *Se visualizará la tarjeta anónima*. Al deshabilitar esta opción se habilita la posibilidad de configurar otras opciones asociadas al mismo ámbito, que pasarán de estar sombreadas a estar activas. El comportamiento es similar para el resto de los parámetros.

El diseño de esta funcionalidad a nivel de persistencia de los cambios garantiza que los ajustes definidos a partir del momento de la instalación seguirán vigentes mientras no se cambien o se desinstale la *App*.

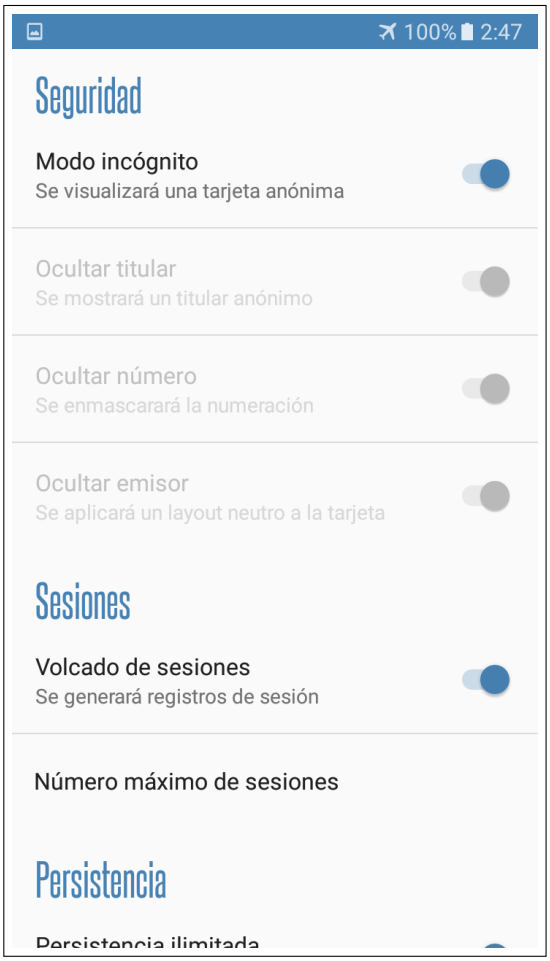


Figura 5.25: Configuración por defecto de la App tras su primera instalación

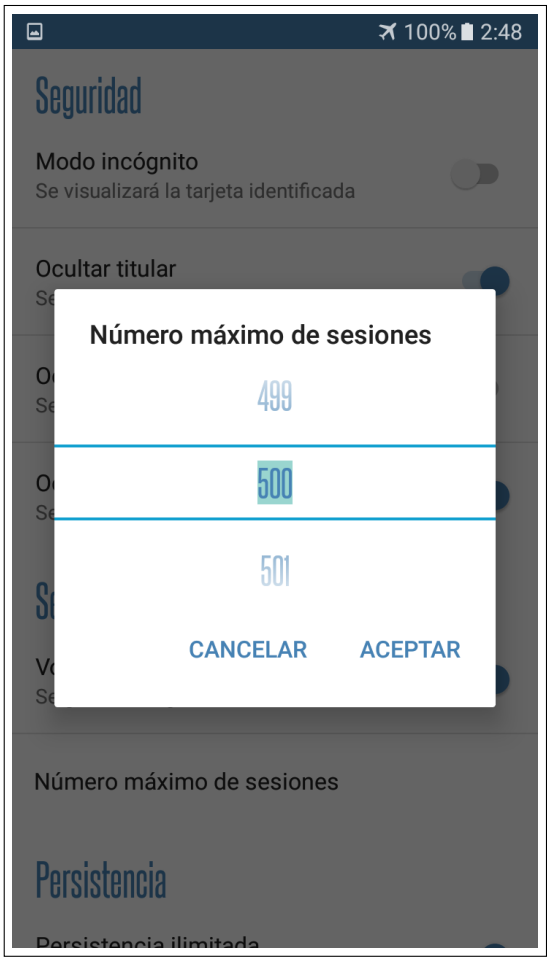


Figura 5.26: Detalle del menú del número de sesiones a mantener

5.2 NFC Playground Remote Driver

5.2.1 Diseño

Dos son los principios que rigen el diseño de este componente. En primer lugar, **ser transparente a nivel de la norma ISO/IEC 7816-4** de forma que toda la lógica subyazca en la *App* anterior, **NFC Playground**. En segundo lugar, su **sencillez**, razón por la cual toda la interfaz de este componente reside en descriptores de salida estándar en un entorno de terminal.

Pese a todo lo anterior, este componente tiene varias misiones que sí son de su absoluta competencia y son fundamentales para completar el estudio. Como primera medida debe escanear los puertos USB del sistema anfitrión en busca de un lector de tarjetas inteligentes. En caso satisfactorio, construirá un canal de comunicación con el lector a través de su interfaz USB.

Una vez establecida comunicación con el lector, si hay una tarjeta insertada cuya conectividad eléctrica es conforme a lo dispuesto en la norma **ISO/IEC 7816-1** —como se vio en **ISO/IEC 7816**—, realizará un primer test de diálogo con el *chip* de la tarjeta para verificar que se trate de una tarjeta que soporte EMV® —ver **Manual de Usuario**—. Si todo es correcto, pondrá a la escucha el puerto **8102** TCP (*Transport Control Protocol*) —elegido de forma arbitraria— en la máquina anfitriona. Este componente implementa resiliencia a nivel de puertos para cubrir la contigencia de que el puerto 8102 no estuviera disponible. En ese caso irá probando sucesivamente todos los puertos no reservados del sistema hasta conseguir uno disponible.

