# Universidad de las Ciencias Informáticas

# Facultad 1


**Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas.**

**Título:** Aplicación móvil para la autenticación con servidor *proxy* web en NovaDroid.

**Autor:** Daniel Alejandro Rodríguez Caballero

**Tutores:** Ing. Asist. Gladys Marsi Peñalver Romero

Ing. Leonardo Quesada Cruz

**Ciudad de la Habana, Cuba, junio 2018.**

"*… yo soy perfectamente feliz con aquellas personas que miran las nubes y las estrellas y dicen: yo quiero ir allí. Pero yo estoy mirando al piso, y quiero arreglar el bache que está justo al frente antes de caer en él. Ese es el tipo de persona que soy.*"

Linus Torvalds

## Declaración de Autoría

Yo, Daniel Alejandro Rodríguez Caballero, declaro ser el autor del presente trabajo de diploma titulado "Aplicación móvil para la autenticación con servidor *proxy* web en NovaDroid" y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales del mismo.

Para que así conste firmo la presente a los ____ días del mes de _____ del año 2018.

_____

Autor

Daniel Alejandro Rodríguez Caballero

_____

Tutora

Ing. Asist. Gladys Marsi Peñalver Romero

_____

Tutor

Ing. Leonardo Quesada Cruz

## Dedicatoria

A mis padres, sin duda, a ellos.

## Agradecimientos

A todas las personas que han estado cerca, todas han contribuido directa o indirectamente: familiares, amigos y compañeros, tutores y profesores. A la universidad, por sus herramientas y procesos para alcanzar conocimiento. A nuestro sistema, por este privilegio de estudiar.

## Resumen

NovaDroid es un sistema operativo móvil desarrollado en la Universidad de las Ciencias Informáticas. Constituye una personalización de Android que tiene como objetivo adaptarlo a las necesidades de los usuarios en Cuba. En la actualidad está siendo instalado y distribuido en las tabletas producidas por la Empresa Industrial para la Informática, las Comunicaciones y la Electrónica. Este sistema operativo móvil no permite la autenticación con servidor *proxy* web a las aplicaciones que se ejecutan sobre él, imposibilitando así el acceso a Internet a través de estos servidores utilizados ampliamente en las instituciones cubanas. Debido a esta problemática se realizó la presente investigación con el objetivo de desarrollar una aplicación móvil que permita la autenticación con servidor *proxy* web en el sistema operativo NovaDroid. Se utilizó Visual Paradigm como herramienta de modelado, Java como lenguaje de programación y Android Studio como entorno de desarrollo. Se obtuvo como resultado el completo desarrollo de la aplicación. Su funcionamiento fue evaluado mediante la aplicación de pruebas, proceso que culminó con la aceptación por parte del cliente.

**Palabras clave:** autenticación, Internet, NovaDroid, servidor *proxy*.

## Abstract

NovaDroid is a mobile operating system developed at the University of Informatics Sciences. It consists of an Android customization that aims to adapt it to the needs of users in Cuba. It's currently being installed and distributed in the tablets produced by the Industrial Company for Information Technology, Communications and Electronics. This mobile operating system doesn't allow authentication with a web proxy server to the applications that run on it, making it impossible to access the Internet through these servers that are widely used in Cuban institutions. Due to this problem, the present investigation was carried out in order to develop a mobile application that allows authentication with a web proxy server in the NovaDroid operative system. Visual Paradigm was used as a modeling tool, Java as a programming language and Android Studio as a development environment. The complete development of the application was obtained as a result. Its performance was evaluated through the application of tests, a process that culminated with the acceptance by the client.

**Keywords:** authentication, Internet, NovaDroid, proxy server.

# Índice de contenido

## Introducción

Internet, poderosa red conformada por millones de computadoras interconectadas que comparten información todo el tiempo; brinda un amplio conjunto de servicios entre los que destacan: la WWW[1] *(World Wide Web*), envío de correo electrónico, transmisión de archivos, mensajería instantánea y comunicación multimedia. El 30 de junio del 2017 se reportaron un total de 3 885 567 619 usuarios de Internet en todo el mundo alcanzando un rango de penetración del 51.7% de la población total («*World Internet Users Statistics and 2017 World Population Stats*» 2017).

El acceso a la red de redes en Cuba y en el mundo es efectuado diariamente por una gran variedad de tecnologías, dentro de estas, una de las más recurrentes son los dispositivos móviles como: teléfonos inteligentes (*smartphones*) y tabletas (*tablets*), debido principalmente a su gran portabilidad y facilidad de uso. Según StatCounter («*Mobile and tablet internet usage exceeds desktop for first time worldwide*» 2016) en octubre del 2016 los teléfonos inteligentes y las tabletas abarcaron un 51.3% del uso de Internet en todo el mundo mientras que el 48.7% restante fue ocupado por los dispositivos de escritorio (PCs y laptops). Las tecnologías móviles se están convirtiendo en las nuevas PCs, los consumidores están usando cada vez más *smartphones* y *tablets* en lugar de las computadoras de escritorio para acceder a productos y servicios (Rauch 2011).

Los dispositivos móviles basan su funcionamiento en los sistemas operativos (SO) móviles, estos constituyen un conjunto de programas de bajo nivel que permiten la abstracción de las peculiaridades del hardware específico del dispositivo y proveen servicios a las aplicaciones que se ejecutan sobre él. Existen en la actualidad varios SO móviles entre los que destacan: Android, iOS y Windows Phone. En la UCI (Universidad de las Ciencias Informáticas), Cuba, se desarrolló un SO móvil nombrado NovaDroid, el cual, es una personalización de Android que tiene como objetivo adaptarse a las condiciones tecnológicas del país, y de esa manera facilitar y extender su uso. En la actualidad está siendo ampliamente distribuido a través de su instalación en las tabletas producidas por GEDEME (Empresa Industrial para la Informática, las Comunicaciones y la Electrónica), lo que implica que está siendo usado por numerosas personas en el país.

Gran parte del acceso a Internet en Cuba se realiza a nivel institucional (universidades o empresas)

1 *World Wide Web*: conjunto de protocolos que permite, de forma sencilla, la consulta remota de hipertexto.

mediante servidores *proxy* web. Un servidor *proxy* web constituye un intermediario que intercepta los mensajes HTTP (Protocolo de Transferencia de Hipertexto) entre la conexión del cliente y el servidor objetivo (Gourley y Totty 2002). Su uso permite el aseguramiento de la red local para la restricción del acceso a contenido inseguro, establecimiento de políticas de acceso, brindar velocidad en las conexiones, así como el ahorro de recursos. En orden de ejercer el control de acceso de los usuarios de la red los servidores *proxy* web realizan un proceso de autenticación el cual bloquea las peticiones del cliente hasta que sus credenciales (usuario y contraseña) sean correctas.

El sistema operativo NovaDroid no permite realizar el proceso antes mencionado, es decir no permite la autenticación con servidor *proxy* web. Esta situación provoca que las aplicaciones que no tengan implementado por ellas mismas este mecanismo no pueden acceder a Internet*,* por tal razón, se imposibilita el uso de la mayoría de las *aplicaciones clientes*[2] como *Play Store* o *Google Maps,* actualizar los datos de la mayor cantidad de las aplicaciones estándares o actualizar el sistema operativo mediante Internet. En general se pierden una gran parte de las funcionalidades que brinda la conexión. Además, implica que si en un futuro se desarrollan soluciones nacionales que necesitan autenticarse con un servidor *proxy* web, se deba implementar en cada una de ellas el soporte para realizar este proceso.

Teniendo en cuenta la situación anterior como punto de partida surge el siguiente **problema de investigación**: ¿cómo permitir la autenticación con servidor *proxy* web en el sistema operativo NovaDroid para el acceso a Internet?

Para dar respuesta a la problemática planteada la presente investigación centra su **objeto de estudio** en el proceso de autenticación con servidor *proxy* web, delimitándose como **campo de acción** el proceso de autenticación con servidor *proxy* web en el sistema operativo NovaDroid.

Se establece entonces como **objetivo general**: desarrollar una aplicación móvil que permita la autenticación con servidor *proxy* web en el sistema operativo NovaDroid.

Este se desglosa en los siguientes **objetivos específicos**:

- Investigar los aspectos teóricos fundamentales que sustentan el proceso de autenticación con un servidor *proxy* web.
- Diseñar la aplicación móvil para la autenticación con servidor *proxy* web en NovaDroid.

2 Aplicación cliente: aplicación informática que consume un servicio remoto en otro ordenador.

- Implementar la aplicación móvil para la autenticación con servidor *proxy* web en NovaDroid.
- Evaluar el correcto funcionamiento de la solución.

Con el propósito de dar cumplimiento a lo previamente planteado se definen las siguientes **tareas de investigación**:

- Definición de los elementos teóricos relacionados con la autenticación con servidor *proxy* web.
- Análisis de las diferentes soluciones informáticas que permiten la autenticación con servidor *proxy* web.
- Selección de las tecnologías y la metodología más adecuada para el desarrollo de la solución.
- Identificación de los requisitos de la solución a desarrollar.
- Diseño de la aplicación móvil para la autenticación con servidor *proxy* web en NovaDroid.
- Codificación de la aplicación móvil para la autenticación con servidor *proxy* web en NovaDroid.
- Realización de las pruebas necesarias para identificar posibles errores en la solución.

Se establece en la presente investigación la siguiente **idea a defender**: el desarrollo de una aplicación móvil que permita la autenticación con servidor proxy web en NovaDroid garantizará el acceso a Internet en el dispositivo.

Para el desarrollo de la presente investigación se hace uso de los siguientes **métodos científicos**:

**Teóricos**

- **Analítico-Sintético**: es utilizado con el objetivo de estudiar los elementos más significativos del proceso de autenticación con servidor *proxy* web, el cual es analizado y descompuesto en sus múltiples relaciones y componentes para facilitar su estudio.

**Empíricos:**

- **Observación**: es utilizado para realizar un análisis de las diferentes soluciones que resuelven de forma parcial el problema de investigación planteado.
- **Entrevista**: es utilizado con el propósito de obtener información de especialistas acerca de los esquemas de autenticación más utilizados en los servidores *proxy* web de las instituciones en Cuba.

El presente documento está desglosado en los siguientes capítulos:

- **Capítulo 1. Fundamentación teórica:** Estudio de los principales conceptos y elementos teóricos

asociados a la investigación con el objetivo de lograr una mayor comprensión de la misma. Se realiza además un estudio de las principales soluciones que permiten la autenticación con servidor *proxy* web, se analiza el estado del arte de las mismas teniendo en cuenta sus ventajas y sus desventajas. Por último, se establecen las tecnologías que servirán para el posterior desarrollo de la solución y se selecciona la metodología que guiará su proceso de desarrollo.

- **Capítulo 2. Análisis y diseño**: Se describe la solución propuesta, se identifican los requisitos funcionales y no funcionales y sus descripciones mediante las historias de usuario. Además, se establece la arquitectura a utilizar y los patrones de diseño.
- **Capítulo 3. Implementación y prueba**: Se describe el proceso de implementación de la solución. Se establecen los estándares de codificación para la implementación y por último se detallan las pruebas realizadas con el objetivo de evaluar correcto funcionamiento de la aplicación resultante.

# Capítulo 1. Fundamentación teórica

En el presente capítulo se analizan los conceptos y elementos teóricos relacionados con la autenticación con servidor *proxy* web. Se plasma además el resultado del estudio realizado a las principales soluciones que permiten la autenticación con servidor *proxy* web. Por último, se describe la metodología seleccionada para guiar el proceso de desarrollo de software y las tecnologías que se seleccionaron para el diseño e implementación de la solución.

## 1.1 Conceptos asociados al dominio del problema

Con el objetivo de realizar un correcto proceso de investigación se detallan a continuación los principales conceptos asociados al dominio del problema.

### *1.1.1 HTTP*

Constituye el protocolo de control e instrucción usado para gestionar las comunicaciones entre una aplicación web y un servidor web. Es el lenguaje común del Internet moderno (Gourley y Totty 2002).

### *1.1.2 Servidor proxy web*

Un servidor *proxy* web actúa como un intermediario entre los dos extremos de una conexión de red cliente / servidor; se ubica entre el cliente y el servidor y actúa como un “hombre en el medio”, interceptando mensajes HTTP entre las partes (Gourley y Totty 2002).

Existen además dos tipos de *proxy* web atendiendo a quien quiere implementar las funcionalidades de este:

- ***Proxy* local**: en este caso el que quiere implementar las funcionalidades, es el mismo que realiza la petición. Por esta razón se llama local. Suelen estar en el mismo dispositivo donde el cliente hace las peticiones. Son usados para el control del tráfico del cliente, establecer reglas de filtrado y además llevar a cabo procesos de autenticación con páginas web u otros servidores *proxy*.
- ***Proxy* de red o *proxy* de entorno**: el que quiere implementar las funcionalidades es una entidad externa. Se suelen usar para bloquear contenidos, controlar el tráfico de la red y compartir direcciones de IP (Protocolo de Internet, por sus siglas en inglés).

### *1.1.3 Autenticación*

Proceso que valida la información de inicio de sesión de un usuario a un determinado sistema. La autenticación puede involucrar la comparación del nombre de usuario y contraseña con una lista de usuarios autorizados; si es encontrada una coincidencia el usuario puede acceder al sistema en correspondencia con los derechos o permisos asignados a su cuenta (Dyson y Shafer 2000).

### *1.1.4 Esquema de autenticación*

Un esquema, política o protocolo de autenticación es un método de cifrado especialmente diseñado para transferir datos de autenticación entre dos entidades. Constituye la capa más importante de protección para la comunicación segura en las redes de computadora (Duncan 2001).

## 1.2 Autenticación con servidor *proxy* web

En el presente epígrafe se detallan los aspectos teóricos relacionados con la autenticación con servidor *proxy* web con el objetivo de alcanzar un mayor entendimiento de su funcionamiento y aplicar los conocimientos obtenidos en la solución a desarrollar.

Este proceso de autenticación está basado enteramente en un *framework*[3] definido por HTTP y en los esquemas de autenticación, aspectos que se explican a continuación en las siguientes secciones.

### *1.2.1 Framework de autenticación HTTP*

La RFC (*Request For Critics*) 7235 define el *framework* de autenticación HTTP, el cual, es usado por un servidor para autenticar a los usuarios que acceden a él. Este proceso consta de los siguientes pasos («*HTTP authentication*» 2017; Gourley y Totty 2002):

1. El cliente realiza una petición de un recurso al servidor.
2. El servidor responde que para acceder al recurso son necesarias las credenciales del usuario.
3. El cliente vuelve a realizar la petición con las credenciales del usuario incluidas.
4. El servidor comprueba que las credenciales sean válidas, y si es el caso, envía el recurso requerido al cliente.

Para una mayor comprensión del proceso se presenta en la *Figura 1* una ejemplificación de este.

3 *Framework*: marco de trabajo o estructura para realizar determinada tarea.

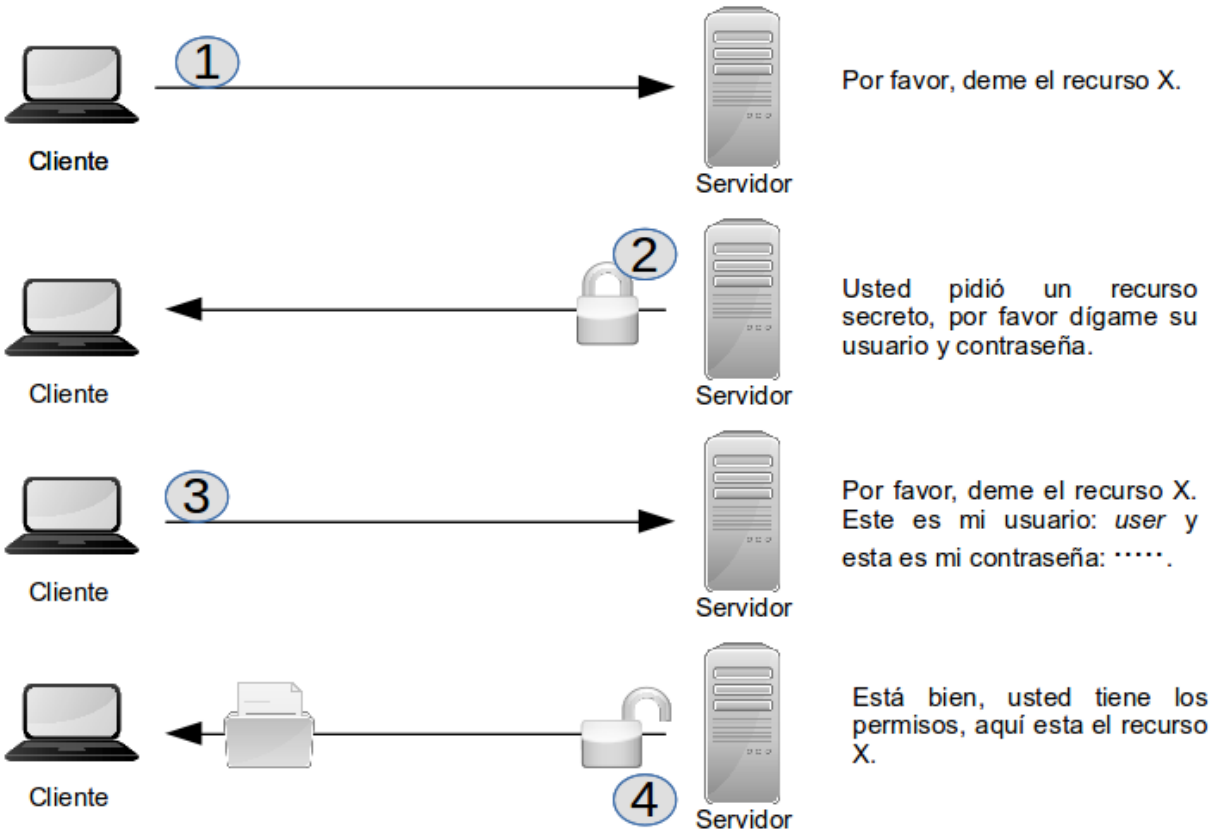



*Figura 1: Proceso de autenticación HTTP.*

### *1.2.2 Autenticación proxy*

El mismo mecanismo de autenticación detallado anteriormente es utilizado para la autenticación con un servidor *proxy*, con la diferencia de que en este caso el servidor *proxy,* el intermediario, es el responsable de realizar el proceso para resolver o no la petición del cliente («*HTTP authentication*» 2017). La *Figura 2* detalla el uso de este mecanismo para la autenticación *proxy*.

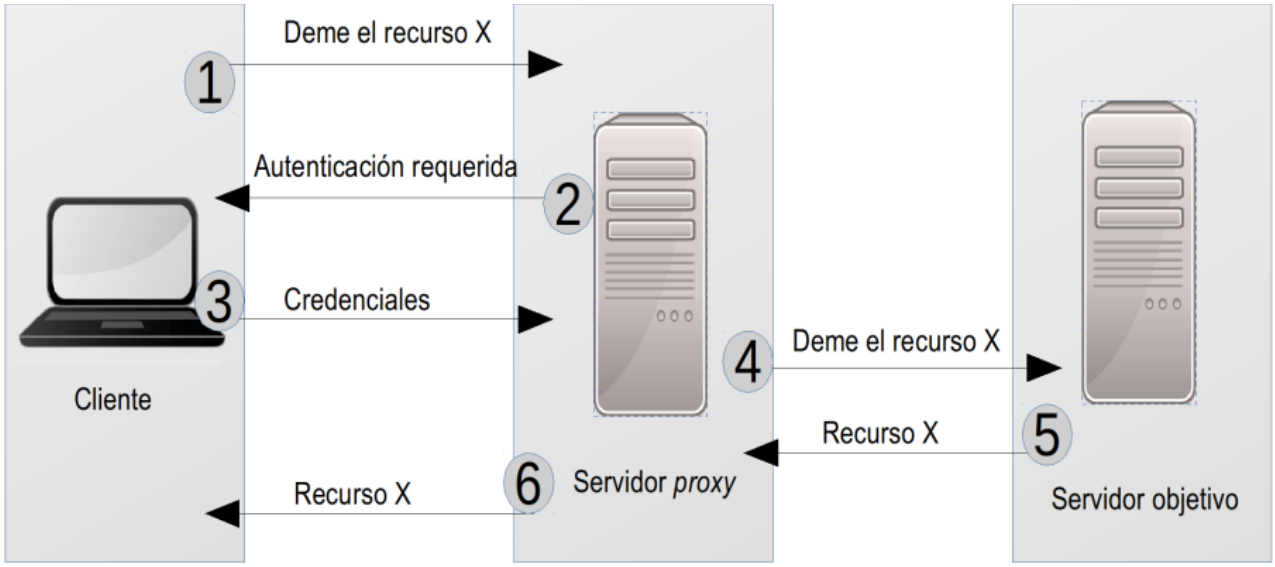



*Figura 2: Proceso de autenticación proxy.*

### *1.2.3 Esquemas de autenticación.*

El *framework* de autenticación HTTP es utilizado por diferentes esquemas de autenticación. Estos pueden variar en su seguridad y en su disponibilidad en el software del cliente o del servidor. Se muestra a continuación en la *Tabla 1* una breve descripción de los esquemas examinados en la presente investigación.

*Tabla 1: Descripción de los esquemas de autenticación estudiados.*

| Esquema de autenticación | Descripción |
|---|---|
| Basic | Este esquema de autenticación envía una cadena de caracteres codificada con Base64[4] que contiene las credenciales del cliente. Debido a que Base64 no es una forma de cifrado, debe considerarse igual a enviar los datos en texto claro (Gourley y Totty 2002). |

4 Base-64: Esquema de codificación de binario a texto que representa datos binarios como una cadena de caracteres.

| Esquema de autenticación | Descripción |
|---|---|
| Digest | Es un esquema de reto-respuesta que fue creado con el objetivo de remplazar la autenticación Basic. El servidor envía una cadena de datos aleatorios al cliente llamado *nonce* como un reto. El cliente responde con el resultado de aplicar una función resumen al nombre de usuario, la contraseña y el *nonce* entre información adicional. La complejidad de este intercambio y la aplicación de una función resumen hace más complicado el robo de las credenciales (Gourley y Totty 2002). |
| NTLM | NT LAN Manager (NTLM) es una suite que incluye las versiones NTLMv1, NTLMv2 y NTLM2 *Session*. Estos esquemas funcionan de una manera reto-respuesta que es más segura que Digest. Utilizan las credenciales de Windows para transformar los datos del reto en lugar de codificar las credenciales. Estos esquemas requieren múltiples intercambios entre el cliente y el servidor (Glass 2006). |
| *Kerberos* | Desarrollado por el MIT (*Massachusetts Institute of Technology*). Posee dos componentes principales: un *ticket*, usado para la autenticación del usuario y para la seguridad de los datos; y un autenticador que es usado para verificar que el usuario es el mismo que al que el *ticket* fue otorgado. Cuando un usuario entra en un sistema, el sistema se conecta al servidor *Kerberos* para extraer una llave de sesión para ser usada entre el usuario y el *ticket*. Esta llave es cifrada teniendo como llave la contraseña del usuario. Entonces si el usuario proporciona la contraseña correcta el sistema es capaz de descifrar la llave de sesión. Después de realizar este proceso la contraseña es eliminada de la memoria para evitar ser comprometida (Duncan 2001). |
| *Negotiate* | Es un seudo-mecanismo utilizado para negociar entre el esquema NTLM y *Kerberos* dependiendo de la disponibilidad de estos. *Kerberos* es utilizado si está disponible, en caso contrario es utilizado NTLM (Erikre 2017). |
| *Bearer* | Permite a los clientes acceder a recursos protegidos mediante la obtención |

| Esquema de autenticación | Descripción |
|---|---|
| | de un *token*, el cual es una cadena que representa un acceso autorizado emitido al cliente en lugar de usar las credenciales del usuario directamente (Hardt y Jones 2012). |
| HOBA | Esquema de autenticación que puede ser usado en autenticaciones basadas en el lenguaje de programación JavaScript. El principal objetivo de este esquema es ofrecer un protocolo de autenticación fácil de implementar y que no está basado en contraseñas. El uso de HOBA puede reducir o eliminar contraseñas en base de datos con beneficios de seguridad significantes (Farrell, Thomas y Hoffman 2015). |

Con el objetivo de seleccionar los esquemas de autenticación a los cuales se les implementará soporte en la solución a desarrollar se realizó lo siguiente:

- Estudio de los esquemas de autenticación que soportan las aplicaciones más usadas en el sistema operativo Android que permiten la autenticación con servidor *proxy* web.
- Entrevistas con especialistas acerca de los esquemas de autenticación más usados en los servidores *proxy* de las instituciones cubanas.
- Comprobación del soporte del servidor *proxy Squid*, usado por cientos de proveedores de Internet en todo el mundo («*squid : Optimising Web Delivery*» 2013), para los esquemas de autenticación resultantes de las acciones anteriores.