A partir de entonces, quedará a la espera de conexión remota para transmitir peticiones y respuestas entre el interlocutor remoto y la ICC hasta que se cumpla al menos una de las siguientes: la finalización de la comunicación por el fin controlado de la sesión EMV®, la ruptura del canal físico provocada por ejemplo por la retirada prematura de la tarjeta, o por último la ruptura del canal de transporte por la finalización no controlada de la sesión, por ejemplo por fallo en la conexión del interlocutor remoto.

Cabe señalar además que se implementa también un diseño multihilo de forma que tenga capacidad de atender a diferentes instancias de la *App* **NFC Playground** en sucesivos puertos según necesidad. Aunque el canal sea *half-duplex*, una gestión eficiente permitiría cumplir con las ventana de tiempos vistas en los apartados **3.1.6** y **3.2.3** toda vez que el ciclo completo de recepción, consulta y envío de respuesta no supera los 200 ms como se verá en la siguiente sección.

5.2.2 Arquitectura

NFC Playground Remote Driver está formado por tres clases. Una clase principal que gestiona los hilos y el flujo de la ejecución, una clase que implementa un *socket* TCP (*Transport Control Protocol*) específico para este escenario y una clase de utilidad que implementa los métodos de conexión, test de diálogo, envío y recepción de APDU y conversión de la información en bytes a su representación en hexadecimal.

De forma análoga al apartado dedicado a **NFC Playground**, se presenta el diagrama de clases y sus relaciones en el mismo formato UML.

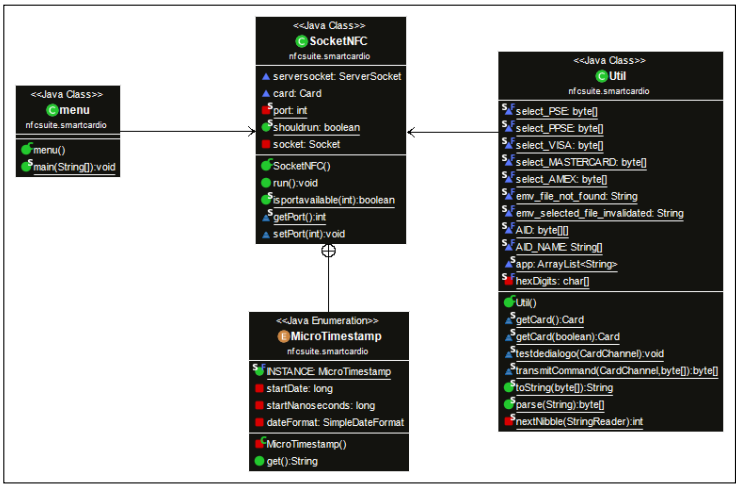


Figura 5.27: Diagrama de Clases de *NFC Playground Remote Driver*

5.2.3 Manual de Usuario

Como se indicaba en **Diseño**, en primer lugar tras su ejecución tendrá lugar un escaneo en busca de lectores de tarjeta. En caso de que ningún dispositivo de estas características responda en ningún puerto USB mostrará el mensaje de la **Figura 5.28**.

Si consigue detectar un lector, intentará averiguar si hay alguna tarjeta insertada. En caso de no encontrarla durante un tiempo de 5 segundos, mostrará el mensaje de la **Figura 5.29**.

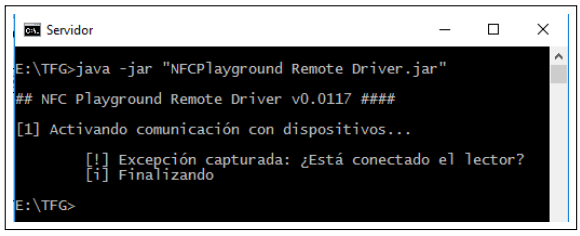


Figura 5.28: Detalle del mensaje tras no encontrar ningún lector

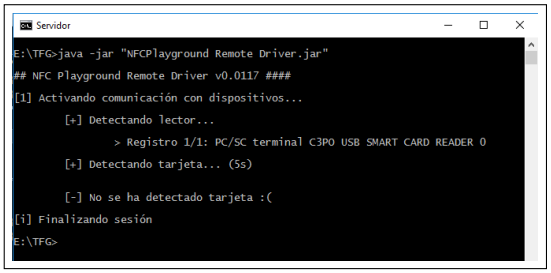


Figura 5.29: Detalle del mensaje tras no encontrar ninguna tarjeta

Si la comunicación con el lector es correcta y se ha detectado una tarjeta, se procederá a realizar un test de diálogo para evaluar la operatividad de la tarjeta así como las aplicaciones que soporta.

A continuación puede verse una captura de todas las fases comentadas hasta ahora. Puede apreciarse el ATR recibido —ir a la sección **ISO/IEC 7816**— cumpliendo con la segunda fase de la sesión EMV® descrita en **Fases de una sesión EMV®**. Es fácil ver que en este caso se ha detectado una tarjeta con los entornos PPSE, PSE —ir al apartado **Acceso a la información**— y aplicación de AMEX. Puede comprobarse viendo cómo la solicitud de estas aplicaciones es respondida con una secuencia acabada en los dos octetos `90:00` correspondientes a la palabra formada por **SW1** y **SW2** —ver **Tabla Campos de una respuesta APDU** [5]—, que en EMV® significa que el comando fue ejecutado de forma satisfactoria. Por el contrario, el APDU de respuesta `6a:82` se interpreta como un código de error con el literal *File not found* (Archivo no encontrado).

Por último, quedará a la escucha en el puerto **8102 TCP** en el hilo correspondiente y a la espera de conexión remota.

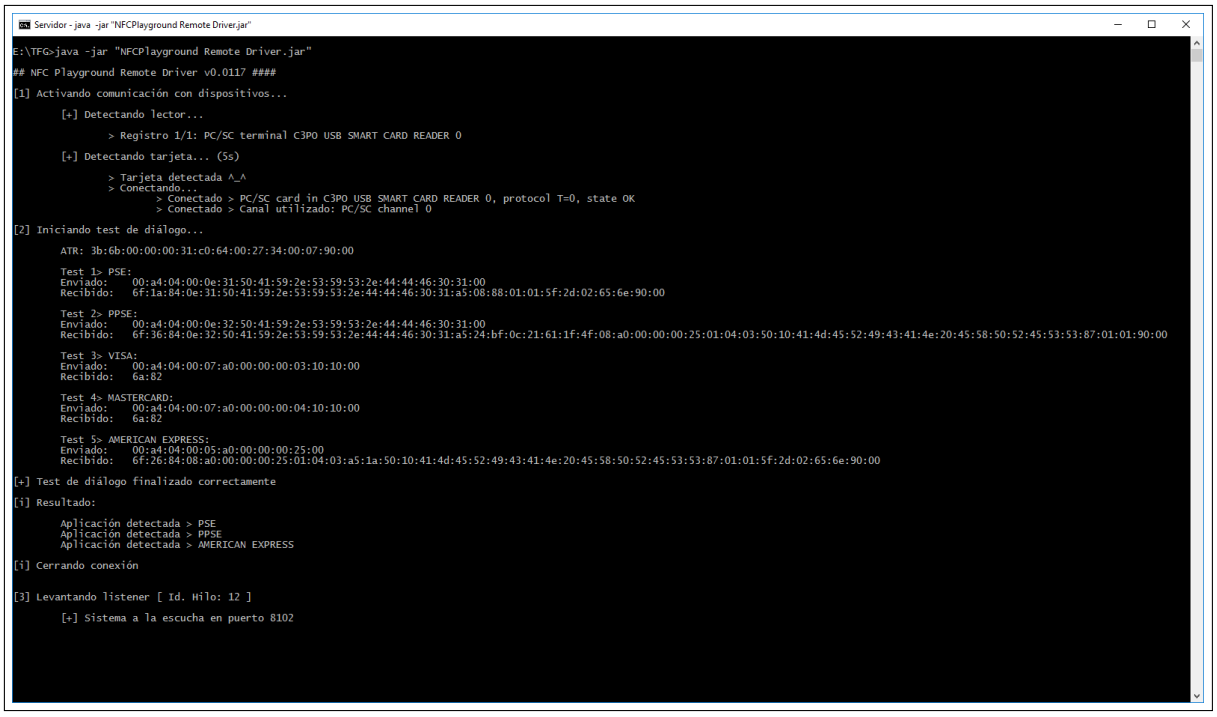


Figura 5.30: Secuencia completa de inicialización de la sesión

Para ilustrar el funcionamiento de *NFC Playground Remote Driver*, en la siguiente página se muestra la secuencia en la que un cliente —diseñado a efectos de pruebas— local dialoga y finaliza una sesión con una ICC a través de *NFC Playground Remote Driver*.