Para la selección de las aplicaciones más usadas en el sistema operativo Android que permiten la autenticación con servidor *proxy* web se tuvo como referencia la cantidad de descargas en *Google Play Store*[5]. Luego se procedió a escoger los esquemas de autenticación soportados por estas soluciones. Las aplicaciones seleccionadas fueron ProxyDroid (de 1 millón a 5 millones de descargas), Drony (de 100 000 a 500 000 descargas) y SandroProxy (de 50 000 a 100 000 descargas), estas tecnologías también serán examinadas en el estudio de soluciones similares. ProxyDroid brinda soporte para los esquemas Basic,

5 *Google Play Store*: plataforma de distribución digital de aplicaciones móviles con sistema operativo Android desarrollada por Google.

NTLMv1 y NTLMv2, mientras que Drony y SandroProxy implementan los esquemas Basic, Digest y los pertenecientes a la suite NTLM.

Las entrevistas fueron realizadas al Ing. Yosel Lázaro Vera Gonzalez (especialista de migración en la UCI), al Mcs.Yoandy Pérez Villazón (director del Centro de Soluciones Libres) y al Ing. Gustavo Quesada Arévalo (especialista de migración en la UCI). Se les cuestionó sobre los esquemas de autenticación más usados en los servidores *proxy* desplegados en las instituciones cubanas (ver Anexo 1). Se obtuvieron los siguientes resultados:

- Ing. Yosel Lázaro Vera Gonzalez: NTLMv1 y NTLMv2.
- Mcs.Yoandy Pérez Villazón: Digest, NTLMv1 y NTLMv2.
- Ing. Gustavo Quesada Arévalo: suite NTLM (NTLMv1, NTLMv2 y NTLM2 *Session*).

Entonces realizando una referencia cruzada entre los esquemas soportados por las soluciones estudiadas y los resultados de las entrevistas realizadas se seleccionan Basic, Digest y los pertenecientes a la suite NTLM (NTLMv1, NTLMv2 y NTLM2 *Session*).

Por último, como paso final, se comprobó satisfactoriamente el soporte en el servidor *proxy Squid* a cada uno de los seleccionados. De esta manera se decide soportar los esquemas de autenticación Basic, Diget y los pertenecientes a la suite NTLM en la solución a desarrollar. Se detalla continuación con más profundidad el funcionamiento de estos.

***Basic:***

El esquema de autenticación Basic está definido en la RFC 7617, este transmite las credenciales del usuario codificadas en Base64. Este protocolo no es considerado un método seguro a menos que se use en unión con algún sistema de seguridad externo como TLS (*Transport Layer Security*) o SSL (*Secure Socket Layer*) puesto que el usuario y la contraseña son enviadas en texto claro por la red. Está basado en que el cliente necesita autenticarse él mismo con una identificación de usuario y una contraseña por cada *realm*[6], este valor es una cadena de caracteres que puede ser solamente comparada con otros *realms* en el mismo servidor. El servidor proporcionará la petición solamente si este puede validar la identificación del usuario y la contraseña para el *realm* aplicado al recurso requerido (Reschke 2015). Este esquema utiliza un proceso de cuatro etapas para autenticar a los usuarios.

6 *realm*: espacio de protección, reinado, referido al entorno desde donde se va a autenticar el usuario.

Primeramente, el cliente HTTP, que puede ser un navegador web, realiza una petición al servidor web; si el servidor observa que el recurso requerido necesita autenticación entonces responde que no está autorizado. Luego el cliente envía las credenciales codificadas en Base64 y por último si las credenciales son correctas el servidor responde que se ha establecido correctamente la conexión.

El protocolo de autenticación Basic es el más predominante en los esquemas de autenticación HTTP debido a su simplicidad. Para el envío de las credenciales al servidor objetivo combina el nombre de usuario el y la contraseña y los cifra utilizando el método antes mencionado que puede ser fácilmente descifrado. Este esquema es simple y muy conveniente, pero no es seguro (Gourley y Totty 2002).

***Digest:***

Digest es otro de los protocolos de autenticación HTTP, está definido en la RFC 7616. Este fue desarrollado como una compatible y más segura alternativa al esquema Basic. Utiliza el mismo proceso de cuatro etapas utilizado por Basic, su principal diferencia es que en lugar de enviar las credenciales envía el resultado de una función resumen de varios campos incluidos el usuario y la contraseña utilizando por defecto el algoritmo MD5[7] el cual puede ser cambiado por cualquier otro. En particular la autenticación Digest nunca envía contraseñas secretas por la red en texto claro, impide los ataques de hombre en el medio, opcionalmente puede proteger contra la alteración del contenido del mensaje y protege de otras formas de ataques (Gourley y Totty 2002).

Digest no es el protocolo más seguro posible comparado con los mecanismos de llave pública. Es también recomendable su uso con TLS o SSL. Sin embargo, es significativamente más fuerte que la autenticación Basic, puesto que fue diseñado para remplazarla. Además, en la RFC 2617 se recomienda que cualquier servicio que utilice Basic debe ser remplazado por Digest.

***NTLM:***

NTLM es una suite de protocolos de autenticación propietaria desarrollada por Microsoft y optimizado para plataforma Windows, incluye las versiones NTLMv1, NTLMv2 y NTLM2 *Session*; estos protocolos son considerados más seguros que Digest («*Chapter 4. HTTP authentication*» 2017).

Las variables con las que trabajan los protocolos pertenecientes a NTLM se basan en los datos obtenidos durante el proceso de inicio de sesión interactivo y se componen de: un nombre de dominio, un nombre de usuario y una función resumen unidireccional de la contraseña del usuario (Pérez Vera y Díaz Orozco

7 MD5: Algoritmo de reducción criptográfico de 128 bits ampliamente usado.

2015).

La autenticación NTLM generalmente involucra a tres sistemas: un cliente, un servidor y un controlador de dominio que es donde la información relacionada con las contraseñas de los usuarios es almacenada y donde en ocasiones se realizan los cálculos en representación del servidor.

Los siguientes pasos representan un contorno de la autenticación NTLM («Microsoft NTLM (Windows)» 2017):

1. Un usuario accede a una computadora cliente y proporciona un nombre de dominio, nombre de usuario y contraseña. El cliente calcula una función resumen criptográfica y descarta la contraseña real.
2. El cliente envía el nombre de usuario al servidor (en texto plano).
3. El servidor genera un número aleatorio de 16 *bytes*, llamado desafío, y lo envía al cliente.
4. El cliente cifra este desafío con la función resumen de la contraseña y devuelve el resultado al servidor.
5. El servidor envía los siguientes tres elementos al controlador de dominio: nombre de usuario, desafío que se envió al cliente y respuesta recibida del cliente.
6. El controlador de dominio utiliza el nombre de usuario para recuperar el valor de la función resumen de la contraseña de la base de datos y la utiliza para cifrar el desafío.
7. El controlador de dominio compara el desafío cifrado que se calculó en el paso 6 con la respuesta dada por el cliente en el paso 4. Si ellos son idénticos la autenticación es correcta.

La siguiente figura muestra el proceso de autenticación de NTLM.

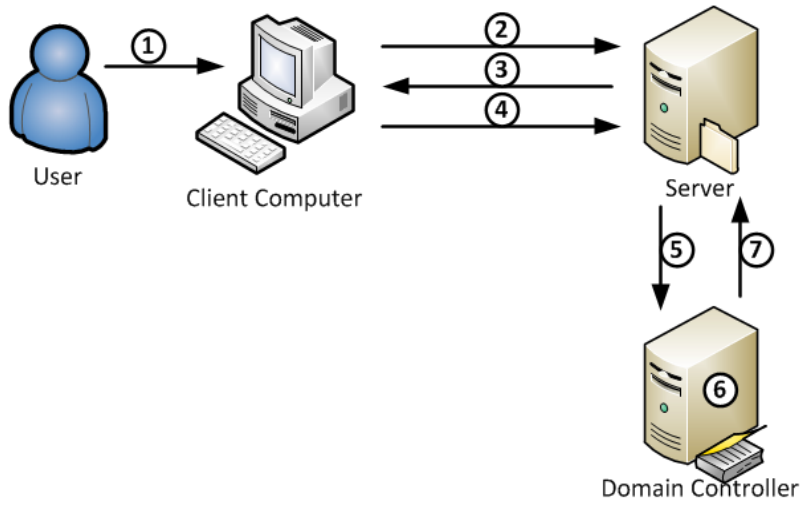



*Figura 3: Proceso de autenticación de NTLM*

Este protocolo es significativamente más costoso en términos de recursos computacionales y rendimiento que los esquemas Basic y Digest. Esto es debido principalmente a que una vez autenticado el usuario su identidad es asociada con la conexión por todo su intervalo de vida («*Chapter 4. HTTP authentication*» 2017).

## 1.3 Estudio de soluciones similares

En el presente epígrafe se realiza un análisis de las principales soluciones que permiten la autenticación con servidor *proxy* web, desarrolladas tanto a nivel nacional como internacional. Se tienen en cuenta las ventajas y desventajas del uso de cada una de ellas.

### *1.3.1 ProxyDroid*

Aplicación desarrollada para el sistema operativo Android que proporciona servicio de *proxy* local para el acceso a Internet. Posee soporte para los tipos de *proxy* HTTP, HTTPS[8], SOCKS[9]4 y SOCKS5; soporta además los esquemas de autenticación Basic, NTLMv1 y NTLMv2 y es de código abierto (Lv 2016). Posibilita la creación de varios perfiles de conexión para su uso en diferentes redes o con diferentes configuraciones, mantiene registro de las trazas generadas en el dispositivo y permite la aplicación de reglas de cortafuegos con el objetivo de bloquear peticiones no deseadas. Esta solución presenta como problema que no implementa los esquemas de autenticación Digest y NTLM *Session.* Además, necesita permisos de administrador para ejecutarse correctamente lo que limita su uso a grandes cantidades de usuarios.

### *1.3.2 SandroProxy*

Aplicación desarrollada para el sistema operativo Android que tiene como objetivo permitir el acceso a Internet mediante un *proxy* con autenticación. Soporta los esquemas Basic, Digest y los pertenecientes a la suite NTLM y es de código abierto («SandroProxy» 2016). Además, mantiene registro de las trazas generadas en el dispositivo y posibilita el uso de reglas de cortafuegos. Para que la aplicación se ejecute correctamente es necesario permisos de administrador en el dispositivo.

8 HTTPS: protocolo de comunicación segura sobre HTTP.

9 SOCKS: protocolo de comunicación cliente-servidor.

### *1.3.3 Drony*

Tecnología para el sistema operativo Android que brinda servicio de *proxy* local que puede operar con autenticación *proxy*. Presenta soporte para los tipos de *proxy* HTTP, HTTPS, SOCKS4a, SOCKS5 y además implementa los esquemas Basic, Digest y los pertenecientes a la suite NTLM («Drony» 2017). Permite llevar registro de las trazas generadas durante la conexión y aplicar reglas de cortafuegos. No posee traducción al idioma español lo que dificulta su uso por los usuarios de habla hispana. Además, su código fuente no está disponible, esto imposibilita su reusabilidad y adiciona un alto grado de desconfianza de cara al usuario final puesto que no es posible conocer cómo se manejan los datos transmitidos a través de esta aplicación.

### *1.3.4 UCIntlm*

Aplicación móvil desarrollada para el sistema operativo Android desarrollada en la UCI que permite usar Internet en las instituciones autenticando su dispositivo en el servidor *proxy* de la escuela o compañía. Soporta los protocolos NTLMv1 y NTLMv2, es de código abierto y presenta una adecuada documentación puesto que fue el resultado de una tesis de pregrado realizada en la universidad (Pérez Vera y Díaz Orozco 2015). Como problema presenta que no soporta los esquemas Basic, Digest y NTLM *Session,* limitando así su número de usuarios.

### *1.3.5 Conclusiones sobre el estudio de las soluciones.*

Como resultado del análisis anterior y a modo resumen se muestra la *Tabla 2* donde las anteriores aplicaciones son comparadas sobre la base de las características de: soporte para los protocolos Basic, Digest y los pertenecientes a la suite NTLM, necesidad de poseer permisos de administrador y la disponibilidad de su código fuente.

*Tabla 2: Comparación entre las soluciones estudiadas.*

| Aplicaciones | Soporte Basic | Soporte Digest | Soporte NTLMv1 | Soporte NTLMv2 | Soporte NTLM2 *Session* | No necesidad de permisos de administrador | Código abierto |
|---|---|---|---|---|---|---|---|

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| ProxyDroid | X | | X | X | | | X |
| SandroProxy | X | X | X | X | X | | X |
| Drony | X | X | X | X | X | X | |
| UCIntlm | | | X | X | | X | X |

Es posible concluir que no existe una solución de código abierto y que no requiera permisos de administrador que permita la autenticación con servidor *proxy* web en el sistema operativo NovaDroid mediante los protocolos Basic, Digest, NTLMv1, NTLMv2 y NTLM2 *Session.*

Se extraen de las aplicaciones estudiadas funcionalidades claves para el desarrollo de la solución a desarrollar como son: gestión de perfiles de conexión, soporte para cortafuegos y visualización de las trazas generadas. Se decide además la reutilización de la aplicación UCIntlm teniendo en cuenta principalmente que es un producto de la universidad y se puede contribuir a su mejoramiento, además su código fuente está disponible y posee documentación sobre este lo que permite su rápido aprendizaje y adaptación a las necesidades de la solución a desarrollar.

## 1.4 Tecnologías

La selección de las tecnologías constituye un factor clave en el desarrollo de una solución informática. Esta elección está guiada principalmente por los requerimientos que presenta el software a desarrollar. En la presente sección se definen las tecnologías escogidas.

### *1.4.1 Lenguaje de programación*

***Java v1.8.0:***

Lenguaje de programación de propósito general, concurrente y orientado a objeto. Permite que el programa se escriba una sola vez y se ejecute en diferentes dispositivos, es decir, el código que se ejecuta en una plataforma no tiene que volverse a compilar para ejecutarse en otra diferente. Su sintaxis se deriva principalmente de los lenguajes C y C++ mejorando principalmente la facilidad de uso y con la diferencia de ser orientado a objetos por completo (Evans y Flanagan 2014).

Se decide la utilización de Java teniendo en cuenta principalmente a que posee un poderoso conjunto de herramientas de desarrollo de software (Android SDK), desarrolladas por Google que permite la

implementación de soluciones de calidad para Android pudiendo ser ejecutadas en dispositivos con diferentes configuraciones de hardware. Además, el equipo de desarrollo cuenta con vasta experiencia en su uso.

### *1.4.2 Lenguaje de modelado*

***UML v2.0:***

UML (Lenguaje de Modelado Unificado, por sus siglas en inglés) es un lenguaje de modelado estandarizado que consiste en un integrado conjunto de diagramas, fue creado con el objetivo de ayudar a desarrolladores de software y de sistema en la especificación, visualización, construcción y documentación de los artefactos de software, así como también para el modelado de negocio («*What is Unified Modeling Language?*» 2017).

Para modelar los artefactos generados se utilizará UML en su versión 2.0 pues el equipo de desarrollo posee amplia experiencia en el uso de este lenguaje.

### *1.4.3 Herramienta de modelado*

***Visual Paradigm v8.0:***

Herramienta de software diseñada para modelar los sistemas de información empresarial y gestionar el proceso de desarrollo de software. Soporta lenguajes de modelado y estándares claves como UML, SysML (*System Modeling Language*), BPMN (*Bussines Process Model and Notation*), entre otros. Ofrece además un conjunto completo de herramientas de software para tareas como la captura de requisitos, análisis de procesos, diseño de sistemas y diseño de base de datos («*Visual Paradigm Frequently Asked Questions*» 2017).

Se selecciona esta herramienta debido al conjunto de funcionalidades útiles que ofrece para llevar a cabo el proceso de desarrollo de software y a su adecuada integración con UML, el lenguaje de modelado seleccionado.

### *1.4.4 Entorno de Desarrollo Integrado (IDE)*

***Android Studio v3.0.1:***

IDE oficial para el desarrollo de aplicaciones Android, desarrollado por Google publicado de forma gratuita bajo la licencia de código abierto Apache 2.0. Está basado en el IDE Intellj IDEA de la compañía JetBrains

y está disponible para las plataformas Windows, MacOS y GNU/Linux («*Android Studio Release Notes | Android Studio*» 2017).

La selección de este IDE se basa principalmente en la experiencia del equipo de desarrollo en su uso, la cual es amplia, y además en que es el recomendado por Google para el desarrollo de aplicaciones Android puesto que está implementado con tal propósito.

### *1.4.5 Herramienta de construcción*

***Gradle v4.1:***

Herramienta de construcción enfocada en la construcción automática y el soporte para el desarrollo en diferentes lenguajes de programación. Proporciona un modelo flexible que puede soportar todo el ciclo de desarrollo, desde compilar y empaquetar hasta publicar sitios web. Soporta la construcción automática para diferentes lenguajes como Java, Scala, Android, C/C++ y Groovy; además se integra estrechamente con varios IDEs como Eclipse, Intellj y Android Studio. Ofrece también funcionalidades como la gestión de dependencias, facilitando su resolución a través de repositorios centrales publicados en Internet («gradle» 2017).

Esta selección está condicionada principalmente a que la herramienta es utilizada por defecto en Android Studio para la construcción de proyectos y la gestión de dependencias, ahorrando así tiempo de configuración.

### *1.4.6 Agente HTTP*

***Apache HttpClient v4.4.1:***

Librería de código abierto desarrollada por Apache bajo la licencia Apache 2.0. Consta de un agente HTTP que utiliza la versión 1.1 de este protocolo, está basada en el módulo HttpCore desarrollado por la misma compañía. Proporciona componentes reutilizables para autenticación del lado del cliente, gestión de estados y de la conexión HTTP («Apache HttpComponents – Apache HttpComponents» 2017).

Luego de analizar varias librerías similares como OkHttp, HttpUrlConnection y Volley, se decide utilizar esta librería debido principalmente a que posee soporte para la autenticación HTTP mediante los protocolos Basic, Digest, NTLMv1, NTLMv2, NTLM2 *Session*, *Kerberos y Spengo.* Esto permite la implementación de los esquemas de autenticación sin excesivo esfuerzo en el desarrollo que puede

conllevar a errores. Además, HttpClient posee una amplia y robusta comunidad y cuenta con una rápida curva de aprendizaje.

### *1.4.7 Sistema de control de versiones*

***Git v2.7.4:***

Git es un sistema de control de versiones diseñado para rastrear cambios en un conjunto de ficheros y para realizar trabajo en paralelo sobre esos ficheros entre varias personas. Es usado principalmente para la gestión de código fuente en el desarrollo de software, pero puede ser utilizado para mantener un registro de cambios sobre cualquier conjunto de ficheros. Fue creado por Linus Torvalds y otros desarrolladores en 2005 para el desarrollo del núcleo de Linux. Git es software libre distribuido bajo los términos de la Licencia Pública General GNU versión 2 (Loeliger y McCullough 2012).

Se decide la utilización de Git teniendo en cuenta la necesidad de mantener un registro sobre los cambios realizados al proyecto y además debido a su excelente integración con Android Studio.

### *1.4.8 Gestor de base de datos*

***Realm v3.5.0:***

Sistema gestor de base de datos de código abierto desarrollado inicialmente para las plataformas Android e iOS. Propone un modelo de bases de datos orientada a objetos que permite la realización de consultas de manera sencilla. Es reactivo, concurrente, ligero y permite el trabajo con objetos nativos («*Realm Platform Documentation*» 2017).

La selección de Realm en lugar de SQLite (sistema gestor de base de datos incluido por defecto en el SDK de Android) está condicionada a que posee una rápida curva de aprendizaje debido a su gran facilidad de uso. Además, debido a su arquitectura y manera de realizar consultas a la base de datos permite ahorrar grandes cantidades de RAM (*Random Access Memory*) en el dispositivo, lo que conlleva al ahorro de batería, aspectos clave a tener en cuenta en el desarrollo de aplicaciones móviles. Según comparaciones realizadas en el centro CESOL (Centro de Soluciones Libres) perteneciente a la universidad, es más eficiente que SQLite en términos de tiempo de ejecución de consultas («Realm, el competidor de SQLite para los dispositivos móviles. | HumanOS» 2016).

## 1.5 Metodología de desarrollo

En ingeniería de software una metodología de desarrollo constituye un conjunto de métodos, procedimientos, técnicas, herramientas y soportes documentales utilizados para estructurar, planificar y controlar el proceso de desarrollo de sistemas de información (Pinzón y Guevara Bolaños 2006; «Selecting a Development Approach» 2005).

Para la adecuada implementación de la solución propuesta es necesario la selección de una metodología que guíe el ciclo de vida del proyecto para asegurar un producto de calidad. Se selecciona en consecuencia la metodología AUP-UCI teniendo en cuenta que es la metodología adaptada al ciclo de vida de los proyectos productivos de la universidad, es ampliamente usada en el área y es extremadamente flexible al proceso de desarrollo de software.

AUP-UCI constituye una variante de AUP (Proceso Unificado Ágil, por sus siglas en inglés) surge con el objetivo de ser una metodología que se adapte al ciclo de vida definido por la actividad productiva en la universidad. Se elaboró teniendo en cuenta el Modelo CMMI-DEV v1.3 que constituye una guía para aplicar las mejores prácticas en una entidad desarrolladora, estas prácticas se centran en el desarrollo de productos y servicios de calidad (Rodríguez Sánchez 2014).

Esta metodología propone tres fases para el desarrollo del software: Inicio, Ejecución y Cierre, siete disciplinas: Modelado de negocio, Requisitos, Análisis y diseño, Implementación, Pruebas internas, Pruebas de liberación y Pruebas de aceptación. Once roles: Jefe de proyecto, Planificador, Analista, Arquitecto de información, Desarrollador, Administrador de la configuración, Cliente o proveedor de requisitos, Administrador de calidad, Probador, Arquitecto de software y Administrador de bases de datos. (Rodríguez Sánchez 2014).

AUP-UCI propone además cuatro escenarios a utilizar para modelar el sistema en los proyectos. Se selecciona el escenario número cuatro, el cual establece que proyectos que no modelen negocio solo pueden modelar el sistema con historias de usuario. Para esta elección se tuvo en cuenta principalmente que no se realiza modelado del negocio puesto que este no está bien definido, y que la solución a desarrollar no consta de un proyecto muy extenso lo que permite que las historias de usuario no posean demasiada información y sean lo más concretas posible.