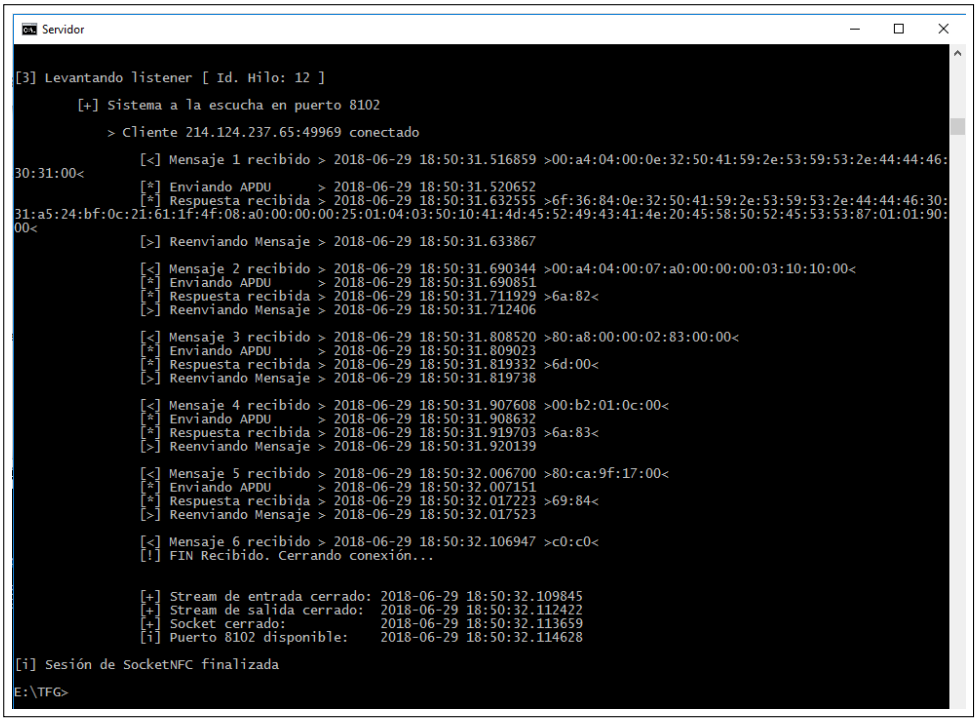


Figura 5.31: Secuencia vista desde *NFC Playground Remote Driver*

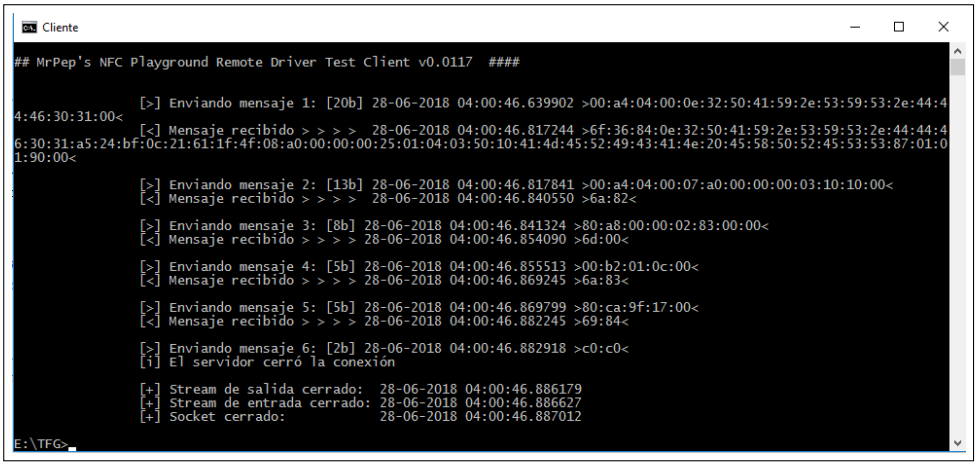


Figura 5.32: La secuencia de la **Figura 5.31** vista desde el cliente

Aunque es una conexión local, puede comprobarse cómo el retardo total —ida y vuelta sobre TCP, retardo de procesamiento del APDU y resto de retardos— apenas supone **178 ms**.

5.3 NFC Playground PIN Pad Demo Component

Este componente tiene como finalidad preparar un entorno local de comunicación con un PCD (PIN Pad) a través de un driver USB de Ingenico —ir a **PIN Pad Ingenico IPP 320**— para realizar una pequeña demostración el día de la defensa de este trabajo. Su único cometido es la gestión de los estados de actividad, reposo o inicialización del PCD a través de un pequeño *batch* o archivo de procesamiento por lotes.
Al arrancar, dispondrá al PCD en modo lectura para habilitarlo a realizar transacciones. Puede verse una imagen de esta secuencia en la siguiente imagen.

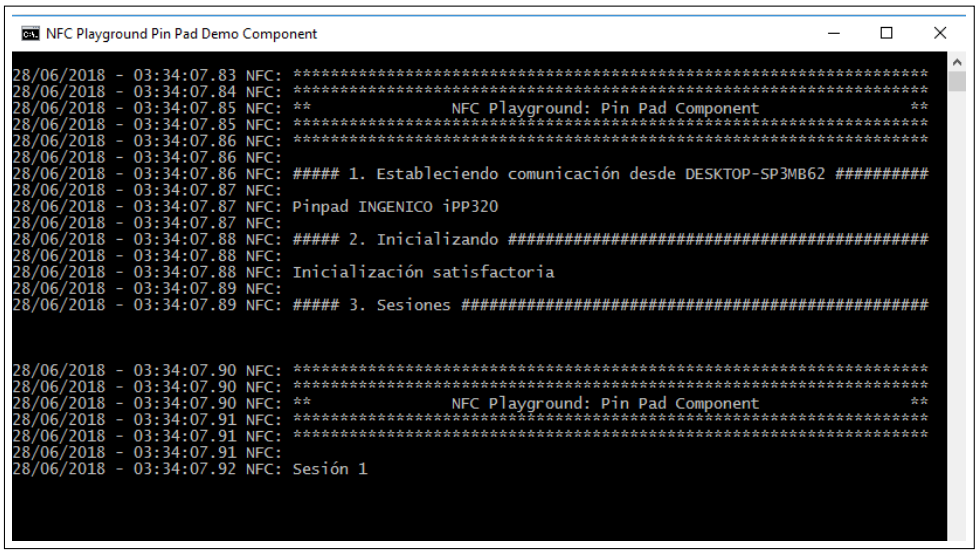


Figura 5.33: Inicialización del PCD para realizar la demostración

Ante un error o *timeout* —agotamiento del tiempo de espera sin detectar *tags*— enviará al PCD a reposo y solicitará confirmación para habilitar nuevamente al PIN Pad. En caso de no contestar en el plazo de 5 segundos volverá a prepararlo para una transacción.

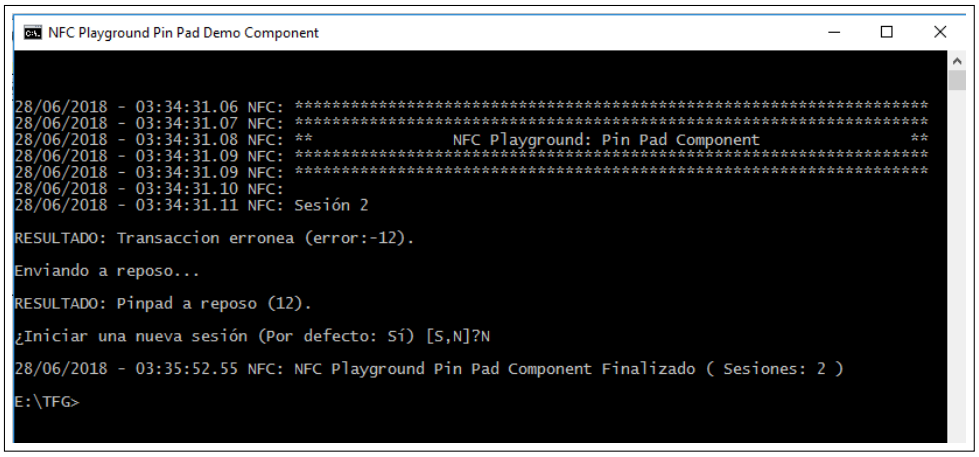


Figura 5.34: Detalle del comportamiento ante un timeout o agotamiento del tiempo de espera sin detectar *tags*

Capítulo 6

Resultados experimentales

Como se ha mencionado con anterioridad en la sección **Acceso a la información**, la estructura de la jerarquía interna de archivos en una ICC depende tanto del emisor como del operador, por tanto las consideraciones se desglosarán de acuerdo con el perfil de cada tarjeta de las descritas en **Material utilizado**.

Si bien es cierto que la existencia de perfiles de ICC sin soporte de *aplicaciones*[1] *contactless* era un hecho conocido con carácter previo al presente trabajo, se ha tenido en consideración incluirlas entre las tarjetas de prueba para tener una representación de este perfil y en el peor de los casos dejar constancia documental del resultado.

Tarjetas VISA

VISA sin soporte PPSE[2] ni PSE[3] y 1 AID (3 unidades)

Este tipo de ICC sería uno de los mencionados en el párrafo anterior al no presentar soporte *contactless*. Se observó cómo ante la solicitud de selección del entorno PPSE por parte del PIN Pad, la tarjeta responde con el APDU de respuesta `6a:82` cuyo literal es *File not found* (Archivo no encontrado), lo que provoca que el PIN Pad aborte la comunicación en cuestión de milisegundos y solicite otro intento sin ofrecer otras aplicaciones alternativas. Tras el fallo de este segundo intento el PIN Pad solicita el paso de la tarjeta por banda magnética —situación conocida como ***fall back*** **de banda**—:

PIN Pad:
`00:a4:04:00:0e:31:50:41:59:2e:53:59:53:2e:44:44:46:30:31:00`

ICC:
`6a:82`

[1] En este contexto se hace referencia al concepto de aplicación visto en **Acceso a la información**
[2] *Proximity Payment System Environment* - Entornos inalámbricos
[3] *Payment System Environment* - Entornos de contactos

Tarjetas *American Express* (AMEX)

1. **AMEX con soporte PPSE y PSE (1 unidad)**

   Se observó cómo ante la solicitud de selección del entorno PPSE por parte del PIN Pad, el APDU de respuesta de la tarjeta —*Field Control Information Template*[4] (FCI)— provoca que el PIN Pad aborte la comunicación hasta llegar al *fall back* de banda:

   PIN Pad:
   `00:a4:04:00:0e:31:50:41:59:2e:53:59:53:2e:44:44:46:30:31:00`

   ICC:
   `6f:36:84:0e:32:50:41:59:2e:53:59:53:2e:44:44:46:30:31:a5:24:bf:0c:21:61:1f:4f:08:a0:00:00:00:25:01:04:03:50:10:41:4d:45:52:49:43:41:4e:20:45:58:50:52:45:53:53:87:01:01:90:00`

2. **AMEX con soporte PPSE (1 unidad)**

   En este caso comprueba que el FCI devuelto por la ICC es idéntico por lo que la resolución de este escenario es idéntica al anterior:

   PIN Pad:
   `00:a4:04:00:0e:31:50:41:59:2e:53:59:53:2e:44:44:46:30:31:00`

   ICC:
   `6f:36:84:0e:32:50:41:59:2e:53:59:53:2e:44:44:46:30:31:a5:24:bf:0c:21:61:1f:4f:08:a0:00:00:00:25:01:04:03:50:10:41:4d:45:52:49:43:41:4e:20:45:58:50:52:45:53:53:87:01:01:90:00`

3. **AMEX con soporte PPSE y un AID (5 unidades)**

   Este perfil, que tiene soporte *contactless* y no soporta el entorno PSE, soporta además dos AID adicionales: `a0:00:00:00:25:01` y `a0:00:00:00:25:01:09`. Se observa que el PIN Pad tampoco acepta este FCI y aborta la comunicación inmediatamente como en casos anteriores:

   PIN Pad:
   `00:a4:04:00:0e:31:50:41:59:2e:53:59:53:2e:44:44:46:30:31:00`

   ICC:
   `6f:2c:84:08:a0:00:00:00:25:01:09:01:a5:20:50:10:41:4d:45:52:49:43:41:4e:20:45:58:50:52:45:53:53:9f:38:03:9f:6e:04:87:01:01:5f:2d:02:65:6e:90:00`

4. **AMEX con soporte PSE y un AID (7 unidades)**

   Estas tarjetas no tienen funcionalidad *contactless* y únicamente soportan el entorno PSE y el AID tradicional de AMEX: `a0:00:00:00:25:01:09:01`. En una aproximación directa como en casos anteriores, el diálogo fue idéntico que el visto en el caso de las tarjetas VISA:

[4] `https://emvlab.org/emvtags/show/t6F`

PIN Pad:
00:a4:04:00:0e:31:50:41:59:2e:53:59:53:2e:44:44:46:30:31:00

ICC:
6a:82

Sin embargo, como aproximación alternativa y teniendo en cuenta que esta tarjeta soporta PSE, modificando ligeramente el código de **NFC Playground** se intentó manipular el FCI que devolvería al PIN Pad tomando como referencia la respuesta ante la selección de entornos PSE[5]:

6f:1a:84:0e:**31**:50:41:59:2e:53:59:53:2e:44:44
:46:30:31:a5:08:88:01:01:5f:2d:02:65:6e:90:00

cambiando la parte correspondiente a PSE —en negrita— del FCI por la correspondiente a PPSE, quedando el FCI de la siguiente manera:

6f:1a:84:0e:**32**:50:41:59:2e:53:59:53:2e:44:44
:46:30:31:a5:08:88:01:01:5f:2d:02:65:6e:90:00

En este caso, se observa que **el PIN Pad da por bueno el FCI manipulado y** dado que la ICC se presenta como AMEX, el PIN Pad **continúa la conversación** solicitando la selección de aplicación particular de AMEX (00:a4:04:02:07:a0:00:00:00:25:01:00), ante lo que finalmente la ICC responde con el APDU 6a:80 cuya interpretación es *The parameters in the data field are incorrect* (Los parámetros en el campo de datos son incorrectos) tras lo cual el PIN Pad aborta la comunicación. Esta sería la secuencia:

PIN Pad:
00:a4:04:00:0e:31:50:41:59:2e:53:59:53:2e:44:44:46:30:31:00

ICC:
6f:1a:84:0e:32:50:41:59:2e:53:59:53:2e:44:44:46:30:31:a5:08:88:01:01:5f:
2d:02:65:6e:90:00

PIN Pad:
00:a4:04:02:07:a0:00:00:00:25:01:00

ICC:
6a:80

Este comportamiento pudo reproducirse en todas las tarjetas AMEX con soporte de entornos PSE.