## 1.6 Conclusiones parciales

Luego del análisis realizado en el presente capítulo es posible afirmar que el estudio del *framework* de autenticación HTTP, la autenticación *proxy* y los esquemas de autenticación permitieron la adquisición del conocimiento teórico necesario para el posterior desarrollo de la solución. Además, el estudio de las soluciones existentes arrojó como resultado que las aplicaciones estudiadas no brindan una solución completa al problema a resolver, debido principalmente a la disponibilidad del código fuente, la necesidad de permisos de administrador y al incompleto soporte para los esquemas de autenticación Basic, Digest, NTLMv1, NTLMv2 y NTLM2 *Session*. Por último, es posible afirmar que el análisis realizado sobre las diferentes tecnologías, incluyendo la selección de la metodología de desarrollo AUP-UCI permitió la creación del entorno de trabajo para el desarrollo de la solución.

# Capítulo 2. Análisis y diseño

En el presente capítulo se exponen las principales características de la solución a desarrollar. Como parte de la disciplina de Requisitos definida por la metodología de desarrollo AUP-UCI se realiza la captura de los requisitos funcionales y no funcionales, y se describen las historias de usuarios correspondientes a estos. Por último, como parte de la disciplina Análisis y diseño se presenta la arquitectura de la solución, el diagrama de paquetes, los patrones de diseño utilizados y un fragmento del diagrama de clases del diseño.

## 2.1 Propuesta de solución

Se propone como solución una aplicación para dispositivos móviles con sistema operativo NovaDroid que brinde servicio de *proxy* local. Este recibirá las peticiones realizadas por las aplicaciones del dispositivo, las redireccionará hacia el servidor *proxy* web de la institución y realizará la autenticación con este. Deberá soportar los esquemas de autenticación Basic, Digest, NTLMv1, NTLMv2 y NTLM2 *Session*. Además, permitirá la creación de perfiles de conexión con el objetivo de posibilitar su uso en diferentes redes por el mismo usuario sin realizar cambios en la configuración de la aplicación cada vez que se cambie de red. Estos perfiles contarán con la configuración necesaria para conectarse al servidor *proxy* de la red en la cual se está operando. Adicionalmente deberá brindar funcionalidades de cortafuegos en orden de bloquear peticiones no deseadas por las aplicaciones y además deberá mantener un registro de las trazas de navegación generadas para que el usuario tenga conocimiento de las peticiones que se realizan en el dispositivo. Para el desarrollo de la solución se tiene como base la aplicación UCIntlm, reutilizando parte de su código fuente, específicamente en lo referente al servicio de *proxy* local.
Con el objetivo de lograr una mayor comprensión de la solución propuesta se muestra en la siguiente figura su funcionamiento básico.

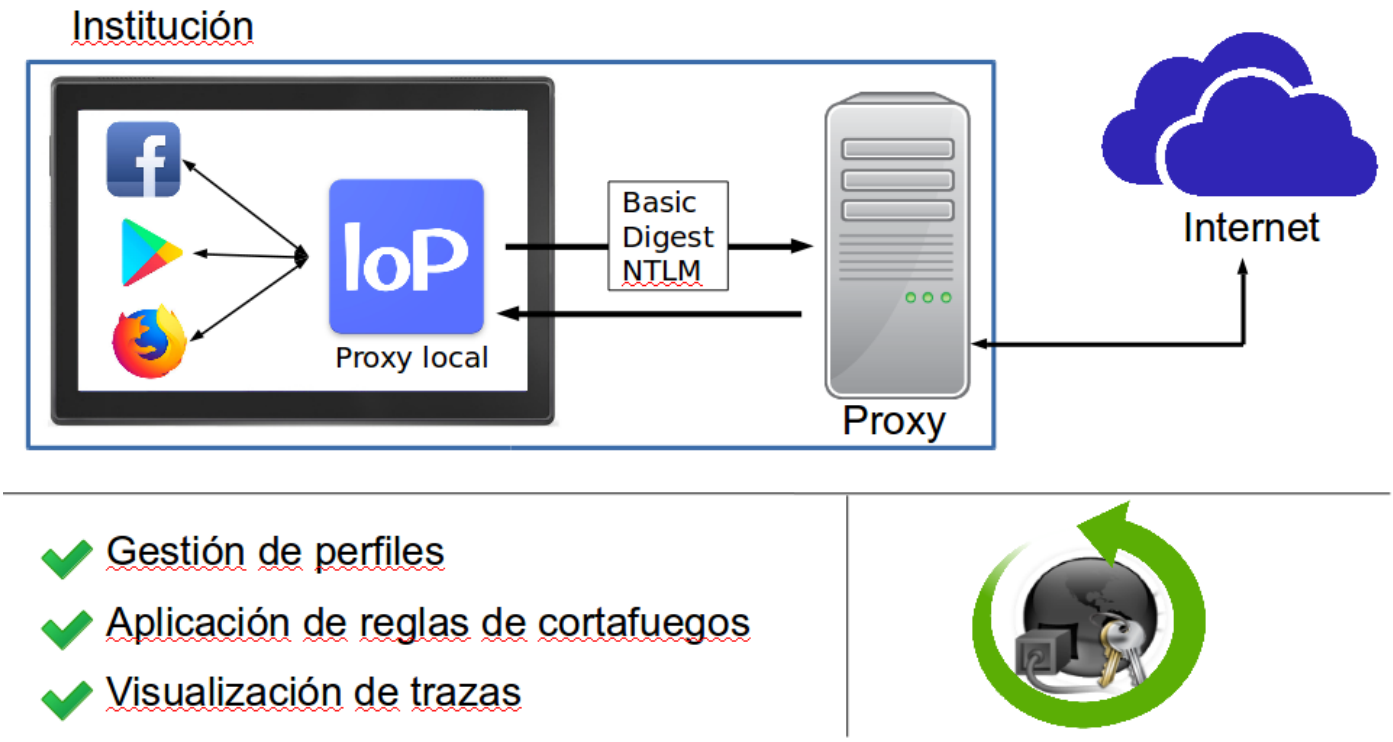



*Figura 4: Propuesta de solución*

## 2.2 Captura de requisitos

Según el estándar 1233 de la IEEE[10]: Guía para el desarrollo de Especificaciones de Requerimientos de Sistemas, un requisito se define como:

- Condición o capacidad que necesita un usuario para resolver un problema o lograr un objetivo.
- Condición o capacidad que tiene que ser alcanzada o poseída por un sistema o componente de un sistema para satisfacer un contrato, estándar, u otro documento impuesto formalmente.

Es posible concluir que los requisitos de software son características y funcionalidades que debe poseer un sistema y están enfocados hacia lo que debe hacer el software. Además, pueden ser clasificados en

10 IEEE: del inglés *Institute of Electrical and Electronics Engineers*. Traducido al español Instituto de Ingeniería Eléctrica y Electrónica.

funcionales y no funcionales.

### *2.2.1 Técnicas para la captura de requisitos*

En el proceso de desarrollo de un sistema el equipo de desarrollo siempre se enfrenta al problema de la identificación de requisitos. La definición de estos es un proceso complejo, pues hay que identificar los requisitos que el sistema debe cumplir en orden de satisfacer las necesidades de los usuarios finales y clientes. Para realizar este proceso existen diferentes técnicas, su selección y resultados dependen en gran medida del equipo de desarrollo, como de los propios usuarios o clientes que participen en ellas. Se presentan a continuación las técnicas utilizadas para la identificación de los requisitos.

**Análisis de sistemas existentes:**

Mediante el análisis de sistemas existentes es posible estudiar aplicaciones similares a la que se necesita obtener. Una vez que se tiene la concepción del funcionamiento de un software similar en cuanto a funcionalidades y características es más sencillo identificar los requisitos del sistema que se necesita implementar.

Durante la investigación se realizó un estudio de aplicaciones similares a la solución a desarrollar, en las cuales se observaron los diseños de sus interfaces, las funcionalidades que ofrecen, el grado de dificultad a la hora de interactuar con la aplicación, entre otros rasgos importantes que contribuyeran a obtener un producto con la mejor calidad posible.

**Tormenta de ideas:**

Es una técnica de reuniones en grupo cuyo objetivo es que los participantes muestren sus ideas de forma libre. Consiste en la mera acumulación de ideas y/o información sin evaluar las mismas. El grupo de personas que participa en estas reuniones no debe ser muy numeroso (máximo 10 personas), una de ellas debe asumir el rol de moderador de la sesión, pero sin carácter de controlador (Escalona y Koch 2002).

Se efectuaron reuniones con el equipo de desarrollo de NovaDroid y a partir de un conjunto de ideas propuestas se identificaron importantes requisitos con los que debería cumplir la solución.

### *2.2.2 Requisitos funcionales (RF)*

Los requisitos funcionales son capacidades o condiciones que el sistema debe cumplir, cómo debe

comportarse en situaciones específicas. En algunos casos también pueden plantear explícitamente qué no debe hacer el sistema (Sommerville 2011)**.**

Con el objetivo de satisfacer las necesidades de los usuarios finales, se definieron los siguientes requisitos funcionales:

*Tabla 3: Descripción de los requisitos funcionales.*

| No. | Requisito Funcional | Descripción | Complejidad | Prioridad para el cliente |
|---|---|---|---|---|
| **1.** | Adicionar perfil de conexión. | La aplicación deberá permitir al usuario crear perfil de conexión, este contendrá la configuración necesaria para realizar las peticiones al servidor *proxy* de la entidad. | Media | Alta |
| **2.** | Mostrar perfil de conexión. | La aplicación deberá permitir mostrar un perfil de conexión previamente creado, debe mostrar todos sus datos. | Media | Alta |
| **3.** | Modificar perfil de conexión. | La aplicación deberá permitir modificar todos los datos de un perfil de conexión previamente creado. | Media | Alta |
| **4.** | Eliminar perfil de conexión. | La aplicación deberá permitir eliminar un perfil de conexión previamente creado. | Media | Alta |
| **5.** | Listar perfiles de conexión. | La aplicación deberá mostrar al usuario el listado de los perfiles de conexión creados. | Media | Alta |
| **6.** | Autenticar usuario. | La aplicación deberá permitir al usuario insertar nombre de usuario y contraseña con el objetivo de realizar la autenticación con el servidor *proxy*. | Media | Alta |
| **7.** | Iniciar servicio de *proxy* local. | La aplicación deberá permitir al usuario | Alta | Alta |

| | | | | |
|---|---|---|---|---|
| | | iniciar el servicio de *proxy* local, el cual se ejecutará en segundo plano, redireccionará las peticiones realizadas en el dispositivo y realizará la autenticación con el servidor *proxy* de la institución. | | |
| **8.** | Detener servicio de *proxy* local. | La aplicación deberá permitir al usuario detener el servicio de *proxy* local. | Alta | Alta |
| **9.** | Restablecer configuración. | Al iniciar la aplicación se deberá restablecer la última configuración con la que se ejecutó el servicio de *proxy* local. | Media | Media |
| **10.** | Adicionar regla de cortafuegos. | La aplicación deberá permitir al usuario crear una regla de cortafuegos. Tiene como objetivo bloquear la peticiones realizadas por una o todas las aplicaciones a una o varias URLs o direcciones de IPs. | Media | Media |
| **11.** | Mostrar regla de cortafuegos. | La aplicación deberá permitir al usuario visualizar una regla de cortafuegos previamente creada. | Media | Media |
| **12.** | Modificar regla de cortafuegos. | La aplicación deberá permitir al usuario modificar todos los datos de una regla de cortafuegos previamente creada. | Media | Media |
| **13.** | Eliminar regla de cortafuegos. | La aplicación deberá permitir al usuario eliminar una regla de cortafuegos previamente creada. | Media | Media |
| **14.** | Listar reglas de cortafuegos. | La aplicación deberá mostrar el listado de las reglas de cortafuegos creadas. | Media | Media |
| **15.** | Activar regla de cortafuegos | La aplicación deberá permitir al usuario | Media | Baja |

| | | | | |
|---|---|---|---|---|
| | | activar una regla de cortafuegos. Esto indicará al servicio de *proxy* local que tenga en cuenta dicha regla para bloquear peticiones. | | |
| **16.** | Desactivar regla de cortafuegos. | La aplicación deberá permitir al usuario desactivar una regla de cortafuegos. Esto indicará al servicio de *proxy* local que no tenga en cuenta dicha regla para bloquear peticiones. | Media | Baja |
| **17.** | Adicionar traza de navegación. | La aplicación deberá crear una traza de navegación por cada petición realizada en el dispositivo. | Alta | Media |
| **18.** | Listar trazas de navegación. | La aplicación deberá mostrar un listado con las trazas de navegación creadas por cada petición. | Media | Media |
| **19.** | Añadir traza de navegación como regla de cortafuegos. | La aplicación deberá permitir añadir una traza de navegación como regla de cortafuegos. | Media | Media |
| **20.** | Buscar trazas de navegación. | La aplicación deberá permitir al usuario buscar trazas de navegación a partir de un criterio de búsqueda insertado por este. | Media | Media |
| **21.** | Ordenar trazas de navegación | La aplicación deberá permitir al usuario ordenar el listado de las trazas de navegación atendiendo al criterio de cantidad *bytes* consumidos por la petición. | Media | Media |

### *2.2.3 Requisitos no funcionales (RNF)*

Los requisitos no funcionales son aquellos que no se refieren directamente a las funciones específicas que proporciona el sistema, sino a las propiedades de este como fiabilidad, tiempo de respuesta y la capacidad de almacenamiento. Incluyen además restricciones de tiempo sobre el proceso de desarrollo y estándares (Sommerville 2011). A continuación, se definen los requisitos no funcionales que debe cumplir la aplicación basándose en los atributos de calidad establecidos por el documento "Especificación_de_requisitos_de_software" que propone la metodología AUP-UCI.

**Requisitos no funcionales de Seguridad:**

RNF 1: Almacenar las contraseñas de los usuarios cifradas en la base de datos.

RNF 2: Enviar las credenciales al servidor *proxy* cifradas en consecuencia al esquema de autenticación que se utilice.

RNF 3: No modificar el contenido de las peticiones realizadas en el dispositivo.

**Requisitos no funcionales de Portabilidad**:

RNF 4: Funcionar correctamente en todas las versiones del sistema operativo NovaDroid.

RNF 5: Funcionar correctamente en todos los dispositivos con memoria RAM mayor o igual a 512 MB.

**Requisitos no funcionales de Usabilidad:**

RNF 6: Soportar los idiomas: inglés y español.

RNF 7: Poseer interfaz de usuario basada en Material Design.

**Requisitos no funcionales de Funcionalidad:**

RNF 8: Soportar los esquemas de autenticación Basic, Digest, NTLMv1, NTLMv2 y NTLM2 *Session*.

## 2.3 Historias de usuario (HU)

Las historias de usuarios son parte del desarrollo ágil de software que ayuda a enfocarse en hablar de los requerimientos en lugar de escribir acerca de ellos. Incluyen una o dos sentencias y, más importante, una serie de conversaciones acerca de la funcionalidad deseada. Las historias de usuario son cortas descripciones de una funcionalidad desde la perspectiva de la persona que la desea, usualmente un usuario o cliente (Cohn 2018).

En correspondencia con la selección del escenario número cuatro de la metodología empleada se procede

a modelar el sistema con historias de usuario, donde se define una por cada requisito funcional, a excepción de los requisitos 1, 2, 3, 4; 10, 11, 12, 13 y 15 y 16 los cuales se agruparon en las historias de usuario “Gestionar perfil de conexión”, “Gestionar regla de cortafuegos” y “Activar o desactivar regla de cortafuegos” respectivamente, lográndose un total de 14 HU. Se muestran a continuación las HU “Autenticar usuario”, “Gestionar perfil de conexión” y “Iniciar servicio de *proxy* local” por ser las que describen las funcionalidades que poseen mayor prioridad para el cliente. El resto de las HU se pueden encontrar del Anexo 2 al Anexo 12.

*Tabla 4: Historia de usuario “Autenticar usuario”.*

| **HU Autenticar usuario** | |
|---|---|
| **Número:** 1 | **Nombre del requisito:** Autenticar usuario |
| **Programador:** Daniel A. Rodríguez Caballero | **Iteración Asignada:** 1 |
| **Prioridad:** Alta | **Tiempo Estimado:** 10 días |
| **Riesgo en Desarrollo:** N/A | **Tiempo Real:** 12 días |
| **Descripción:**<br>Se listan a continuación las funcionalidades requeridas:<br>• Permitir que el usuario inserte sus credenciales (nombre de usuario y contraseña) para realizar el proceso de autenticación con el servidor *proxy* de la institución.<br>• Posibilitar que el usuario decida si se debe recordar su contraseña para futuras ejecuciones.<br>• Mientras se introduce el nombre de usuario mostrar una lista emergente de los usuarios previamente insertados que empiecen con los caracteres escritos hasta el momento.<br>• Si el usuario selecciona uno de los nombres de usuarios mostrados en la lista emergente se debe auto-completar el campo de usuario con el seleccionado y además debe auto-completarse la contraseña en caso de que el usuario haya decidido recordarla.<br>• Permitir que el usuario observe la contraseña insertada hasta el momento.<br>La funcionalidad tiene lugar en la vista inicial de la aplicación. Debe haber dos campos de texto, uno para el nombre de usuario y otro para la contraseña. Además, debe haber un botón con una imagen | |

sugerente para que el usuario pueda observar la contraseña insertada hasta el momento y un *checkbox* para que decida si recordar o no su contraseña. Además, en el momento de iniciar el servicio de *proxy* local descrito por la HU número 3 debe realizarse el proceso de autenticación con el servidor *proxy* web de la institución para poder iniciar el servicio.

**Observaciones:**

**Prototipo elemental de interfaz gráfica de usuario:**

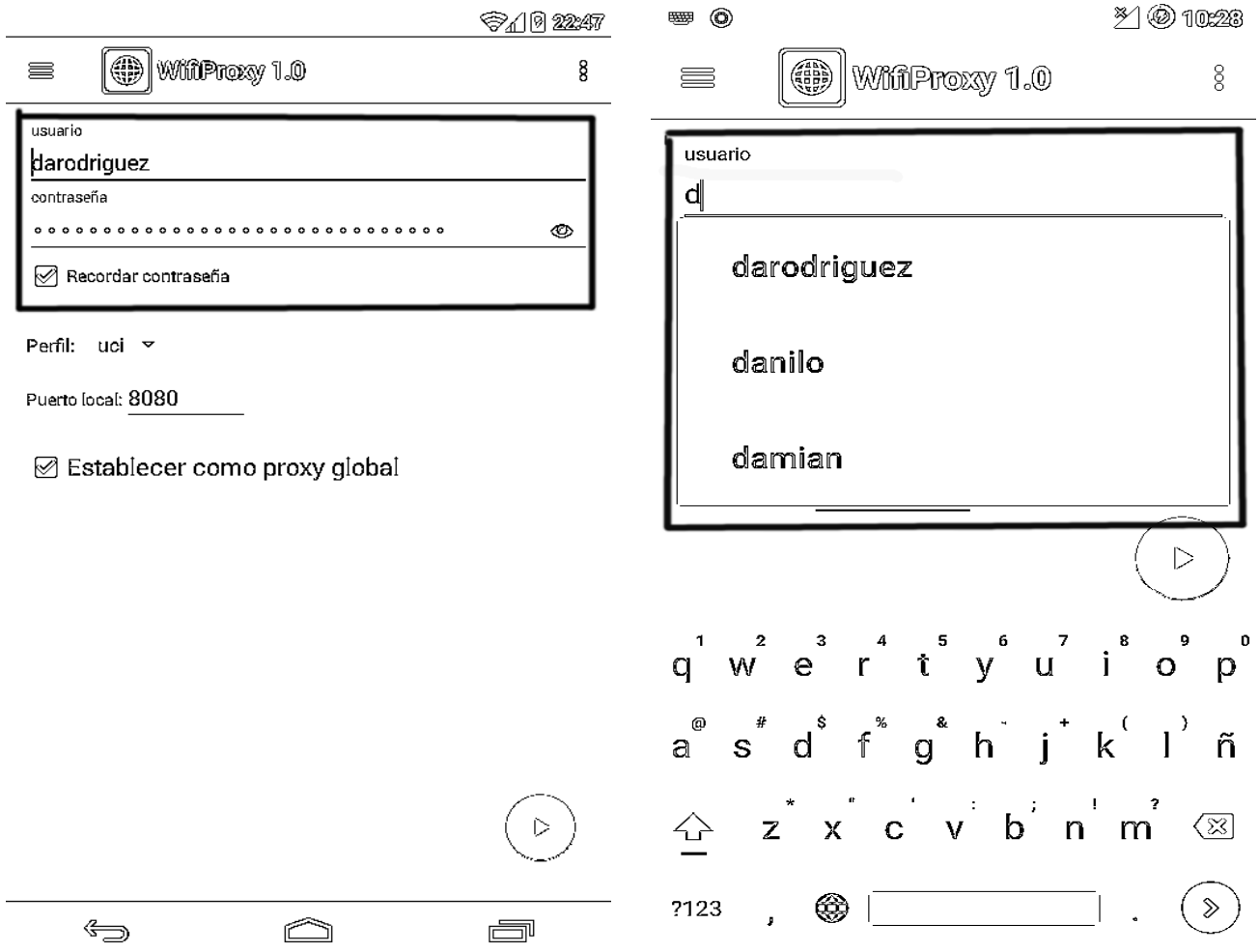


*Tabla 5: Historia de usuario "Gestionar perfil de conexión".*

| **HU Gestionar perfil de conexión** | |
|---|---|
| **Número:** 2 | **Nombre del requisito:** Adicionar perfil de conexión, Mostrar perfil de conexión, Modificar perfil de conexión, Eliminar perfil de conexión. |
| **Programador:** Daniel A. Rodríguez Caballero | **Iteración Asignada:** 1 |
| **Prioridad:** Alta | **Tiempo Estimado:** 8 días |
| **Riesgo en Desarrollo:** N/A | **Tiempo Real:** 8 días |

**Descripción:**

Se debe permitir que el usuario adicione, modifique, visualice y elimine los perfiles de conexión. Estos deben poseer los siguientes datos:

1. Nombre.
2. Dirección URL o IP del servidor *proxy*.
3. Puerto de entrada del servidor *proxy*.
4. Reglas para omitir el uso del *proxy* (opcional).

**Adicionar perfil de conexión:**

Se debe mostrar un campo de texto por cada dato que posee un perfil de conexión en orden de que el usuario los complete según sus necesidades, y además un botón para completar la acción de adicionar.

Se deben realizar las siguientes validaciones:

1. El campo del nombre del perfil no debe estar vacío.
2. El campo de la dirección URL o IP del servidor *proxy* no debe estar vacío y debe ser una dirección IP o URL.
3. El campo del puerto de entrada del servidor *proxy* no debe estar vacío y debe ser un número entero en el rango de 0 a 65535 (rango de puertos disponibles).

Se podrá acceder a la funcionalidad desde el listado de perfiles de conexión definido por la HU "Listar perfiles de conexión" donde habrá un botón indicando la acción de crear un nuevo perfil.

**Mostrar perfil de conexión:**

Se debe permitir que el usuario visualice todos los datos de un perfil de conexión previamente creado. Debe mostrarse un campo de texto no editable por cada dato del perfil de conexión. Es necesario además mostrar un botón para la funcionalidad de eliminar el perfil y otro para modificarlo.

La funcionalidad será accesible desde el listado de perfiles de conexión, donde al seleccionar uno de ellos, se ejecutará.

**Modificar perfil de conexión:**

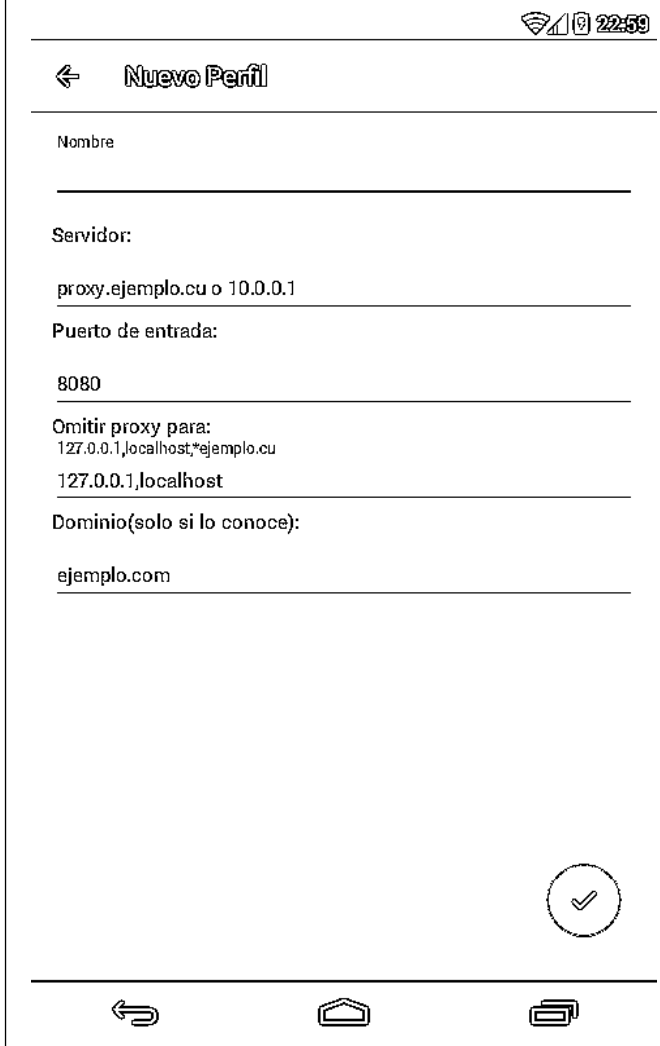
Se debe permitir que el usuario modifique un perfil de conexión previamente creado. Debe mostrarse un campo texto por cada dato que posee el perfil y completarlos con la información que ya posee, estos campos deben ser editables y deben realizarse las mismas validaciones que en el caso de "Adicionar perfil de conexión". Debe mostrarse además un botón para completar la acción de modificar.

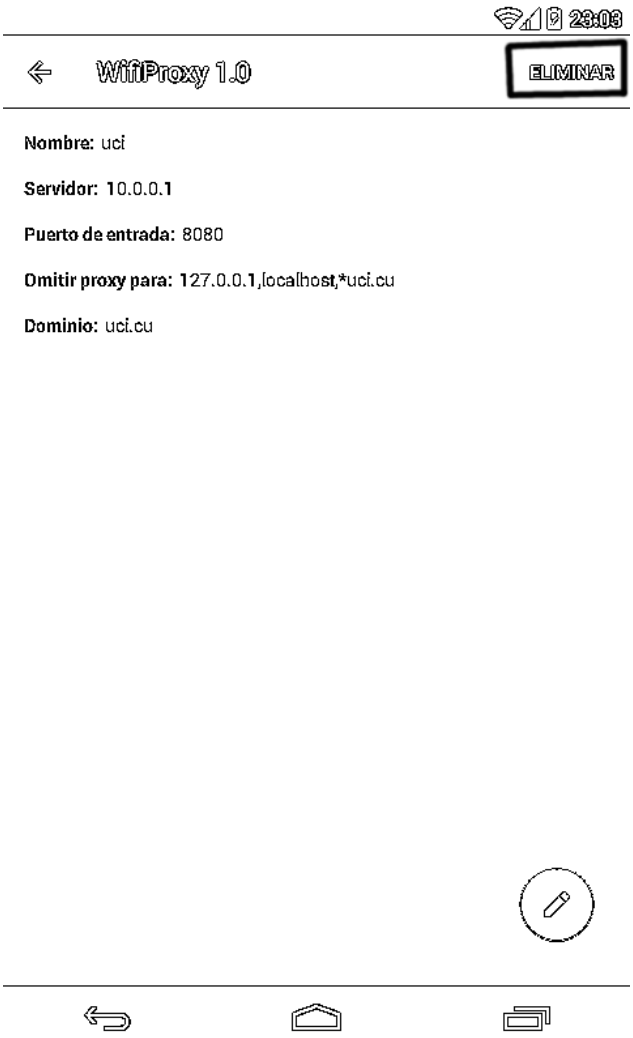
**Eliminar perfil de conexión:**

Se debe permitir que el usuario elimine un perfil de conexión previamente creado. La acción estará disponible desde un botón en la vista de "Mostrar perfil de conexión" donde al presionarlo se elimina por completo el perfil que se estaba visualizando.

**Observaciones:**

**Prototipo elemental de interfaz gráfica de usuario:**


22:59
Nuevo Perfil
Nombre
Servidor:
proxy.ejemplo.cu o 10.0.0.1
Puerto de entrada:
8080
Omitir proxy para:
127.0.0.1,localhost,*ejemplo.cu
127.0.0.1,localhost
Dominio(solo si lo conoce):
ejemplo.com


23:03
WifiProxy 1.0
ELIMINAR
Nombre: uci
Servidor: 10.0.0.1
Puerto de entrada: 8080
Omitir proxy para: 127.0.0.1,localhost,*uci.cu
Dominio: uci.cu

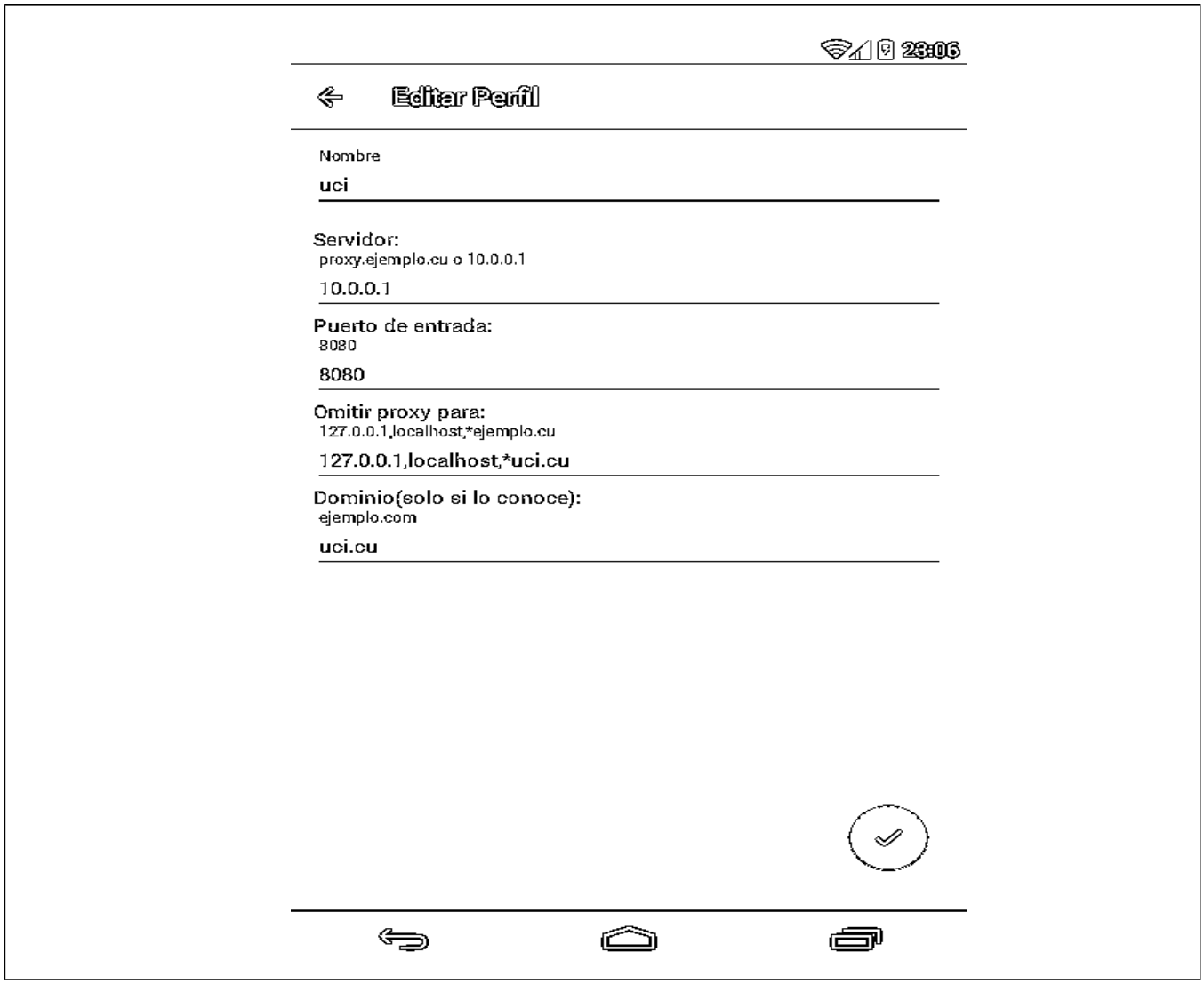



*Tabla 6: Historia de usuario "Iniciar servicio de proxy local".*

| **HU Iniciar servicio de *proxy* local** | | |
|---|---|---|
| **Número:** 3 | **Nombre del requisito:** Iniciar servicio de *proxy* local | |
| **Programador:** Daniel A. Rodríguez Caballero | | **Iteración Asignada:** 1 |

| | |
|---|---|
| **Prioridad:** Alta | **Tiempo Estimado:** 12 días |
| **Riesgo en Desarrollo:** N/A | **Tiempo Real:** 15 días |

**Descripción:**

Se debe permitir que el usuario inicie el servicio de *proxy* local. Este debe ejecutarse en segundo plano y debe redireccionar las peticiones recibidas hacia el servidor *proxy* de la institución y manejar el proceso de autenticación con este, soportando los esquemas de autenticación Basic, Digest, NTLMv1, NTLMv2 y NTLM2 *Session.*

Para su ejecución deben haberse insertado previamente las credenciales del usuario, así como también debe haberse seleccionado el perfil de conexión a utilizar y el puerto local por el cual debe recibir las peticiones el servicio.

La funcionalidad tiene lugar en la vista inicial de la aplicación junto con la funcionalidad de “Autenticar usuario”. Debe mostrarse una lista emergente con los perfiles de conexión creados hasta el momento en orden de que el usuario seleccione el que desea utilizar con el servicio. Además, debe mostrarse un campo de texto que permita la inserción del puerto por el cual se recibirán las peticiones y un botón que indique la acción de iniciar el servicio.

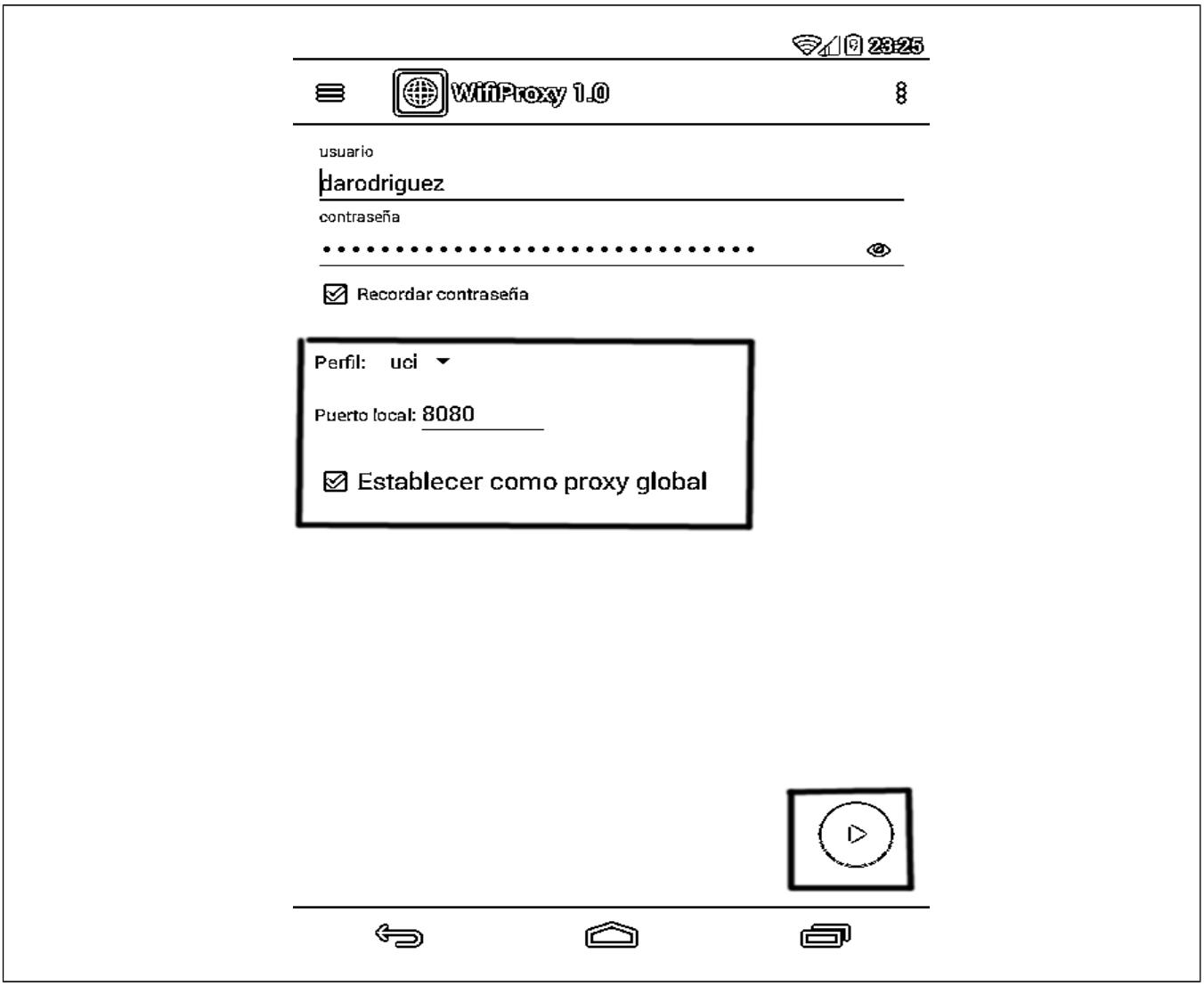
Deben realizarse además las siguientes validaciones:

- Validar que ni el usuario ni la contraseña sean vacíos.
- Validar que se seleccionó un perfil de conexión.
- Validar que el campo del puerto local no esté vacío y sea un número entero en el rango de 1023 a 65535 (rango de puertos disponibles que no interfiere con los puertos del sistema).

Luego de haberse validado correctamente todos los datos necesarios se debe guardar la configuración con la cual se inició el servicio: perfil de conexión seleccionado y puerto local de escucha. Además, se deben almacenar en la base de datos el nombre de usuario y la contraseña cifrada, esta última si fue marcada la opción “Recordar contraseña”. Por último se debe remplazar el botón flotante que indica la acción de iniciar por un botón flotante que indique la acción de detener el servicio, esto relativo a la historia de usuario ”Detener servicio de *proxy* local”.

**Observaciones:**

**Prototipo elemental de interfaz gráfica de usuario:**




## 2.4 Diseño

El papel del diseño en el ciclo de vida de un software es facilitar la comprensión de su funcionamiento y proveer una representación o modelo del mismo con el propósito de definirlo con los suficientes detalles como para permitir su realización física. El modelo de diseño provee una representación arquitectónica del software que sirve de punto de partida para las tareas de implementación.

### *2.4.1 Descripción de la arquitectura*

El estándar IEEE define la arquitectura de software como la organización fundamental de un sistema encarnada en sus componentes, las relaciones entre ellos y el ambiente y los principios que orientan su diseño y evolución («ISO/IEC/IEEE 42010: *Defining* «*architecture*»» 2011). Según Roger Pressman: “En su forma más simple, la arquitectura del software es la estructura u organización de los componentes del programa, la manera en que estos interactúan y la estructura de datos que utilizan” (Pressman 2010).

Es decir, la arquitectura de software es una forma de representar sistemas mediante el uso de la abstracción, de forma que aporte el más alto nivel de comprensión de los mismos. Esta representación incluye los componentes fundamentales del software, su comportamiento y formas de interacción para satisfacer los requisitos del sistema.

Para el desarrollo de la solución propuesta en el presente trabajo de diploma se propone una arquitectura Modelo-Vista-Presentador (MVP) teniendo en cuenta principalmente su bien definida separación de conceptos, su adecuada adaptación al Android SDK y a su amplio uso en el desarrollo de aplicaciones Android (TinMegali 2016; Leiva 2014)

Se presenta a continuación una representación de esta arquitectura utilizando el diagrama de paquetes proporcionado por UML.

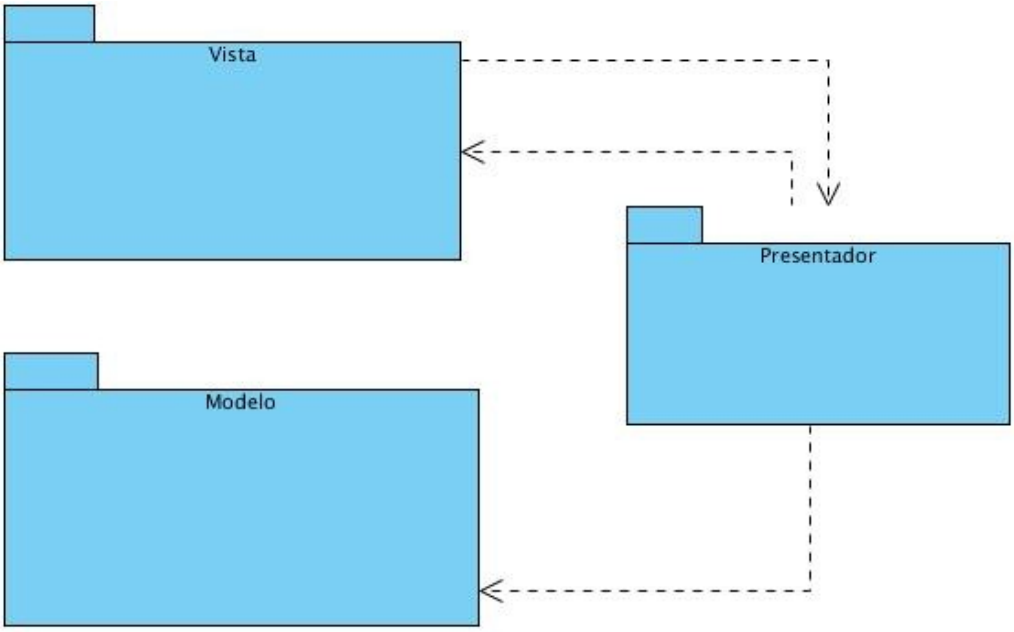



*Figura 5: Arquitectura MVP*

Como es posible observar en la Figura 5 el sistema se estructura en tres componentes lógicos que interactúan entre sí. El presentador es responsable de actuar como hombre en el medio entre la vista y el modelo, este recupera datos del modelo y los devuelve con un formato específico a la vista, en diferencia con la arquitectura Modelo-Vista-Controlador (MVC, *Model-View-Controller*), este también decide que sucede cuando el usuario interactúa con ella. La vista, posee referencia al presentador, lo único que debe hacer es llamar a un método del presentador cada vez que ocurre una acción en la interfaz. Por último, el modelo debe ser solamente un portal a la lógica de negocio, se puede ver como el proveedor de los datos que se desean mostrar en la vista (Leiva 2014).

Para el desarrollo del diseño arquitectónico se tuvo como base la arquitectura descrita en el ejemplo "*todo-mvp*" perteneciente al proyecto "*Android Architecture Blueprints*" desarrollado por Google. Este incluye una serie de ejemplos con diferentes arquitecturas con el objetivo de demostrar estrategias para ayudar a resolver problemas comunes como son clases extensas, inconsistente nombrado de esquemas, así como también desajustes o arquitecturas faltantes («*android-architecture*» 2018).

Se muestra a continuación el diagrama de paquetes de la solución.

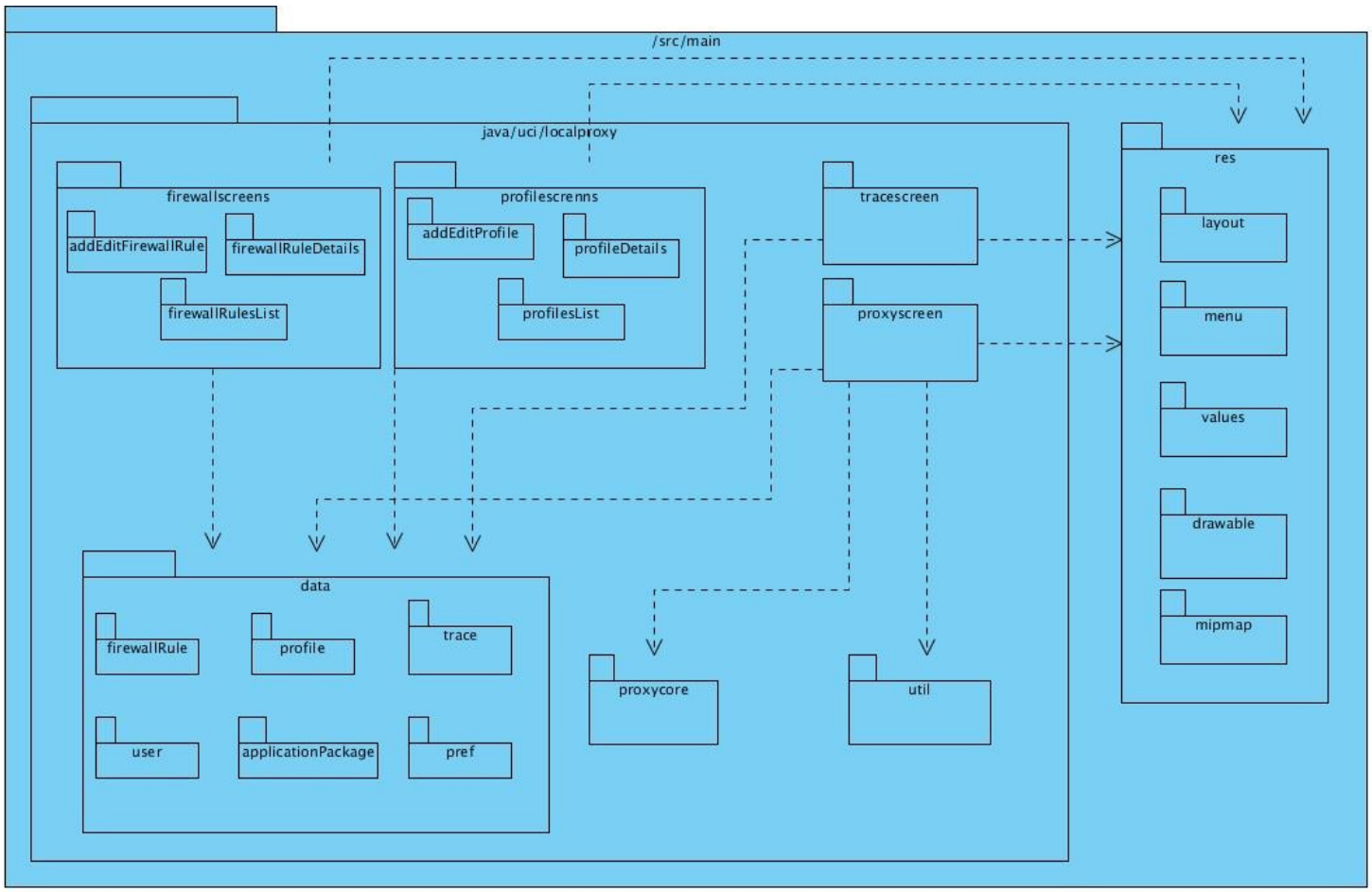



*Figura 6: Diagrama de paquetes de la solución*

El código de la solución está contenido en dos paquetes: *res* y *localProxy*. En el primero se almacena el código con formato XML[11] que define las vistas, las imágenes y las cadenas de texto que se utilizan tanto en español como en inglés. En el segundo están todas las clases Java de la solución agrupadas en dependencia de su funcionalidad.