[5] 00:a4:04:00:0e:31:50:41:59:2e:53:59:53:2e:44:44:46:30:31:00

Capítulo 7

Conclusiones, limitaciones y líneas futuras

7.1 Conclusiones

El objetivo principal de este estudio tal y como se vio en la sección **Objetivos** era conocer si era posible establecer una sesión EMV® basada en HCE partiendo de otra basada en contactos, quedando la ICC en tal caso expuesta a la posibilidad de realizar transacciones sin autenticación de usuario si éstas fueran de un importe inferior al de Floor Limit por la naturaleza de esta funcionalidad. Según se ha documentado en el capítulo **Resultados experimentales**, **no es posible completar la sesión EMV®**.

El segundo de los objetivos definidos era implícito a este viaje: profundizar en el desarrollo de aplicaciones en el universo Android. Se ha conseguido diseñar y desarrollar las herramientas necesarias para desplegar el escenario descrito en el **Diagrama del escenario inicial** y, para completar el producto final, la App *NFC Playground* ha sido provista de una interfaz intuitiva, cómoda y con funcionalidades propias de *wallets* comerciales desarrollados por operadores bancarios. La envergadura de este desarrollo ha permitido tomar el testigo del desarrollo de aplicaciones para sistemas operativos Android donde lo dejó la formación recibida en la Escuela y extenderlo a ámbitos de una complejidad mucho mayor. Lo anterior tiene, más aún si cabe, mayor relevancia en una época en la que las ICC de medios de pago tal y como las conocemos tienen fecha de caducidad y el sector está apostando fuertemente a favor de HCE.

Gracias a las herramientas anteriores se ha conseguido evaluar y documentar el comportamiento de una sesión EMV® ante el escenario propuesto, que ha sido el principal hilo conductor del presente estudio. Así, como se comentaba al inicio de estas conclusiones, si bien una aproximación directa en la que la retransmisión de la información desde el PCD a la ICC sin ningún tipo de modificación no ha permitido la emulación completa de la sesión por las razones vistas en **Resultados experimentales**, el escenario en el que el PIN Pad ha continuado la conversación tras manipular el FCI de una tarjeta sin funcionalidad *contactless* debe considerarse muy positivo en la medida que abre una linea de investigación muy interesante alrededor de la posibilidad de

extender la sesión EMV® hasta su finalización en un entorno como el que plantea este estudio mediante la modificación de los APDU de respuesta.

7.2 Limitaciones

En contra de las estimaciones iniciales en las que se atribuía a la ventana de tiempos contemplada en las normas ISO/IEC 7618 e ISO/IEC 14443 un factor condicionante, uno de los puntos más limitantes que se ha revelado en el estudio ha sido el comportamiento del PIN Pad al ofrecer como única alternativa de pago el entorno PPSE. Si bien es cierto que existen dispositivos que ante la ausencia de entornos PPSE optan por una aplicación alternativa, el PIN Pad de este estudio únicamente ofrece este entorno de pago.

7.3 Líneas futuras

Teniendo en cuenta los resultados experimentales obtenidos, se abre una línea de trabajo en la que se debería estudiar la manipulación de los *templates* en los APDU de respuesta del ICC e incorporar dicho procedimiento a la lógica de NFC Playground. Puesto que una sesión completa en escenarios de *Floor Limit* en los que no llega a tener lugar la fase de autenticación del titular puede constar de 5 preguntas y respuestas entre PCD e ICC —en función del perfil de la ICC y del PIN Pad— como se vio en ISO/IEC 7816, los resultados obtenidos nos situarían en una posición más próxima a completar la sesión de cara a investigaciones futuras.

Del mismo modo, se entiende que el estudio debería extenderse a un abanico mucho más extenso tanto de ICC como de PCD para estudiar el impacto de sus prestaciones en el marco tanto de una aproximación directa —como la planteada en este estudio— como en el de una variante que contemple la manipulación de los APDU.

Referencias bibliográficas

[1] I. O. for Standardization. (2003). Financial transaction card originated messages, dirección: https://www.iso.org/obp/ui/#iso:std:iso:8583:-1:ed-1:v1:en (vid. pág. 2).

[2] Gemalto. (2018). EMV security vastly reduces card fraud in the US, dirección: https://www.gemalto.com/emv/contactless-us/emv-security (vid. pág. 2).

[3] E. LLC, «EMVCo Security Guidelines», *EMVCo Documentation*, vol. 1, n.º 5.1, pág. 38, 2017. DOI: https://www.emvco.com/wp-content/uploads/2017/04/EMVCo-SEWG-14-P02-V5-1_EMVCo_Security_Evaluation_Process_20160725082101992.pdf (vid. pág. 2).

[4] StatCounter. (2018). Mobile Operating System Market Share Worldwide May 2017 - May 2018, dirección: http://gs.statcounter.com/os-market-share/mobile/worldwide (vid. pág. 2).