El paquete *localproxy* contiene a *proxycore, data y utils*, constituyendo el modelo de la aplicación, encargados de almacenar las clases para el funcionamiento del servicio de *proxy* local, para el acceso y persistencia de datos y para funcionalidades útiles respectivamente. Además, se encuentran *firewallscreens*, *profilescreens*, *proxyscreen* y *tracescreen* que se encargan de gestionar las vistas relativas a cortafuegos, perfiles, *proxy* y trazas respectivamente. Cada paquete que gestiona una vista, como por ejemplo *addEditProfile* (ver Figura 6) contiene principalmente las siguientes clases e interfaces:

- Una interfaz llamada "*contract*" que define la conexión entre la vista y el presentador mediante interfaces.
- Una *Activity* la cual crea *fragments* y presentadores.
- Un *Fragment* el cual implementa la vista definida en "*contract*".
- Un presentador el cual implementa el presentador definido en "*contract*".

El presentador aloja la lógica relacionada con cada evento ocurrido en la vista y accede a los datos necesitados en el paquete *data* o las funcionalidades en *proxycore* y en *utils*, y la vista correspondiente maneja el trabajo de la interfaz de usuario (IU) de Android. Esta última casi no contiene lógica; convierte los comandos del presentador a acciones de IU y escucha las acciones del usuario que luego se pasan a este.

### *2.4.2 Patrones de diseño*

Un patrón de diseño es una descripción de la comunicación entre objetos y clases, personalizada para resolver un problema de diseño general en un contexto particular. Identifica clases, instancias, roles, colaboraciones y la distribución de responsabilidades (GAMMA et al. 1994). Se presentan como pares de problema-solución con nombre, sugiriendo aspectos relacionados con la asignación de responsabilidades.
Los patrones de diseño se caracterizan por:

- Representar soluciones técnicas a problemas concretos.

11 XML: del inglés *eXtensible Markup Language*, traducido al español como Lenguaje de Marcado Extensible.

- Propiciar la reutilización.
- Representar problemas frecuentes.

***Patrones GRASP***

Los patrones GRASP[12] describen los principios fundamentales de la asignación de responsabilidades a objetos (Larman 2004). El nombre se eligió para indicar la importancia de captar estos principios si se quiere diseñar un software de manera eficaz. Fueron utilizados estos patrones con el objetivo de realizar una correcta asignación de responsabilidades a las clases de objetos diseñadas. Se muestra a continuación los detalles de su uso.

- **Experto**: se usa más que cualquier otro al asignar responsabilidades, es un principio básico que suele utilizarse en el diseño orientado a objetos (Larman 2004). Consiste en la asignación de una responsabilidad a la clase que cuenta con la información necesaria para llevarla a cabo. El uso de este patrón da pie a un bajo acoplamiento y una alta cohesión, lo que favorece al hecho de tener sistemas más robustos y de fácil mantenimiento. El cumplimiento de una responsabilidad requiere a menudo información distribuida en varias clases de objetos. En la aplicación móvil para la autenticación con servidor *proxy* web en NovaDroid se evidencia este patrón en las clases HttpForwarder, FirewallRuleLocalDataSource, ProfilesLocalDataSource, TraceDataSource y UsersLocalDataSource, las cuales, puesto que contienen la información necesaria, se encargan únicamente de realizar las funcionalidades de *proxy*, de acceder a las reglas de cortafuegos almacenadas, de acceder a los perfiles almacenados, de acceder a las trazas almacenadas y de acceder a los usuarios almacenados respectivamente.

12 GRASP: Del inglés *General Responsibility Assignment Software Patterns*. Traducido al español Patrones Generales de Software para Asignar Responsabilidades.

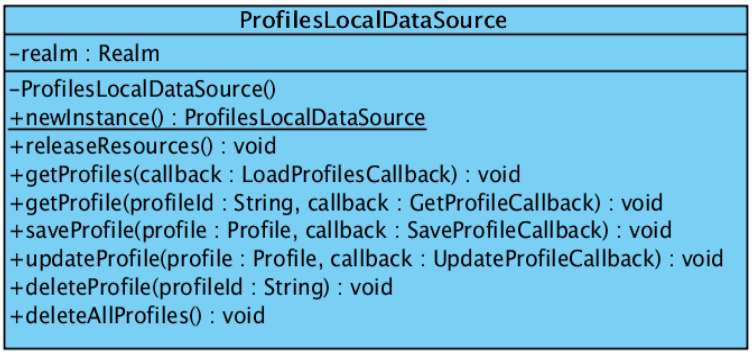



*Figura 7: Aplicación del patrón experto en la clase ProfilesLocalDataSource*

- **Creador**: el patrón Creador guía la asignación de responsabilidades relacionadas con la creación de objetos, una tarea muy común. Su intención básica es encontrar un creador que necesite conectarse al objeto creado en alguna situación. Su uso favorece el bajo acoplamiento (Larman 2004). En la solución se puede evidenciar el uso de este patrón en las clases ProfilesLocalDataSource, TraceDataSource, FirewallRuleLocalDataSource y UsersLocalDataSource, cada una de estas clases son expertas en la creación de perfiles, trazas de navegación, reglas de cortafuegos y usuarios respectivamente.

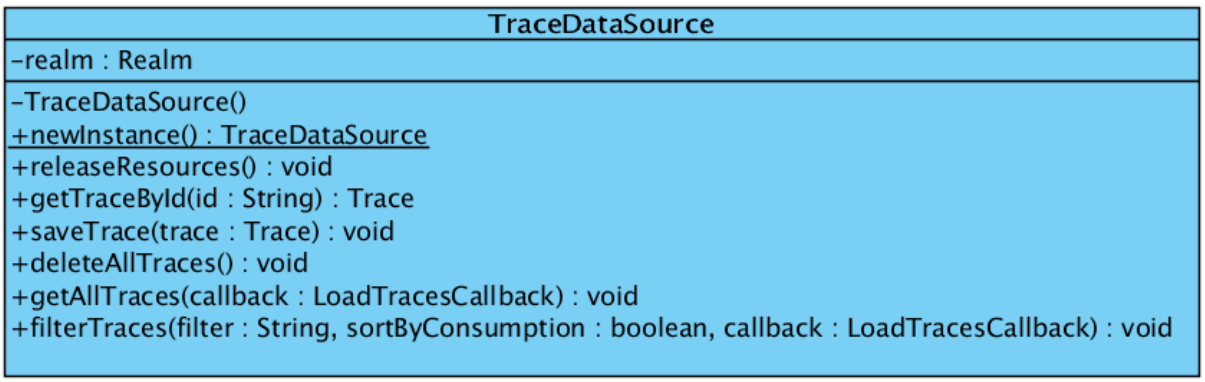



*Figura 7: Aplicación del patrón Creador en la clase TraceDataSource*

- **Bajo Acoplamiento**: el acoplamiento es una medida de la fuerza con que una clase está conectada a otras clases, con que las conoce y con que recurre a ellas. Una clase con bajo (o débil) acoplamiento no depende de muchas otras (Larman 2004). El bajo acoplamiento soporta el diseño de clases más independientes y reutilizables, lo cual reduce el impacto de los cambios y acrecienta la oportunidad de una mayor productividad. Ejemplo del uso de este patrón en la solución se muestra en las clases WifiUtils, ActivityUtils, StringUtils, Piper, Encripter, ProxyService, Profile, Trace, User y FirewallRule donde se minimizan las relaciones de estas con el resto de las clases por lo que permite que sean reutilizadas con facilidad.

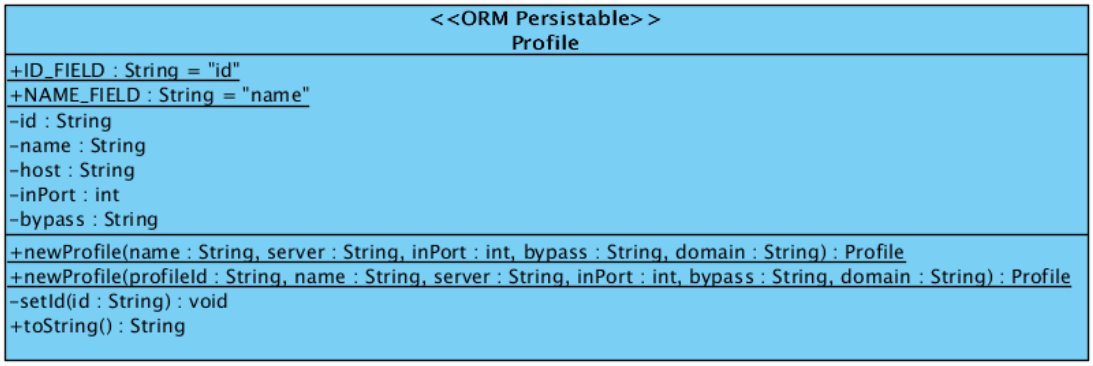



*Figura 8: Aplicación del patrón Bajo Acoplamiento en la clase Profile*

- **Alta Cohesión**: en la perspectiva del diseño orientado a objetos, la cohesión (o más exactamente, la cohesión funcional) es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme, clases con responsabilidades moderadas en un área funcional que colaboran con las otras para llevar a cabo las tareas (Larman 2004). En el diseño de la solución, se evidencia este patrón en la clase HttpForwarder, la cual redirecciona las peticiones y maneja el proceso de autenticación, y para estas tareas se utilizan funciones de la clase Piper para realizar copias asíncronas de datos y funciones de la clase HttpParser para analizar peticiones HTTP.

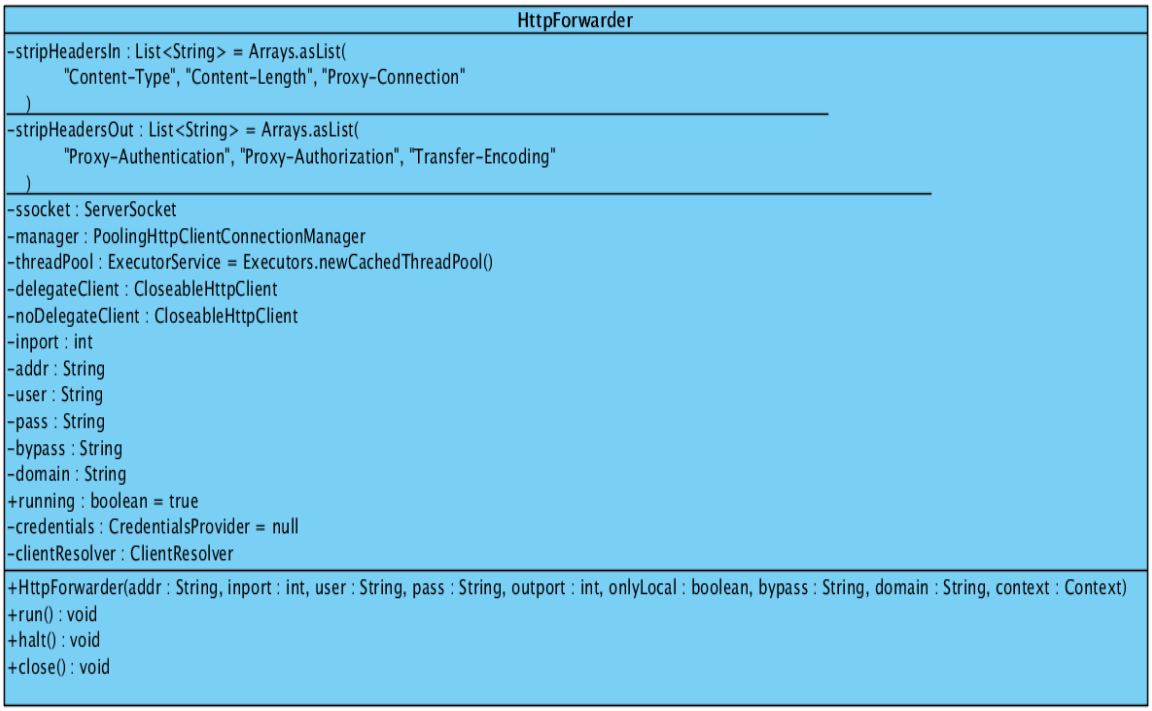



*Figura 9: Aplicación del patrón Alta Cohesión en la clase HttpForwarder*

- **Controlador**: un controlador es un objeto de interfaz no destinada al usuario que se encarga de manejar un evento del sistema (Larman 2004). En la solución se evidencia este patrón en las clases presentadoras, correspondientes a la arquitectura seleccionada donde tienen como responsabilidad manejar los eventos ocurridos en las vistas.

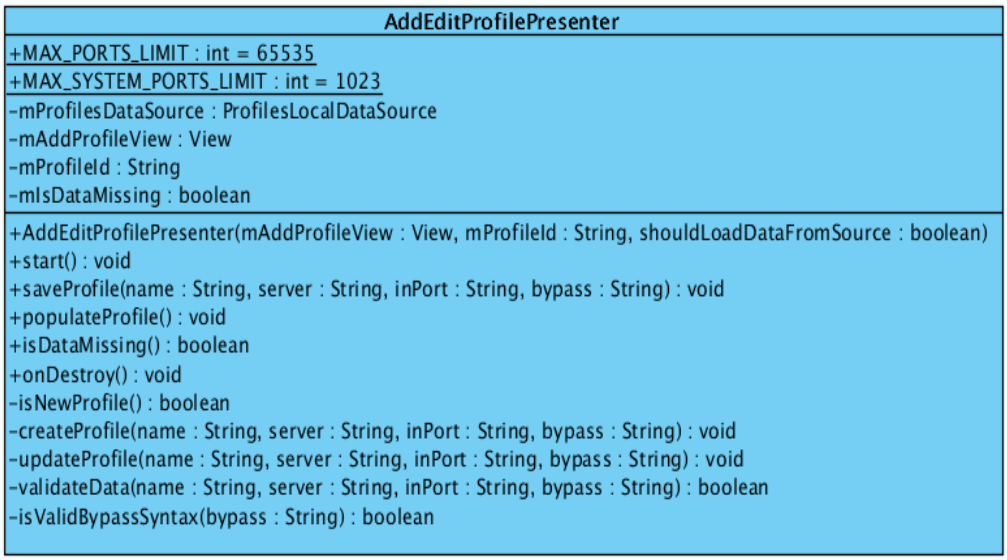



*Figura 10: Aplicación del patrón Controlador en la clase AddEditProfilePresenter*

***Patrones GOF***

Los patrones GOF[13] son alternativas de solución a problemas conocidos pero son mucho más específicas las situaciones en las que se aplican. Se clasifican en creacionales, estructurales y de comportamiento (GAMMA et al. 1994). A continuación, se enuncian brevemente los utilizados en el sistema.

- **Singleton**: Está diseñado para restringir la creación de objetos pertenecientes a una clase o el valor de un tipo a un único objeto. Su intención consiste en garantizar que una clase solo tenga una instancia y proporcionar un punto de acceso global a ella (GAMMA et al. 1994). Se empleó en las clases ApplicationPackageLocalDataSource y AppPreferencesHelper debido a la necesidad de que exista una sola instancia de cada una en orden de minimizar el consumo de recursos de memoria.

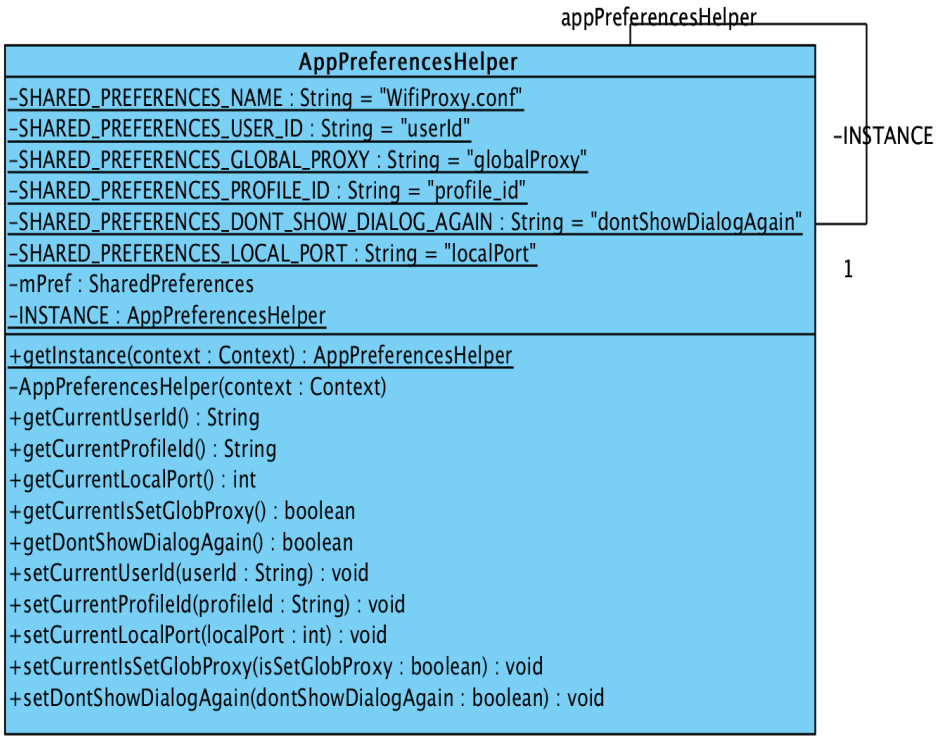



*Figura 11: Aplicación del patrón Singleton en la clase AppPreferencesHelper*

- **Fachada**: Sirve para proveer de una interfaz unificada sencilla que haga de intermediaria entre un

13 GOF: Del inglés *Gang of Four*. Traducido al español La Pandilla de los Cuatro.

cliente y una interfaz o grupo de interfaces más complejas. Se utiliza ampliamente para hacer más fácil y entendible el trabajo con bibliotecas de software (GAMMA et al. 1994). Se empleó para la gestión de cada una de las vistas de la aplicación donde existe una interfaz llamada View y otra Presenter que son implementadas por la clase para la vista y por la clase para el presentador facilitando de esta manera el trabajo con ellas.

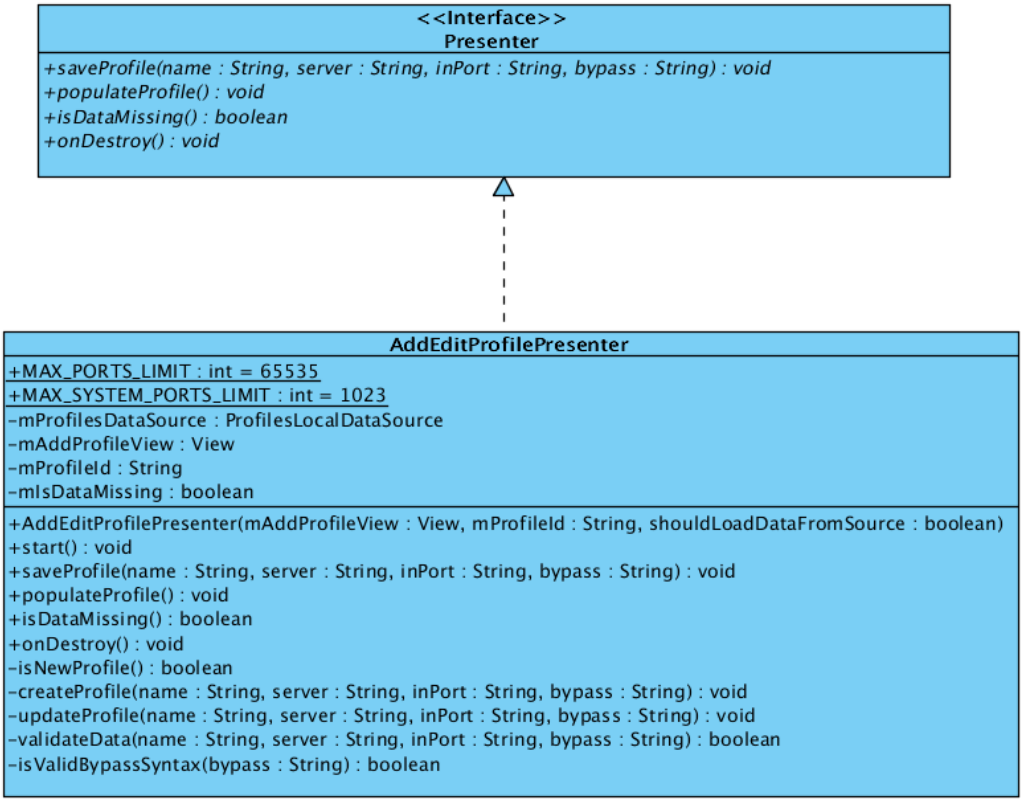



*Figura 12: Aplicación del patrón Fachada en la clase AddEditProfilePresenter*

### *2.4.3 Diagrama de clases del diseño*

Un diagrama de clases de diseño (DCD) representa las especificaciones de las clases e interfaces de software en una aplicación. Entre la información general encontramos:

- Clases, asociaciones y atributos.

- Interfaces, con sus operaciones y constantes.
- Métodos.
- Información acerca del tipo de los atributos.
- Navegabilidad.
- Dependencias.

A diferencia de las clases conceptuales del Modelo del Dominio, las clases de diseño de los DCD muestran las definiciones de las clases de software en lugar de los conceptos del mundo real (Larman 2004). En el Anexo 13 se muestra un fragmento del diagrama de clases del diseño relativo a la persistencia y lógica de acceso a datos.

## 2.5 Conclusiones parciales

A partir del desarrollo del presente capítulo es posible afirmar que la captura de los requisitos y la elaboración de las historias de usuario correspondientes permitió comprender mejor las funcionalidades de la aplicación que se desea desarrollar, el comportamiento de la misma y la secuencia de actividades que debe de realizar el usuario para lograr su objetivo. Además, la utilización de la arquitectura MVP y los patrones de diseño GRASP y GOF contribuyó al diseño de la aplicación, proporcionando una estructura para la misma y posibilitando el empleo de buenas prácticas de programación y la reutilización de código. Como resultado principal de este capítulo quedó plasmada la propuesta de solución para la problemática planteada, detallada a través de los 21 requisitos funcionales y los 8 no funcionales, las 14 historias de usuario, la arquitectura y el uso de patrones de diseño.

# Capítulo 3. Implementación y prueba

Se exponen en el presente capítulo las especificaciones asociadas a la implementación de la aplicación. Se detallan las iteraciones realizadas para lograr el producto deseado y se describen las pautas de codificación utilizadas. Además, se describe el proceso de pruebas al que fue sometida la aplicación resultante con el objetivo de validar el cumplimiento de los requisitos identificados y lograr la aceptación del cliente.

## 3.1 Implementación

La codificación de la solución propuesta tiene lugar una vez que se han definido las historias de usuario y se ha concluido el diseño de la aplicación. Está encaminada a desarrollar de forma iterativa e incremental un producto completo listo para el despliegue, obteniendo versiones útiles de forma rápida, las que paulatinamente completan el desarrollo de la aplicación.

La implementación está incluida en la fase de ejecución definida por la metodología AUP-UCI. Para la obtención del resultado esperado fueron realizadas 3 iteraciones, los resultados de cada iteración se detallan a continuación.

*Tabla 7: Resultados de las iteraciones realizadas en la fase de implementación*

| Iteración | Resultados | Duración |
|---|---|---|
| 1 | Codificación de las historias de usuario necesarias para lograr la aplicación funcional: “Autenticar usuario”, “Gestionar perfil de conexión”, “Iniciar servicio de *proxy* local”, “Listar perfiles de conexión”, “Detener servicio de *proxy* local” y “Restablecer configuración”. | 41 días |
| 2 | Codificación de las historias de usuario necesarias para la aplicación de reglas de cortafuegos: “Gestionar regla de cortafuegos”, “Listar reglas de cortafuegos” y “Activar o desactivar regla de cortafuegos” | 23 días |
| 3 | Codificación de las historias de usuarios relacionadas con | 15 días |

| | | |
|---|---|---|
| | las trazas de navegación: “Adicionar traza de navegación”, “Listar trazas de navegación”, “Añadir traza de navegación como regla de cortafuegos”, “Ordenar trazas de navegación” y “Buscar trazas de navegación”. | |

### *3.1.1 Estándares de codificación*

Los estándares de codificación permiten un mejor entendimiento del código por parte de todos los miembros del equipo de desarrollo y en consecuencia hacen que el código sea fácil de mantener. En la solución se establecieron las siguientes convenciones:

- Se empleará el idioma inglés para la codificación de la solución.
- El código será tabulado y espaciado a través del formato que aplica la combinación de teclas *ALT+Shift+F* definida en la configuración del IDE Android Studio, en la sección referente a los atajos de teclado (*File/Settings/Keymap*), estableciendo la configuración del IDE NetBeans.
- Los comentarios para una línea comenzarán con los caracteres “//”.
- Para la nomenclatura de las clases e interfaces se utilizará el estilo de capitalización *UpperCamelCase* («*Google Java Style Guide*» nd), con el cual se capitaliza la primera letra de cada palabra. Ejemplo: *AddEditProfilePresenter*.
- Los nombres de las variables y los métodos responderán al estilo de capitalización *lowerCamelCase* («*Google Java Style Guide*» nd), con el cual se capitaliza la primera letra de las palabras excepto la primera palabra. Ejemplo: *mView*.
- No se usarán nombres de variables que coincidan con palabras reservadas.
- No se emplearán caracteres especiales (@, #, $, %, ^, &, * u otros) para la nomenclatura.
- Los nombres de variables globales deberán escribirse todo en mayúsculas con las palabras separadas por un guión bajo ("_"). Todas serán declaradas como *public static*.
- Se inicializarán las variables locales donde se declaran. La única razón para no hacerlo será si su valor inicial depende de cálculos posteriores.

## 3.2 Pruebas de software

La prueba del software es un elemento crítico para la garantía de calidad del software y representa una revisión de las especificaciones, del diseño y de la codificación. Una vez generado el código fuente es necesario probar el software para descubrir y corregir la mayor cantidad de errores posibles antes de ser entregado. Su objetivo es diseñar una serie de casos de prueba que tengan una alta probabilidad de encontrar errores. Estas se dividen principalmente en los siguientes niveles: pruebas de unidad, de integración, de validación, de sistema y de aceptación (Pressman 2010).

Como parte del proceso de desarrollo de la solución se decidió la aplicación de pruebas, comenzando “en lo pequeño” y avanzando hacia “lo grande”. Se aplicaron primeramente pruebas en el nivel de unidad, utilizando el método de caja blanca y automatizadas con la herramienta Junit, concentrándose en el código fuente de la solución evaluando el correcto funcionamiento de las funcionalidades más importantes. Luego se procedió a la aplicación de pruebas en el nivel de validación, utilizando el método de caja negra y la técnica de partición de equivalencia. En este nivel los requisitos establecidos, descritos a través de las historias de usuario, se validaron confrontándose con el software construido. Por último, se aplicaron pruebas en el nivel de aceptación, utilizando el proceso de prueba *Alfa* con el fin de permitir al cliente evaluar el correcto funcionamiento de la solución.

Los problemas detectados se clasificaron en: No Conformidades Significativas (NCS) y en No Conformidades No Significativas (NCNS). A continuación, se describen los aspectos que se tuvieron en cuenta en cada clasificación.

- **NCS**: son las no conformidades referentes a las funcionalidades de la aplicación: validaciones incorrectas o respuestas de la aplicación diferente a lo descrito previamente en las historias de usuario.
- **NCNS**: son las no conformidades en cuanto al diseño de la propuesta de solución y errores ortográficos.

Para la ejecución de las pruebas se utilizaron los siguientes dispositivos:

**Teléfono celular:**

- Modelo: Samsung Galaxy S4 Mini GT-I9192.
- Sistema operativo: Android 7.1.2.

- SDK: 25.
- CPU: Qualcomm Snapdragon S4 Plus / 400.
- RAM: 1.5 GB.

**Tableta:**

- Modelo: GDM Allwinner N801L2BC.
- Sistema operativo: NovaDroid 6.0.1.
- SDK: 23.
- CPU: 5x ARM Cortex-A7 @ 1800 MHz
- RAM: 2 GB.

**Emulador Genymotion:**

- Modelo: Genymotion_vbox86p_4.3_150216_21300.
- Sistema operativo: Android 4.3.
- SDK: 18.
- CPU: Intel® Core™ i5-4210U @1.7 GHz.
- RAM: 512 MB.

Se detallan a continuación las pruebas realizadas en cada uno de los niveles.

### *3.2.1 Pruebas de unidad*

Las pruebas de unidad o unitarias son el proceso de probar componentes del programa tales como métodos o clases de objetos. Las funciones o los métodos individuales son el tipo más simple de componente. Las pruebas deben llamarse para dichas rutinas con diferentes parámetros de entrada (Sommerville 2011).

Para el diseño de los casos de prueba fue utilizado el método de caja blanca, donde las pruebas se enfocan en la estructura de control del programa. Los casos de prueba se derivan para asegurar que todos los enunciados en el programa se ejecutaron al menos una vez durante las pruebas y que todas las condiciones lógicas se revisaron (Pressman 2010).

Con el fin de automatizar este tipo de pruebas sobre la solución se decidió emplear la herramienta Junit, la cual está integrada con el IDE Android Studio que se utilizó para la codificación. Se implementaron cuatro

casos de prueba a través de la clase HttpForwarderTest, donde se comprueba el correcto funcionamiento de la clase HttpForwarder, asegurándose que se ejecute cada flujo de control. Esta clase constituye básicamente el núcleo de la solución, es responsable de escuchar las peticiones realizadas en el dispositivo y resolverlas a través del servidor *proxy* de la institución. Los casos de prueba diseñados se enfocan principalmente en comprobar la correcta resolución de las peticiones HTTP y HTTPS.

Los casos de prueba fueron ejecutados satisfactoriamente en su primera iteración, se muestra a continuación una imagen de los resultados de la aplicación de estas pruebas realizadas con Junit la clase HttpForwarder.

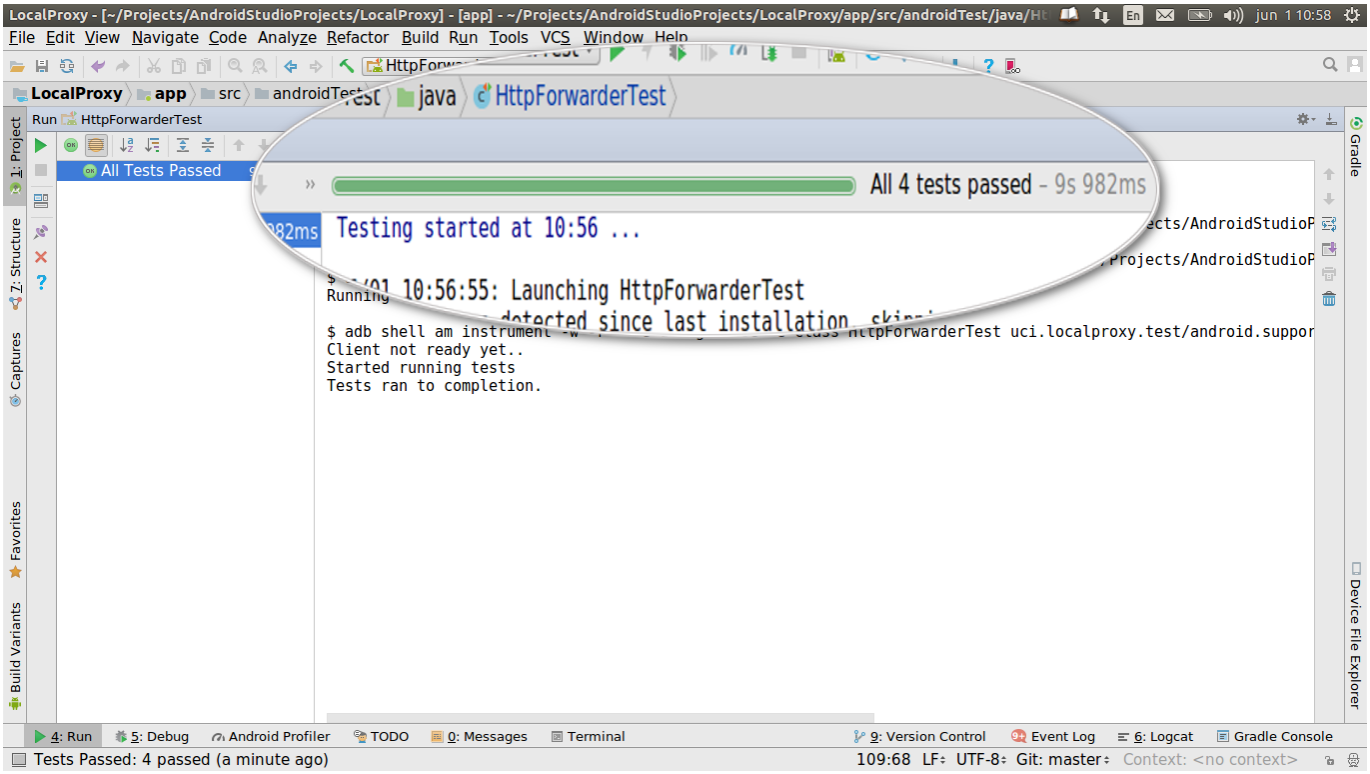



*Figura 13: Resutados de las pruebas aplicadas a la clase HttpForwarder*

A partir de la interpretación de estos resultados se afirma que los mismos garantizan el correcto funcionamiento de los métodos de la clase, la cual constituye el núcleo de la aplicación.

### *3.2.2 Pruebas de validación*

La prueba de validación proporciona un aseguramiento final de que el software cumple con todos los requisitos funcionales, de comportamiento y desempeño. La prueba se concentra en las acciones visibles para el usuario y en la salida que este puede reconocer. La validación se alcanza cuando el software funciona de tal manera que satisface las expectativas razonables (especificación de requisitos de software) del cliente. Se logra mediante una serie de pruebas que demuestran que se cumple con los requisitos (Pressman 2010).

Para realizar estas pruebas se aplicó el tipo de prueba funcional, ya que asegura el apropiado trabajo de los requisitos funcionales, incluyendo la navegación, entrada de datos, procesamiento y obtención de resultados. Las metas de esta prueba es verificar la apropiada aceptación de datos y verificar el procesamiento, recuperación e implementación adecuada de las reglas del negocio (Pressman 2010).

Para el diseño de los casos de prueba se utilizó el método de caja negra, el cual se centra en los requisitos funcionales del software. Es decir, la prueba de caja negra permite al ingeniero del software obtener conjuntos de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa (Pressman 2010).

Como técnica se utilizó la de partición de equivalencia, o sea, dividir el dominio de entrada de un programa en clases de datos a partir de las cuales pueden derivarse casos de prueba. El diseño de casos de prueba para partición equivalente se basa en una evaluación de las clases de equivalencia para una condición de entrada (Pressman 2010).

Los casos de prueba se diseñaron según las funcionalidades descritas cada una de las historias de usuario. La intención que se persigue con estos artefactos es lograr una compresión específica de las condiciones que la solución debe cumplir. Cada plantilla de casos de pruebas recoge la especificación de una historia de usuario, dividida en secciones y escenarios, detallando las funcionalidades descritas en ella y describiendo cada variable.

A continuación, se muestran los casos de prueba correspondientes a las historias de usuario “Autenticar usuario”, “Iniciar servicio de *proxy* local” y “Gestionar perfil de conexión” ya que representan las HU principales de la solución. Los casos de prueba asociados al resto de las historias de usuario se pueden encontrar del Anexo 14 al Anexo 24.

## Descripción general

*El caso de prueba inicia al ejecutarse la aplicación. Se presentan al usuario dos campos de texto para introducir el nombre de usuario y la contraseña para autenticarse con el servidor proxy de la institución. El caso de prueba termina cuando se completan los campos.*

## Condiciones de ejecución

### Sección 1: Insertar nombre de usuario y contraseña

| Escenario | Descripción | Nombre de usuario | Contraseña | Respuesta del sistema | Flujo central |
|---|---|---|---|---|---|
| EC 1.1 Insertar nombre de usuario y contraseña | El usuario introduce el nombre de usuario y contraseña de su cuenta para ser autenticado con e servidor *proxy* de la institución. | V<br><br>parodriguez | V<br><br>test | Insertar el nombre de usuario y contraseña, mientras se introduce semuestra una lista emergente quemuestra los usuarios previamente insertados que comiencen con los caracteres escritos hasta el momento. | Iniciar la aplicación. |

### Sección 2: Autocompletar usuario y contraseña

| Escenario | Descripción | | | Respuesta del sistema | Flujo central |
|---|---|---|---|---|---|
| EC 2.1 Autocompletar nombre de usuario y contraseña | Se secciona un nombre de usuario mostrado en la lista emergente que aparece mientras se inserta el nombre de usuario. | | | Se autocompleta el campo del usuario seleccionado y si este usuario marcó la opción “Recordar contraseña” se autocompleta además la contraseña. | Iniciar la aplicación. |

### Sección 3: Visualizar contraseña

| Escenario | Descripción | | | Respuesta del sistema | Flujo central |
|---|---|---|---|---|---|
| EC 3.1 Visualizar contraseña | Se pulsa en el botón que indica la opción de visualizar contraseña. | | | Se muestra la contraseña insertada por el usuario hasta el momento mientras se mantiene pulsado el botón. | Iniciar la aplicación. |

*Figura 14: Caso de prueba "Autenticar usuario"*

## Descripción general

El caso de prueba inicia al ejecutarse la aplicación. Se muestra al usuario además de los campos para introducir el nombre de usuario y la contraseña un campo para insertar el puerto por el cual se escucharán las peticiones del dispositivo, una opción para seleccionar de una lista emergente el perfil de conexión a utilizar, un checkbox para configurar o no el proxy del sistema y un botón flotante para iniciar el servico de proxy local. El caso de prueba finaliza al iniciar el servicio de proxy local.

## Condiciones de ejecución

Debe haberse creado al menos un perfil de conexión.

## Sección 1: Iniciar servicio de proxy local

| Escenario | Descripción | Nombre de usuario | Contraseña | Perfil | Puerto local | Respuesta del sistema | Flujo central |
|---|---|---|---|---|---|---|---|
| EC 1.1 Iniciar servicio de *proxy* local. | El usuario introduce los datos necesarios para iniciar el servicio de *proxy* local. | V<br>test | V<br>test | V<br>uci | V<br>8080 | Se inicia el servicio de *proxy* local satisfactoriamente , se desabilitan los campos de "Nombre de usuario", "Contraseña", "Perfil" y "Puerto local". Finalmente se muestra una notificación mostrando que el servicio está activo, esta permanece hasta que se detiene el servicio. | Iniciar la aplicación y completar los campos. |
| | | V<br>test | V<br>test | V<br>cujae | V<br>3128 | | |
| EC 1.2 Iniciar servicio de proxy local con datos incorrectos. | El usuario introduce los datos necesarios para iniciar el servicio de *proxy* local. | I | V<br>test | V<br>uci | V<br>8080 | La aplicación verifica que todos los datos introducidos sean correctos, si un dato no es válido no se inicia el servicio y se muestra un mensaje indicando donde está el error. | Iniciar la aplicación y completar los campos. |
| | | V<br>test | I | V<br>uci | V<br>8080 | | |
| | | V<br>test | V<br>test | I | V<br>8080 | | |
| | | V<br>test | V<br>test | V<br>uci | I<br>80 | | |
| | | V<br>test | V<br>test | V<br>uci | I<br>65537 | | |

*Figura 15: Caso de prueba "Iniciar servicio de proxy local"*

## Descripción general

*El caso de prueba inicia cuando el usuario selecciona la opción de Perfiles en el menú lateral de la aplicación. Se presenta entonces al usuario el listado de los perfiles de conexión creados hasta el momento y además la opción de adicionar un nuevo perfil. El caso de prueba termina cuando el usuario abandona la sección de Perfiles.*

## Condiciones de ejecución

Condiciones para SC2, SC3, SC4: debe existir al menos un perfil de conexión creado previamente.

## Sección 1: Adicionar perfil de conexión

| Escenario | Descripción | Nombre | Servidor | Puerto de entrada | Bypass | Respuesta del sistema | Flujo central |
|---|---|---|---|---|---|---|---|
| EC 1.1 Adicionar perfil de conexión. | El usuario introduce los datos necesarios para insertar un nuevo perfil de conexión. | V<br>uci | V<br>10.0.0.1 | V<br>8080 | V<br>127.0.0.1,localhost, | Se introduce el nuevo perfil de conexión y se regresa a la vista del listado de perfiles donde se observa el perfil recién creado. | Seleccionar la opción "Perfiles" del menú lateral de la aplicación, luego pulsar el botón flotante que indica la opción de adicionar, después introducir los datos y pulsar el botón flotante que indicala opción de aceptar. |
| | | V<br>test | V<br>proxy.test.cu | V<br>3128 | V<br>127.0.0.1,localhost, *test.cu | | |
| EC 1.2 Adicionar perfil de conexión con datos incorrectos | El usuario introduce los datos necesarios para insertar un nuevo perfil de conexión y la aplicación verifica que todos los datos sean correctos | I | V<br>10.0.0.3 | V<br>3128 | V<br>127.0.0.1,localhost | La aplicación verifica que todos los datos introducidos sean correctos, si un dato no es válido no se adiciona el perfil y se muesta un mensaje debajo del campo no válido indicando el error. | Seleccionar la opción "Perfiles" del menú lateral de la aplicación, luego pulsar el botón flotante que indica la opción de adicionar, después introducir los datos y pulsar el botón flotante que indicala opción de aceptar. |
| | | V<br>test | I | V<br>3128 | V<br>127.0.0.1,localhost | | |
| | | V<br>test | V<br>10.0.0.3 | I<br>-5 | V<br>127.0.0.1,localhost | | |
| | | V<br>test | V<br>10.0.0.3 | I<br>65536 | V<br>127.0.0.1,localhost | | |
| | | V<br>test | V<br>10.0.0.3 | V<br>65535 | I | | |

## Sección 2: Modificar perfil de conexión

| Escenario | Descripción | Nombre | Servidor | Puerto de entrada | Bypass | Respuesta del sistema | Flujo central |
|---|---|---|---|---|---|---|---|
| EC 2.1 Modificar perfil de conexión. | El usuario selecciona un perfil previamente creado y modifica los datos que desee | V<br>uci | V<br>10.0.0.1 | V<br>8080 | V<br>127.0.0.1,localhost, | Se modifica el nuevo perfil de conexión, se regresa a la vista de visualizar el perfil y se muestra un mensaje indicando que se modificó correctamente | Seleccionar la opción "Perfiles" del menú lateral de la aplicación, luego seleccionar un perfil previamente creado de la lista mostrada, después pulsar el botón flotante que indica la acción de modificar, entonces modificar los datos y pulsar el botón flotante que indicala opción de aceptar. |
| | | V<br>test | V<br>proxy.test.cu | V<br>3128 | V<br>127.0.0.1,localhost, *test.cu | | |
| EC 2.2 Modificar perfil de conexión con datos incorrectos | El usuario selecciona un perfil previamente creado, modifica los datos que desee y la aplicación verifica que estos datos sean correctos | I | V<br>10.0.0.3 | V<br>3128 | V<br>127.0.0.1,localhost | La aplicación verifica que todos los datos introducidos sean correctos, si un dato no es válido no se modifica el perfil y se muesta un mensaje debajo del campo no válido indicando el error. | Seleccionar la opción "Perfiles" del menú lateral de la aplicación, luego seleccionar un perfil previamente creado de la lista mostrada, después pulsar el botón flotante que indica la acción de modificar, entonces modificar los datos y pulsar el botón flotante que indicala opción de aceptar. |
| | | V<br>test | I | V<br>3128 | V<br>127.0.0.1,localhost | | |
| | | V<br>test | V<br>10.0.0.3 | I<br>-5 | V<br>127.0.0.1,localhost | | |
| | | V<br>test | V<br>10.0.0.3 | I<br>65536 | V<br>127.0.0.1,localhost | | |

## Sección 3: Mostrar perfil de conexión

| Escenario | Descripción | | | | | Respuesta del sistema | Flujo central |
|---|---|---|---|---|---|---|---|
| EC 3.1 Mostrar perfil de conexión. | Se muestran los datos de un perfil de conexión previamente creado. | | | | | Se muestra en una nueva vista todos los datos de un perfil de conexión previamente creado | Seleccionar la opción "Perfiles" del menú lateral de la aplicación y luego seleccionar un perfil previamente creado de la lista mostrada. |

## Sección 4: Eliminar perfil de conexión

| Escenario | Descripción | | | | | Respuesta del sistema | Flujo central |
|---|---|---|---|---|---|---|---|
| EC 4.1 Eliminar perfil de conexión. | Se elimina un perfil de conexión previamente creado | | | | | Se se elimina el perfil de conexión y se regresa a la vista del listado de perfiles. | Seleccionar la opción "Perfiles" del menú lateral de la aplicación, luego seleccionar un perfil previamente creado de la lista mostrada y después pulsar la opción de "eliminar perfil" mostrada en la parte izquierda de la barra de herramisntas de la vista |

*Figura 16: Caso de prueba "Gestionar perfil de conexión"*

Fueron realizadas tres iteraciones de pruebas, ejecutándose al término de cada una de ellas pruebas de regresión con el objetivo de asegurar que, al resolverse las no conformidades detectadas, estas no introdujeran nuevos errores en la solución.