[5] E. LLC, «Application Specification», *EMVCo Documentation*, vol. 3, n.º 4.3, pág. 230, 2011. DOI: https://www.emvco.com/wp-content/uploads/2017/05/EMV_v4.3_Book_3_Application_Specification_20120607062110791.pdf (vid. págs. 3, 14, 16, 19, 20, 63).

[6] T. S. P. A. e.V. (2016). An Overview of Contactless Payment Benefits and Worldwide Deployments, dirección: http://www.smartpaymentassociation.com/images/news/16-04-26-SPA-Contactless-Payment-Benefits-WP-Final.pdf (vid. pág. 4).

[7] V. Europe. (2015). 1 billion Visa contactless purchases made in last year, dirección: https://www.visaeurope.com/media/pdf/26104.pdf (vid. pág. 4).

[8] E. LLC. (2018). EMVCo Deployment Statistics, dirección: https://www.emvco.com/about/deployment-statistics (vid. pág. 8).

[9] S. LLC. (2018). Android operating system share worldwide by OS version from 2013 to 2018, dirección: https://www.statista.com/statistics/271774/share-of-android-platforms-on-mobile-devices-with-android-os/ (vid. pág. 10).

[10] ——, (2018). Smartphone unit shipments worldwide by operating system from 2016 to 2022 (in million units), dirección: https://www.statista.com/statistics/309448/global-smartphone-shipments-forecast-operating-system/ (vid. pág. 11).

[11] I. O. for Standardization. (1987). Part 1: Physical characteristics, dirección: https://www.iso.org/standard/14732.html (vid. pág. 13).

[12] ——, (2011). Part 1: Cards with contacts - Physical characteristics, dirección: https://www.iso.org/standard/54089.html (vid. págs. 13, 20).

[13] ——, (2007). Part 2: Cards with contacts - Dimensions and location of the contacts, dirección: https://www.iso.org/standard/45989.html (vid. pág. 13).

[14] ——, (2006). Part 3: Cards with contacts - Electrical interface and transmission protocols, dirección: https://www.iso.org/standard/38770.html (vid. pág. 13).

[15] ——, (2013). Part 4: Organization, security and commands for interchange, dirección: https://www.iso.org/standard/54550.html (vid. págs. 13, 18).

[16] ——, (2004). Part 5: Registration of application providers, dirección: https://www.iso.org/standard/34259.html (vid. pág. 13).

[17] ——, (2016). Part 6: Interindustry data elements for interchange, dirección: https://www.iso.org/standard/64598.html (vid. pág. 13).

[18] ——, (1999). Part 7: Interindustry commands for Structured Card Query Language (SCQL), dirección: https://www.iso.org/standard/28869.html (vid. pág. 13).

[19] ——, (2016). Part 8: Commands and mechanisms for security operations, dirección: https://www.iso.org/standard/66092.html (vid. pág. 13).

[20] ——, (2017). Part 9: Commands for card management, dirección: https://www.iso.org/standard/67802.html (vid. pág. 13).

[21] ——, (1999). Part 10: Electronic signals and answer to reset for synchronous cards, dirección: https://www.iso.org/standard/30558.html (vid. pág. 13).

[22] ——, (2017). Part 11: Personal verification through biometric methods, dirección: https://www.iso.org/standard/67799.html (vid. pág. 13).

[23] ——, (2005). Part 12: Cards with contacts - USB electrical interface and operating procedures, dirección: https://www.iso.org/standard/40604.html (vid. pág. 13).

[24] ——, (2007). Part 13: Commands for application management in a multi-application environment, dirección: https://www.iso.org/standard/40605.html (vid. pág. 13).

[25] ——, (2016). Part 15: Cryptographic information application, dirección: https://www.iso.org/standard/65250.html (vid. pág. 13).

[26] E. LLC, «Application Independent ICC to Terminal Interface Requirements», *EMVCo Documentation*, vol. 1, n.º 4.3, pág. 189, 2011. DOI: https://www.emvco.com/wp-content/uploads/2017/04/EMV_v4.3_Book_1_ICC_to_Terminal_Interface_2012060705394541.pdf (vid. págs. 14, 15).

[27] V. Europe. (2016). Europeans "touched to pay" three billion times in the last 12 months, dirección: https://www.visaeurope.com/media/pdf/36283.pdf (vid. pág. 20).

[28] N. Forum. (2018). NFC Forum Technical Specifications, dirección: https://nfc-forum.org/our-work/specifications-and-application-documents/specifications/nfc-forum-technical-specifications/ (vid. págs. 21, 24).

[29] I. O. for Standardization. (2013). Near Field Communication Interface and Protocol (NFCIP-1), dirección: https://www.iso.org/standard/56692.html (vid. págs. 21-23).

[30] ——, (2018). Cards and security devices for personal identification - Contactless proximity objects - Part 1: Physical characteristics, dirección: https://www.iso.org/standard/73596.html (vid. pág. 21).

[31] ——, (2016). Identification cards - Contactless integrated circuit cards - Proximity cards - Part 2: Radio frequency power and signal interface, dirección: https://www.iso.org/standard/66288.html (vid. pág. 21).

[32] ——, (2018). Cards and security devices for personal identification - Contactless proximity objects - Part 3: Initialization and anticollision, dirección: https://www.iso.org/standard/73598.html (vid. págs. 21, 22).

[33] ——, (2016). Identification cards – Contactless integrated circuit cards - Proximity cards - Part 4: Transmission protocol, dirección: https://www.iso.org/standard/70172.html (vid. págs. 21, 27).

[34] N. Forum. (2006). NFC Data Exchange Format (NDEF), dirección: https://nfc-forum.org/product/nfc-data-exchange-format-ndef-technical-specification/ (vid. págs. 23, 26).

[35] I. O. for Standardization. (2012). Near Field Communication Interface and Protocol 2 (NFCIP-2), dirección: https://www.iso.org/standard/56855.html (vid. pág. 23).