Se presenta a continuación un gráfico mostrando los resultados obtenidos en las iteraciones realizadas:

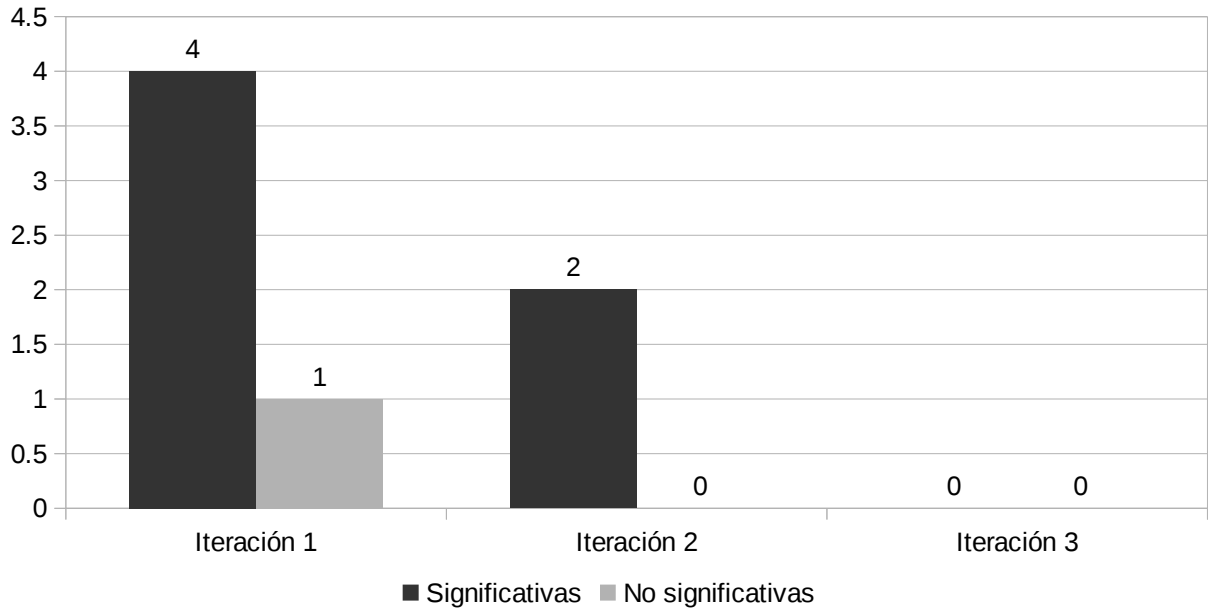



*Figura 17: Gráfica correspondiente a las no conformidades encontradas por iteración en las pruebas de validación*

En la primera iteración se detectaron 4 NCS y 1 NCNS, las cuales fueron resueltas satisfactoriamente.

No Conformidades Significativas:

1. La aplicación se detuvo al iniciar el servicio de *proxy* local.
2. La aplicación no validó que al adicionar un nuevo perfil de conexión el campo del puerto fuese un número entero entre 0 y 65535.
3. La aplicación no validó que al modificar un perfil de conexión el campo del puerto fuese un número entero entre 0 y 65535.
4. La aplicación no mostró ningún perfil en la lista desplegable desde la interfaz principal de la aplicación.

No Conformidades No Significativas:

1. La aplicación posee faltas de ortografía en algunas etiquetas de diferentes vistas.

En la segunda iteración fueron detectadas 2 NCS, estas siendo también solucionadas de igual manera que en la iteración anterior.

No Conformidades Significativas:

1. La aplicación se detuvo al eliminar un perfil.
2. La aplicación se detuvo al auto-completar el nombre de usuario y contraseña.

Por último, en la tercera iteración no fueron detectadas no conformidades, culminando de esta manera las pruebas de validación.

### *3.2.3 Pruebas de aceptación*

Esta es la etapa final en el proceso de pruebas, antes de que el sistema se acepte para uso operacional. El sistema se pone a prueba con datos suministrados por el cliente del sistema, en vez de datos de prueba simulados. Las pruebas de aceptación revelan los errores y las omisiones en la definición de requerimientos del sistema, ya que los datos reales ejercitan el sistema en diferentes formas a partir de los datos de prueba. Asimismo, las pruebas de aceptación revelan problemas de requerimientos, donde las instalaciones del sistema en realidad no cumplan las necesidades del usuario o cuando sea inaceptable el rendimiento del sistema (Sommerville 2011).

Para llevar a cabo las pruebas de aceptación se hizo necesario, a través del tipo de prueba *alfa,* incorporar al cliente o usuario final directamente al proceso de prueba de la aplicación. Entiéndase por prueba *alfa* cuando esta se lleva a cabo en el sitio del desarrollador por un grupo representativo de usuarios finales, es decir, en un ambiente controlado, propiciando que el desarrollador pueda registrar errores y problemas de uso (Pressman 2010).

Estas pruebas fueron llevadas a cabo por un grupo de tres personas pertenecientes al proyecto DroidLab del centro CESOL de la universidad, estas junto con el desarrollador para supervisar la actividad y registrar los errores encontrados.

Fueron ejecutadas tres iteraciones de pruebas para este nivel. Se presenta a continuación un gráfico mostrando los resultados obtenidos en las iteraciones realizadas:

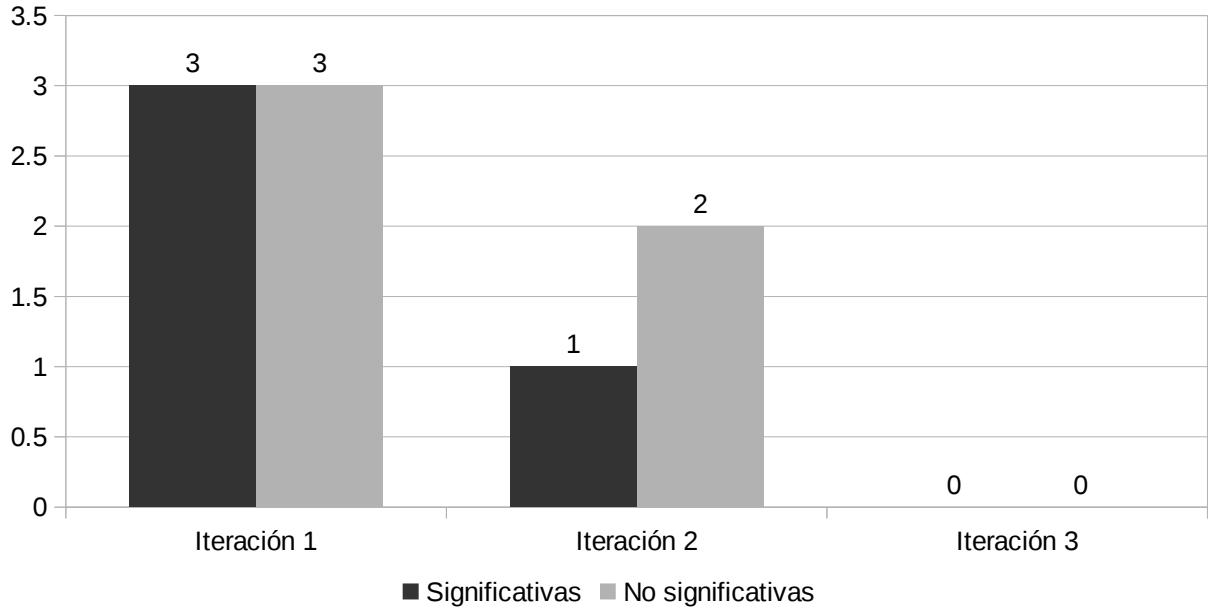



*Figura 18: Gráfica correspondiente a las no conformidades encontradas en las pruebas de aceptación*

En la primera iteración se detectaron 3 NCS y 3 NCNS, estas fueron resueltas satisfactoriamente.

No Conformidades Significativas:

1. La aplicación no resolvió las peticiones que coincidían con la regla de excepciones insertada en la configuración del perfil.
2. La aplicación se detuvo al realizarse una petición de tipo GET por parte del navegador instalado por defecto en el dispositivo.
3. La aplicación no guardó la última configuración con la que se ejecutó.

No Conformidades No Significativas:

1. La aplicación posee faltas de ortografía en algunas etiquetas de diferentes vistas.
2. La aplicación posee etiquetas sin traducir al idioma inglés.
3. La aplicación muestra incorrectamente los textos presentes en la vista que se muestra al seleccionar la opción “Acerca” que se encuentra en el menú lateral desplegable.

En la segunda iteración se detectó 1 NCS y 2 NCNS, las cuales fueron solucionadas de igual manera que en la iteración anterior.

No Conformidades Significativas:

1. La aplicación no resolvió los estilos CSS[14] de las peticiones tipo GET realizadas desde los navegadores.

No Conformidades No Significativas:

1. El texto de ejemplo en el campo de *bypass* al adicionar o modificar perfil se sobrepone al texto insertado por el usuario.
2. Al presionar el botón flotante de iniciar el servicio de *proxy* local este no cambia la imagen que sugiere “iniciar el servicio” por la que sugiere “parar el servicio”.

Para la tercera iteración no fueron detectadas no conformidades. De esta manera el cliente afirmó su aceptación mediante la firma del Acta de Aceptación (ver Anexo 25) y en consecuencia se consideró concluido el desarrollo de la solución.

## 3.4 Conclusiones parciales

Luego del desarrollo del presente capítulo es posible afirmar que la implementación de la solución de forma iterativa e incremental mediante tres iteraciones permitió completar paulatinamente el desarrollo de la solución. La definición de las convenciones utilizadas para la codificación, como *UpperCamelCase* y *lowerCamelCase* principalmente, posibilitó una mejor legibilidad del código, haciéndolo más comprensible y estandarizado. Además, la aplicación de las pruebas unitarias, de validación y de aceptación permitieron comprobar el correcto funcionamiento del código de la solución, validar la completitud de los requisitos y determinar la aceptación por del cliente.

14 CSS: Hojas de estilo en cascada, del inglés Cascading Style Sheets. Lenguaje utilizado para describir la presentación de documentos HTML o XML.

## Conclusiones Generales

A raíz de los resultados obtenidos y tomando como base el cumplimiento de los objetivos trazados al iniciar la investigación, es posible arribar a las siguientes conclusiones:

1. El estudio del *framework* de autenticación HTTP, la autenticación *proxy* y los esquemas de autenticación permitieron establecer las bases teóricas para el posterior desarrollo de la solución.
2. El estudio de las aplicaciones ProxyDroid, SandroProxy, Drony y UCIntlm posibilitó identificar la necesidad de desarrollar la Aplicación móvil para la autenticación con servidor *proxy* web en NovaDroid, así como un conjunto de funcionalidades que deberían ser incorporadas a la misma.
3. La identificación de 21 requisitos funcionales y 8 no funcionales, descritos a través de 14 historias de usuario, facilitó significativamente la posterior codificación de la solución.
4. La implementación de la Aplicación móvil para la autenticación con servidor *proxy* web en NovaDroid permitió darles respuesta a los requisitos definidos y de esa manera responder al problema de investigación planteado.
5. La ejecución de pruebas a la solución permitió la validación de todos los requisitos identificados y la aceptación por parte del cliente.

## Recomendaciones

Tomando como base la investigación realizada y el análisis de los resultados obtenidos se recomienda:

1. Agregar una funcionalidad para la configuración automática del *proxy* del sistema en el momento de iniciar el servicio de *proxy* local.
2. Agregar una funcionalidad para la restauración de la configuración del *proxy* del sistema en el momento de detener el servicio de *proxy* local.
3. Agregar un *widget* que permita iniciar el servicio de *proxy* local de manera más rápida y sin necesidad de abrir la aplicación.

## Bibliografía Referenciada


1. Android Studio Release Notes | Android Studio. [en línea], 2017. [Consulta: 26 noviembre 2017]. Disponible en: https://developer.android.com/studio/releases/index.html.
2. *android-architecture: A collection of samples to discuss and showcase different architectural tools and patterns for Android apps* [en línea], 2018. S.l.: Google Samples. [Consulta: 28 febrero 2018]. Disponible en: https://github.com/googlesamples/android-architecture.
3. Apache HttpComponents – Apache HttpComponents. [en línea], 2017. [Consulta: 29 octubre 2017]. Disponible en: https://hc.apache.org/.
4. Chapter 4. HTTP authentication. [en línea], 2017. [Consulta: 11 octubre 2017]. Disponible en: https://hc.apache.org/httpcomponents-client-4.2.x/tutorial/html/authentication.html.
5. COHN, M., 2018. User Stories and User Story Examples by Mike Cohn. *Mountain Goat Software* [en línea]. [Consulta: 26 febrero 2018]. Disponible en: https://www.mountaingoatsoftware.com/agile/user-stories.
6. *Drony* [en línea], 2017. S.l.: s.n. Disponible en: https://play.google.com/store/apps/details?id=org.sandroproxy.drony&hl=es_419.
7. DUNCAN, R., 2001. *An Overview of Different Authentication Methods and Protocols*. 2001. S.l.: s.n.
8. DYSON, P. y SHAFER, K., 2000. *Dictionary of Networking* [en línea]. S.l.: s.n. ISBN 0-7821-2461-5. Disponible en: http://een.iust.ac.ir/profs/Beheshti/Computer%20networking/Auxilary%20materials/Dictionary%20Of%20Network.pdf.
9. ERIKRE, 2017. Understanding HTTP Authentication. [en línea]. [Consulta: 11 octubre 2017]. Disponible en: https://docs.microsoft.com/en-us/dotnet/framework/wcf/feature-details/understanding-http-authentication.
10. ESCALONA, M.J. y KOCH, N., 2002. *Ingeniería de Requisitos en Aplicaciones para la Web– Un estudio comparativo*. 2002. S.l.: Departamento de Lenguajes y Sistemas Informáticos Escuela Técnica Superior de Ingeniería Informática Universidad de Sevilla.
11. EVANS, B.J. y FLANAGAN, D., 2014. *Java in a Nutshell*. S.l.: O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472. ISBN 978-1-4493-7082-4.

**12.** FARRELL, S., THOMAS, M. y HOFFMAN, P., 2015. HTTP Origin-Bound Authentication (HOBA). [en línea]. [Consulta: 28 noviembre 2017]. Disponible en: https://tools.ietf.org/html/rfc7486#page-3.

**13.** GAMMA, E., HELM, R., JOHNSON, R. y VLISSIDES, J., 1994. *Design Patterns : Elements of Reusable Object-Oriented Software* [en línea]. S.l.: Addison-Wesley. ISBN 0-201-63361-2. Disponible en: http://www.uml.org.cn/c++/pdf/DesignPatterns.pdf.

**14.** GLASS, E., 2006. *The NTLM Authentication Protocol and Security Support Provider* [en línea]. 2006. S.l.: s.n. Disponible en: http://davenport.sourceforge.net/ntlm.html#whatIsNtlm.

**15.** Google Java Style Guide. [en línea], nd. [Consulta: 31 marzo 2018]. Disponible en: https://google.github.io/styleguide/javaguide.html#s5.1-identifier-names.

**16.** GOURLEY, D. y TOTTY, B., 2002. *HTTP The Definitive Guide*. S.l.: O'Reilly. ISBN 1-56592-509-2.

**17.** *gradle: Adaptable, fast automation for all* [en línea], 2017. Java. S.l.: Gradle. [Consulta: 22 noviembre 2017]. Disponible en: https://github.com/gradle/gradle.

**18.** HARDT, D. y JONES, M., 2012. The OAuth 2.0 Authorization Framework: Bearer Token Usage. [en línea]. [Consulta: 28 noviembre 2017]. Disponible en: https://tools.ietf.org/html/rfc6750#page-2.

**19.** HTTP authentication. *Mozilla Developer Network* [en línea], 2017. [Consulta: 24 octubre 2017]. Disponible en: https://developer.mozilla.org/en-US/docs/Web/HTTP/Authentication.

**20.** ISO/IEC/IEEE 42010: Defining «architecture». [en línea], 2011. [Consulta: 28 febrero 2018]. Disponible en: http://www.iso-architecture.org/ieee-1471/defining-architecture.html.

**21.** LARMAN, C., 2004. *An Introduction to Object-Oriented Analysis and Design and Iterative Development* [en línea]. Third Edition. S.l.: Addison Wesley Professional. ISBN 0-13-148906-2. Disponible en: https://aanimesh.files.wordpress.com/2013/09/applying-uml-and-patterns-3rd.pdf.

**22.** LEIVA, A., 2014. MVP for Android: how to organize the presentation layer. *Antonio Leiva* [en línea]. [Consulta: 28 febrero 2018]. Disponible en: https://antonioleiva.com/mvp-android/.

**23.** LOELIGER, J. y MCCULLOUGH, M., 2012. *Version Control with Git*. S.l.: O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472. ISBN 978-1-4493-1638-9.

**24.** LV, M., 2016. *ProxyDroid* [en línea]. Java. S.l.: Max Lv. [Consulta: 24 octubre 2017]. Disponible en: https://play.google.com/store/apps/details?id=org.proxydroid&hl=es_419.

**25.** Microsoft NTLM (Windows). [en línea], 2017. [Consulta: 22 octubre 2017]. Disponible en: https://msdn.microsoft.com/en-us/library/windows/desktop/aa378749(v=vs.85).aspx.

**26.** Mobile and tablet internet usage exceeds desktop for first time worldwide. *StatCounter Global Stats* [en línea], 2016. [Consulta: 10 octubre 2017]. Disponible en: http://gs.statcounter.com/press/mobile-and-tablet-internet-usage-exceeds-desktop-for-first-time-worldwide.

**27.** PÉREZ VERA, L. y DÍAZ OROZCO, J.A., 2015. *Soporte de comunicación proxy para sistemas Android*. 2015. S.l.: Universidad de las Ciencias Informáticas.

**28.** PIERRA FUENTES, A., 2011. NOVA, DISTRIBUCIÓN CUBANA DE GNU/LINUX. REESTRUCTURACIÓN ESTRATÉGICA DE SU PROCESO DE DESARROLLO. ,

**29.** PINZÓN, S. y GUEVARA BOLAÑOS, J.C., 2006. *La gestión, los procesos y las metodologías de desarrollo de software*. 2006. S.l.: QDQDI1DQ.

**30.** PRESSMAN, R.S., 2010. *Software Engineering. A practitioner s Approach*. 7th. New York: McGraw Hill. ISBN 978-0-07-337597-7.

**31.** RAUCH, M., 2011. Mobile Documentation: Usability Guidelines, and Considerations for Providing Documentation on Kindle, Tablets, and Smartphones. *Professional Communication Conference (IPCC), 2011 IEEE International*, pp. 13. ISSN 2158-1002. DOI 10.1109/IPCC.2011.6087221.

**32.** Realm, el competidor de SQLite para los dispositivos móviles. | HumanOS. [en línea], 2016. [Consulta: 27 enero 2018]. Disponible en: http://humanos.uci.cu/2016/10/03/realm-el-competidor-de-sqlite-para-los-dispositivos-moviles/.

**33.** Realm Platform Documentation. [en línea], 2017. [Consulta: 20 noviembre 2017]. Disponible en: https://realm.io/docs/.

**34.** RESCHKE, J., 2015. The «Basic» HTTP Authentication Scheme. [en línea]. [Consulta: 28 noviembre 2017]. Disponible en: https://tools.ietf.org/html/rfc7617#page-3.

**35.** RODRÍGUEZ SÁNCHEZ, T., 2014. *Metodología de desarrollo para la Actividad productiva de la UCI.* 2014. S.l.: Universidad de las Ciencias Informáticas.

**36.** *SandroProxy* [en línea], 2016. S.l.: s.n. Disponible en: https://play.google.com/store/apps/details?id=org.sandroproxy&hl=es.

**37.** *SELECTING A DEVELOPMENT APPROACH*, 2005. 17 febrero 2005. S.l.: Centers for Medicare & Medical Services.

**38.** SOMMERVILLE, I., 2011. *Ingeniería del Software*. 9na. Mexico: Addison Wesley. Pearson Education, Inc. ISBN 978-607-32-0603-7.

**39.** squid : Optimising Web Delivery. [en línea], 2013. [Consulta: 27 abril 2018]. Disponible en: http://www.squid-cache.org/.

**40.** TINMEGALI, 2016. Model View Presenter (MVP) in Android,Part 1. *TinMegali* [en línea]. [Consulta: 28 febrero 2018]. Disponible en: https://medium.com/@tinmegali/model-view-presenter-mvp-in-android-part-1-441bfd7998fe.

**41.** Visual Paradigm Frequently Asked Questions. [en línea], 2017. [Consulta: 26 noviembre 2017]. Disponible en: https://www.visual-paradigm.com/support/faq.jsp.

**42.** What is Unified Modeling Language (UML)? [en línea], 2017. [Consulta: 29 octubre 2017]. Disponible en: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-uml/.

**43.** World Internet Users Statistics and 2017 World Population Stats. [en línea], 2017. [Consulta: 10 octubre 2017]. Disponible en: http://www.internetworldstats.com/stats.htm.

## Bibliografía Consultada


1. android - Given a HTTP request, is it possible to see which application originated it? - Stack Overflow. [en línea], 2013. [Consulta: 14 noviembre 2017]. Disponible en: https://stackoverflow.com/questions/14850445/given-a-http-request-is-it-possible-to-see-which-application-originated-it.
2. Android Studio Release Notes | Android Studio. [en línea], 2017. [Consulta: 26 noviembre 2017]. Disponible en: https://developer.android.com/studio/releases/index.html.
3. *android-architecture: A collection of samples to discuss and showcase different architectural tools and patterns for Android apps* [en línea], 2018. S.l.: Google Samples. [Consulta: 28 febrero 2018]. Disponible en: https://github.com/googlesamples/android-architecture.
4. Apache HttpComponents – Apache HttpComponents. [en línea], 2017. [Consulta: 29 octubre 2017]. Disponible en: https://hc.apache.org/.
5. Apache HttpComponents – NTLM support in HttpClient. [en línea], 2017. [Consulta: 14 noviembre 2017]. Disponible en: https://hc.apache.org/httpcomponents-client-ga/ntlm.html.
6. Chapter 4. HTTP authentication. [en línea], 2017. [Consulta: 11 octubre 2017]. Disponible en: https://hc.apache.org/httpcomponents-client-4.2.x/tutorial/html/authentication.html.
7. COHN, M., 2018. User Stories and User Story Examples by Mike Cohn. *Mountain Goat Software* [en línea]. [Consulta: 26 febrero 2018]. Disponible en: https://www.mountaingoatsoftware.com/agile/user-stories.
8. DARWIN, I.F., 2012. *Android Cookbook*. First Edition. S.l.: O'Reilly. ISBN 978-1-4493-8841-6.
9. *Drony* [en línea], 2017. S.l.: s.n. Disponible en: https://play.google.com/store/apps/details?id=org.sandroproxy.drony&hl=es_419.
10. DUNCAN, R., 2001. *An Overview of Different Authentication Methods and Protocols*. 2001. S.l.: s.n.
11. DYSON, P. y SHAFER, K., 2000. *Dictionary of Networking* [en línea]. S.l.: s.n. ISBN 0-7821-2461-5. Disponible en: http://een.iust.ac.ir/profs/Beheshti/Computer%20networking/Auxilary%20materials/Dictionary%20Of%20Network.pdf.
12. ERIKRE, 2017. Understanding HTTP Authentication. [en línea]. [Consulta: 28 noviembre 2017].

Disponible en: https://docs.microsoft.com/en-us/dotnet/framework/wcf/feature-details/understanding-http-authentication.

**13.** EVANS, B.J. y FLANAGAN, D., 2014. *Java in a Nutshell*. S.l.: O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472. ISBN 978-1-4493-7082-4.

**14.** FARRELL, S., THOMAS, M. y HOFFMAN, P., 2015. HTTP Origin-Bound Authentication (HOBA). [en línea]. [Consulta: 28 noviembre 2017]. Disponible en: https://tools.ietf.org/html/rfc7486#page-3.

**15.** FIELDING, R. y RESCHKE, J., 2014. Hypertext Transfer Protocol (HTTP/1.1): Authentication. [en línea]. [Consulta: 28 noviembre 2017]. Disponible en: https://tools.ietf.org/html/rfc7235.

**16.** FIGUEROA, R.G., SOLÍS, C.J. y CABRERA, A.A., 2008. *METODOLOGÍAS TRADICIONALES VS. METODOLOGÍAS ÁGILES*. 2008. S.l.: Universidad Técnica Particular de Loja, Escuela de Ciencias en Computación.

**17.** GAMMA, E., HELM, R., JOHNSON, R. y VLISSIDES, J., 1994. *Design Patterns : Elements of Reusable Object-Oriented Software* [en línea]. S.l.: Addison-Wesley. ISBN 0-201-63361-2. Disponible en: http://www.uml.org.cn/c++/pdf/DesignPatterns.pdf.