[36] M. H. Thomas Korak, «On the power of active relay attacks using custom-made proxies», *2014 IEEE International Conference on RFID*, 2014. DOI: https://doi.org/10.1109/RFID.2014.6810722 (vid. pág. 24).

[37] E. T. S. Organization. (2018). ETSI SIM Standard List, dirección: http://www.etsi.org/technologies-clusters/technologies/smart-cards/sim#standards (vid. pág. 25).

[38] I. E. T. Force. (1998). Uniform Resource Identifiers (URI): Generic Syntax, dirección: https://tools.ietf.org/html/rfc2396 (vid. pág. 27).

[39] ——, (2016). MIME (Multipurpose Internet Mail Extensions) Part Three: Message Header Extensions for Non-ASCII Text, dirección: https://tools.ietf.org/html/rfc2047 (vid. pág. 27).

[40] G. LLC. (2016). How NFC tags are mapped to MIME types and URIs, dirección: https://developer.android.com/guide/topics/connectivity/nfc/nfc#ndef (vid. pág. 27).

[41] ——, (2018). Intent, dirección: `https://developer.android.com/reference/android/content/Intent` (vid. pág. 27).

[42] ——, (2018). AsyncTask, dirección: `https://developer.android.com/reference/android/os/AsyncTask` (vid. pág. 30).

[43] G. H. J. Vlissades, «Design Patterns: Elements of Reusable Object-Oriented Software», *Addison-Wesley*, pág. 293, 1995. DOI: `http://dx.doi.org/10.1081/E-ESE-120044217` (vid. pág. 39).

[44] R. Eckstein. (2007). Java SE Application Design With MVC, dirección: `http://www.oracle.com/technetwork/articles/javase/mvc-136693.html` (vid. pág. 39).

[45] R. R. Painter, «Software Plans: Multi-Dimensional Fine-Grained Separation of Concerns», *Penn State*, 2014. DOI: `https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.110.9227` (vid. pág. 49).

Apéndice A

ANEXOS

A.1 ANEXO I: Diagrama de Gantt

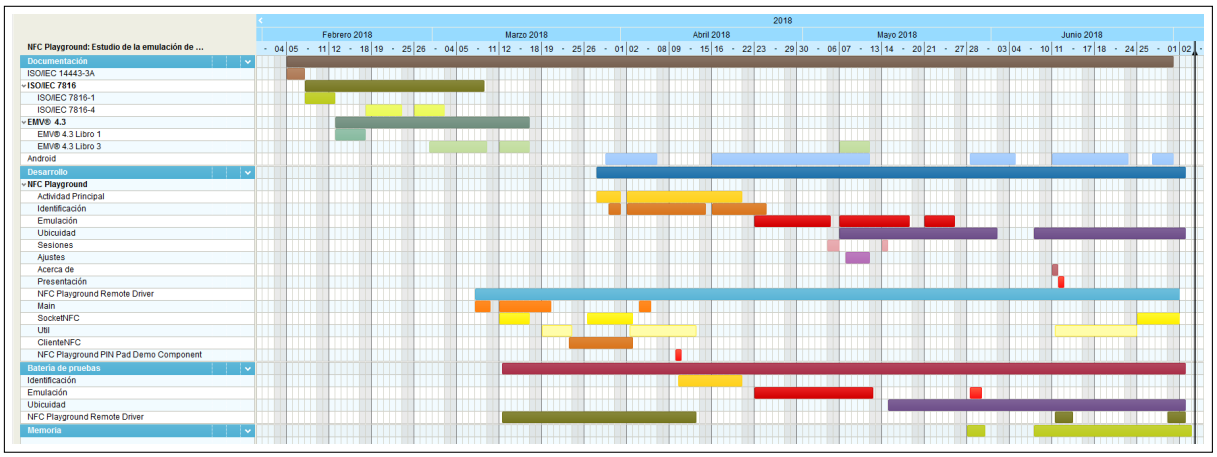


Figura A.1: Diagrama de Gantt con la distribución del tiempo por tareas

A.2 ANEXO II: Floor Limit mundial

Espacio Económico	Floor Limit	Espacio Económico	Floor Limit
Alemania	25€	Islandia	ISK 5.000
Australia	A$100	Italia	25€
Austria	25€	Japón	JP¥20000
Bélgica	25€	Letonia	25€
Brasil	R$50	Lituania	25€
Bulgaria	25 BGN	Malasia	RM250
Canadá	CA$100	Noruega	200 NOK
Chile	$12.000 CLP	Nueva Zelanda	NZ$80
China	CN¥ 1000	Polonia	50 PLN
Croacia	100 HRK	Portugal	20€
Dinamarca	350 DKK	Reino Unido	£30
EEUU	0	República Checa	500 CZK
Eslovaquia	20€	República de Macedonia	750 MKD
Eslovenia	15€	República Dominicana	0
España	20€	Rumanía	100 Lei
Estonia	25€	Rusia	1000€
Eurozona	25€	Serbia	2000RSD
Finlandia	25€	Singapur	S$ 100
Francia	30€	Sudáfrica	500 ZAR
Grecia	25€	Suecia	200 SEK
Holanda	25€	Suiza	40 CHF
Hong Kong	0	Tailandia	1500€
Hungría	5000 HUF	Taiwan	0
India	Rs. 2000	Turquía	90 TL
Irlanda	30€	Ucrania	100 UAH

https://www.mastercard.at	http://www.unionpayintl.com
https://www.swedbank.ee	http://www.jcb.co.jp
https://ib.swedbank.lt	http://www.bnm.gov.my
https://www.ccv.nl	https://www.visa.com
http://www.pbzcard.hr	https://www.mastercard.com.au