**18.** *git: Git Source Code Mirror - This is a publish-only repository and all pull requests are ignored. Please follow Documentation/SubmittingPatches procedure for any of your improvements* [en línea], 2017. C. S.l.: Git. [Consulta: 26 noviembre 2017]. Disponible en: https://github.com/git/git.

**19.** GLASS, E., 2006. *The NTLM Authentication Protocol and Security Support Provider* [en línea]. 2006. S.l.: s.n. Disponible en: http://davenport.sourceforge.net/ntlm.html#whatIsNtlm.

**20.** Google Java Style Guide. [en línea], nd. [Consulta: 31 marzo 2018]. Disponible en: https://google.github.io/styleguide/javaguide.html#s5.1-identifier-names.

**21.** Google Java Style Guide. [en línea], [sin fecha]. [Consulta: 31 marzo 2018]. Disponible en: https://google.github.io/styleguide/javaguide.html#s5.1-identifier-names.

**22.** GOTTARDO, S., 2016. *AndroidTCPSourceApp: Simple and performant Android library that extracts an application's basic information, given a Socket object* [en línea]. Java. S.l.: s.n. [Consulta: 14 noviembre 2017]. Disponible en: https://github.com/dextorer/AndroidTCPSourceApp.

**23.** GOURLEY, D. y TOTTY, B., 2002. *HTTP The Definitive Guide*. S.l.: O'Reilly. ISBN 1-56592-509-2.

**24.** *gradle: Adaptable, fast automation for all* [en línea], 2017. Java. S.l.: Gradle. [Consulta: 22 noviembre 2017]. Disponible en: https://github.com/gradle/gradle.

**25.** HARDT, D. y JONES, M., 2012. The OAuth 2.0 Authorization Framework: Bearer Token Usage. [en línea]. [Consulta: 28 noviembre 2017]. Disponible en: https://tools.ietf.org/html/rfc6750#page-2.

**26.** HGRACA, 2017. Architectural Styles vs. Architectural Patterns vs. Design Patterns. *@herbertograca* [en línea]. [Consulta: 22 marzo 2018]. Disponible en: https://herbertograca.com/2017/07/28/architectural-styles-vs-architectural-patterns-vs-design-patterns/.

**27.** HIMANSHU, 05:43:01 UTC. Architectural styles and patterns. [en línea]. Tecnología. S.l. [Consulta: 22 marzo 2018]. Disponible en: https://es.slideshare.net/himanshuhora/architectural-styles-and-patterns-24377255?from_action=save.

**28.** How to get a list of installed android applications and pick one to run - Stack Overflow. [en línea], 2010. [Consulta: 14 noviembre 2017]. Disponible en: https://stackoverflow.com/questions/2695746/how-to-get-a-list-of-installed-android-applications-and-pick-one-to-run/5097838.

**29.** HTTP Authentication | HttpWatch. [en línea], 2017. [Consulta: 23 octubre 2017]. Disponible en: https://www.httpwatch.com/httpgallery/authentication/.

**30.** HTTP authentication. *Mozilla Developer Network* [en línea], 2017. [Consulta: 24 octubre 2017]. Disponible en: https://developer.mozilla.org/en-US/docs/Web/HTTP/Authentication.

**31.** HttpClient - HttpClient Authentication Guide. [en línea], [sin fecha]. [Consulta: 31 enero 2018]. Disponible en: http://hc.apache.org/httpclient-3.x/authentication.html.

**32.** IBM Knowledge Center. [en línea], [sin fecha]. [Consulta: 31 marzo 2018]. Disponible en: https://www.ibm.com/support/knowledgecenter/SS8PJ7_9.5.0/com.ibm.xtools.modeler.doc/topics/cdepd.html.

**33.** ISO/IEC/IEEE 42010: Defining «architecture». [en línea], 2011. [Consulta: 28 febrero 2018]. Disponible en: http://www.iso-architecture.org/ieee-1471/defining-architecture.html.

**34.** LARMAN, C., 2004. *An Introduction to Object-Oriented Analysis and Design and Iterative Development* [en línea]. Third Edition. S.l.: Addison Wesley Professional. ISBN 0-13-148906-2. Disponible en: https://aanimesh.files.wordpress.com/2013/09/applying-uml-and-patterns-3rd.pdf.

**35.** LEACH, P.J., BERNERS-LEE, T., MOGUL, J.C., MASINTER, L., FIELDING, R.T. y GETTYS, J., 1999. Hypertext Transfer Protocol -- HTTP/1.1. [en línea]. [Consulta: 12 octubre 2017]. Disponible

en: https://tools.ietf.org/html/rfc2616#page-7.

**36.** LEACH, P.J., FRANKS, J., LUOTONEN, A., HALLAM-BAKER, P.M., LAWRENCE, S.D., HOSTETLER, J.L. y STEWART, L.C., 1999. HTTP Authentication: Basic and Digest Access Authentication. [en línea]. [Consulta: 28 noviembre 2017]. Disponible en: https://tools.ietf.org/html/rfc2617.

**37.** LEIVA, A., 2014. MVP for Android: how to organize the presentation layer. *Antonio Leiva* [en línea]. [Consulta: 28 febrero 2018]. Disponible en: https://antonioleiva.com/mvp-android/.

**38.** LOELIGER, J. y MCCULLOUGH, M., 2012. *Version Control with Git*. S.l.: O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472. ISBN 978-1-4493-1638-9.

**39.** LONELYDEVELOPER97, 2017. *android-proxy-changing: Small android library that allows you to change wifi proxy settings programmatically.* [en línea]. Java. S.l.: s.n. [Consulta: 15 noviembre 2017]. Disponible en: https://github.com/lonelydeveloper97/android-proxy-changing.

**40.** LV, M., 2016. *ProxyDroid* [en línea]. Java. S.l.: Max Lv. [Consulta: 24 octubre 2017]. Disponible en: https://play.google.com/store/apps/details?id=org.proxydroid&hl=es_419.

**41.** Microsoft NTLM (Windows). [en línea], 2017. [Consulta: 22 octubre 2017]. Disponible en: https://msdn.microsoft.com/en-us/library/windows/desktop/aa378749(v=vs.85).aspx.

**42.** MITCHELL, B., 2016. A Beginner's Guide to Using Proxy Servers in Computer Networking. *Lifewire* [en línea]. [Consulta: 21 octubre 2017]. Disponible en: https://www.lifewire.com/introduction-to-proxy-servers-computer-networking-816448.

**43.** Mobile and tablet internet usage exceeds desktop for first time worldwide. *StatCounter Global Stats* [en línea], 2016. [Consulta: 10 octubre 2017]. Disponible en: http://gs.statcounter.com/press/mobile-and-tablet-internet-usage-exceeds-desktop-for-first-time-worldwide.

**44.** Mobile OS market share. *Statista* [en línea], 2017. [Consulta: 10 octubre 2017]. Disponible en: https://www.statista.com/statistics/266136/global-market-share-held-by-smartphone-operating-systems/.

**45.** Nova Droid | Portal web para Nova. [en línea], 2017. [Consulta: 30 noviembre 2017]. Disponible en: https://www.nova.cu/es/productos/nova-droid.

**46.** Optimizing NTLM authentication flow in multi-domain environments – Israel Platforms PFE Team. [en línea], 2010. [Consulta: 31 enero 2018]. Disponible en:

https://blogs.technet.microsoft.com/isrpfeplat/2010/11/05/optimizing-ntlm-authentication-flow-in-multi-domain-environments/.

**47.** OROZCO, J.A.D., 2016. *UCIntlm: NTLM proxy for android (no root required)* [en línea]. Java. S.l.: s.n. [Consulta: 29 noviembre 2017]. Disponible en: https://github.com/jadolg/UCIntlm.

**48.** PÉREZ VERA, P.V. y DÍAZ OROZCO, J.A., 2015. *Soporte de comunicación proxy para sistemas Android*. S.l.: Universidad de las Ciencias Informáticas.

**49.** PIERRA FUENTES, P.F., 2011. *NOVA, DISTRIBUCIÓN CUBANA DE GNU/LINUX. REESTRUCTURACIÓN ESTRATÉGICA DE SU PROCESO DE DESARROLLO*. S.l.: Universidad de las Ciencias Informáticas.

**50.** PINZÓN, S. y GUEVARA BOLAÑOS, J.C., 2006. *La gestión, los procesos y las metodologías de desarrollo de software*. 2006. S.l.: QDQDI1DQ.

**51.** PRESSMAN, R.S., 2010. *Software Engineering. A practitioner s Approach*. 7th. New York: McGraw Hill. ISBN 978-0-07-337597-7.

**52.** PRUSTY, N., 2014. How Does HTTP Authentication Work? *QNimate* [en línea]. [Consulta: 21 octubre 2017]. Disponible en: http://qnimate.com/understanding-http-authentication-in-depth/.

**53.** RAUCH, M., 2011. Mobile Documentation: Usability Guidelines, and Considerations for Providing Documentation on Kindle, Tablets, and Smartphones. *Professional Communication Conference (IPCC), 2011 IEEE International*, pp. 13. ISSN 2158-1002. DOI 10.1109/IPCC.2011.6087221.

**54.** Realm, el competidor de SQLite para los dispositivos móviles. | HumanOS. [en línea], 2016. [Consulta: 27 enero 2018]. Disponible en: http://humanos.uci.cu/2016/10/03/realm-el-competidor-de-sqlite-para-los-dispositivos-moviles/.

**55.** Realm Platform Documentation. [en línea], 2017. [Consulta: 20 noviembre 2017]. Disponible en: https://realm.io/docs/.

**56.** RESCHKE, J., 2015. The «Basic» HTTP Authentication Scheme. [en línea]. [Consulta: 28 noviembre 2017]. Disponible en: https://tools.ietf.org/html/rfc7617#page-3.

**57.** RODRÍGUEZ SÁNCHEZ, T., 2014. *Metodología de desarrollo para la Actividad productiva de la UCI.* 2014. S.l.: Universidad de las Ciencias Informáticas.

**58.** SAINI, K., 2011. *Squid Proxy Server 3.1*. S.l.: Packt Publishing Ltd. ISBN 978-1-84951-390-6.

**59.** *SandroProxy* [en línea], 2016. S.l.: s.n. Disponible en: https://play.google.com/store/apps/details?

id=org.sandroproxy&hl=es.

**60.** *SELECTING A DEVELOPMENT APPROACH*, 2005. 17 febrero 2005. S.l.: Centers for Medicare & Medical Services.

**61.** SHEKH-YUSEF, R., AHRENS, D. y BREMER, S., 2015. HTTP Digest Access Authentication. [en línea]. [Consulta: 28 noviembre 2017]. Disponible en: https://tools.ietf.org/html/rfc7616#page-4.

**62.** SHINDER, D., 2001. Understanding and selecting authentication methods. *TechRepublic* [en línea]. [Consulta: 22 octubre 2017]. Disponible en: https://www.techrepublic.com/article/understanding-and-selecting-authentication-methods/.

**63.** SOMMERVILLE, I., 2011. *Ingeniería del Software*. 9na. Mexico: Addison Wesley. Pearson Education, Inc. ISBN 978-607-32-0603-7.

**64.** SUPPSANDROB, 2017. *SandroProxy* [en línea]. JavaScript. S.l.: s.n. [Consulta: 24 octubre 2017]. Disponible en: https://github.com/SuppSandroB/sandrop.

**65.** TINMEGALI, 2016. Model View Presenter (MVP) in Android,Part 1. *TinMegali* [en línea]. [Consulta: 28 febrero 2018]. Disponible en: https://medium.com/@tinmegali/model-view-presenter-mvp-in-android-part-1-441bfd7998fe.

**66.** VINCENZO, A. y GENOVEFFA, T., 1993. *Advances In Software Engineering And Knowledge Engineering*. S.l.: World Scientific. ISBN 978-981-4502-57-3.

**67.** Visual Paradigm Frequently Asked Questions. [en línea], 2017. [Consulta: 26 noviembre 2017]. Disponible en: https://www.visual-paradigm.com/support/faq.jsp.

**68.** What is Unified Modeling Language (UML)? [en línea], 2017. [Consulta: 29 octubre 2017]. Disponible en: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-uml/.

**69.** World Internet Users Statistics and 2017 World Population Stats. [en línea], 2017. [Consulta: 10 octubre 2017]. Disponible en: http://www.internetworldstats.com/stats.htm.

# Anexos

**Anexo 1. Entrevista realizada a especialistas sobre los esquemas de autenticación más utilizados en los servidores *proxy* de las instituciones cubanas.**

Día ___ de _____ del año 2018. Universidad de las Ciencias Informáticas, Carretera a San Antonio de los Baños, km 2 ½, Boyeros, La Habana, Cuba.

Entrevistado: _____.
Cargo del entrevistado: _____.
Entrevistador: _____.

Los esquemas de autenticación son utilizados en los servidores *proxy* con el objetivo de realizar el control de acceso de los usuarios de la red. Estos proporcionan una manera de recoger credenciales, determinar la identidad del usuario y decidir si tiene o no acceso a determinado contenido. Basándose en su experiencia, y teniendo en cuenta las condiciones tecnológicas en Cuba, cuáles considera usted que sean los esquemas de autenticación más utilizados en los servidores *proxy* desplegados en las instituciones cubanas.

**Anexo 2. HU Gestionar regla de cortafuegos.**

| **HU: Gestionar regla de cortafuegos** | |
|---|---|
| **Número:** 4 | **Nombre del requisito:** Adicionar regla de cortafuegos, Mostrar regla de cortafuegos, Modificar regla de cortafuegos y Eliminar regla de cortafuegos |
| **Programador:** Daniel A. Rodríguez Caballero | **Iteración Asignada:** 2 |
| **Prioridad:** Media | **Tiempo Estimado:** 12 días |

| **Riesgo en Desarrollo:** N/A | **Tiempo Real:** 10 días. |
| --- | --- |

**Descripción**

Se debe permitir que el usuario adicione, modifique, visualice y elimine las reglas de cortafuegos Estos deben poseer los siguientes datos:

1. Aplicación a la que se le aplicará la regla.
2. Regla a aplicar.
3. Descripción de la regla.

**Adicionar regla de cortafuegos:**

Se debe mostrar un campo de texto por cada dato que posee una regla de cortafuegos en orden de que el usuario los complete según sus necesidades y además un botón para completar la acción de adicionar.

Se deben realizar las siguientes validaciones:

1. El campo de la regla a aplicar no debe estar vacío.

Se podrá acceder a la funcionalidad desde el listado de reglas de cortafuegos definido por la HU “Listar reglas de cortafuegos” donde habrá un botón indicando la acción de crear una nueva regla, el cual accederá a la funcionalidad.

**Mostrar regla de cortafuegos:**

Se debe permitir que el usuario visualice todos los datos de una regla de cortafuegos previamente creada. Debe mostrarse un campo de texto no editable por cada dato de la regla de cortafuegos. Es necesario además mostrar un botón para la funcionalidad de eliminar la regla de cortafuegos y otro para modificarla.

La funcionalidad será accesible desde el listado de reglas de cortafuegos, donde al seleccionar una de ellas se ejecutará esta funcionalidad.

**Modificar regla de cortafuegos:**

Se debe permitir que el usuario modifique una regla de cortafuegos previamente creada. Debe mostrarse un campo texto por cada dato y completarlos con la información que ya posee, estos campos deben ser editables y deben realizarse las mismas validaciones que en el caso de “Adicionar

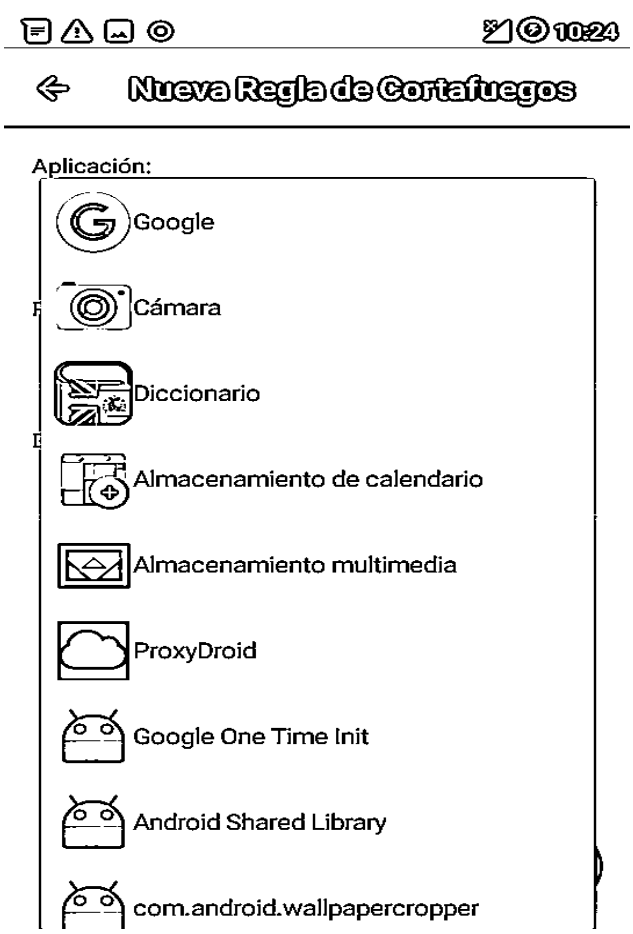
regla de cortafuegos”. Debe mostrarse además un botón para completar la acción de modificar.

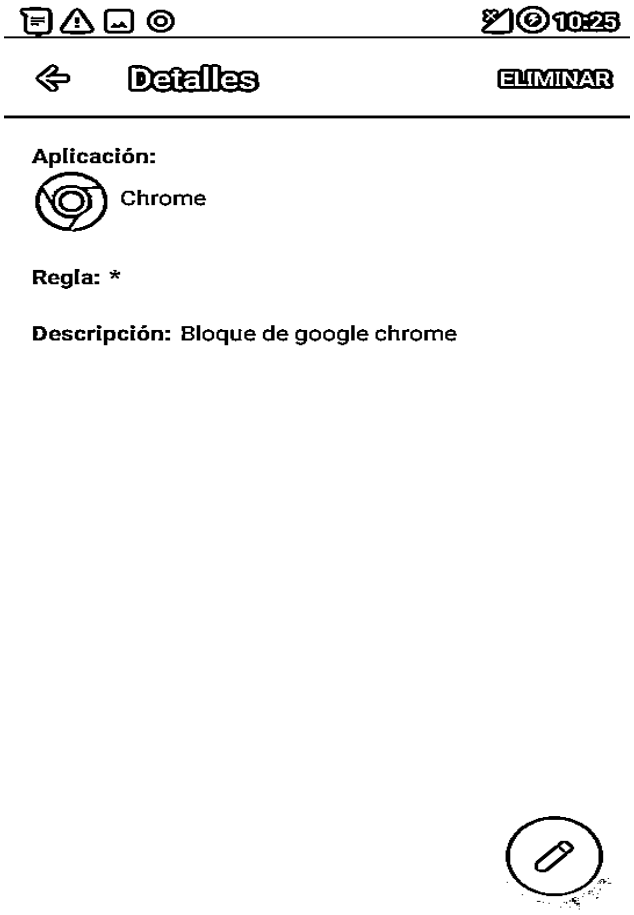
**Eliminar regla de cortafuegos:**

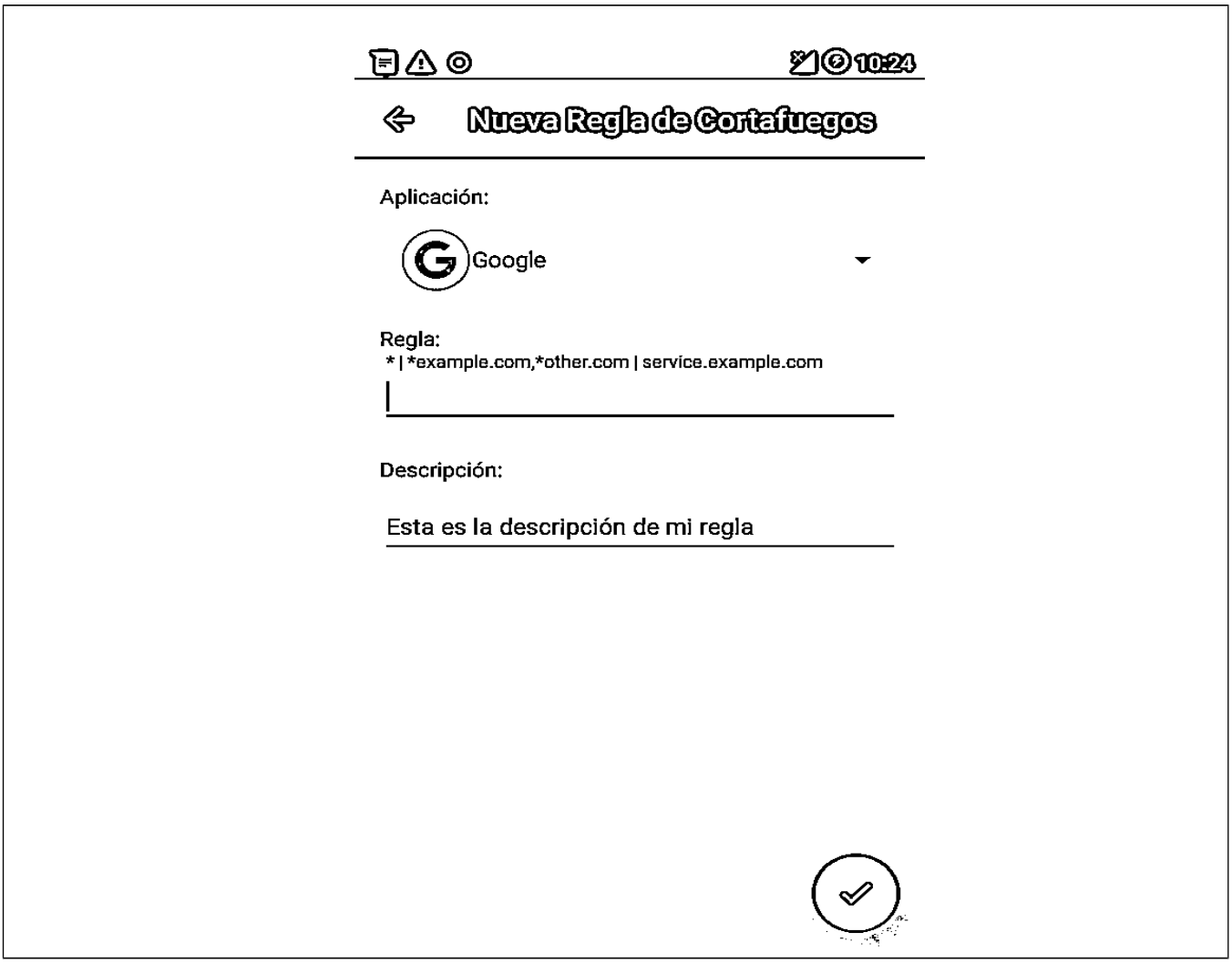
Se debe permitir que el usuario elimine una regla de cortafuegos previamente creada. La acción estará disponible desde un botón la vista de “Mostrar regla de cortafuegos” donde al presionarlo se elimina por completo la regla que se estaba mostrando.

**Observaciones:**

**Prototipo elemental de interfaz gráfica de usuario:**






**Anexo 3. HU Listar perfiles de conexión.**

| **HU: Listar perfiles de conexión** | |
|---|---|
| **Número:** 5 | **Nombre del requisito:** Listar perfiles de conexión |
| **Programador:** Daniel A. Rodríguez Caballero | **Iteración Asignada:** 1 |
| **Prioridad:** Alta | **Tiempo Estimado:** 3 días |

| **Riesgo en Desarrollo:** N/A | **Tiempo Real:** 3 días |
|---|---|

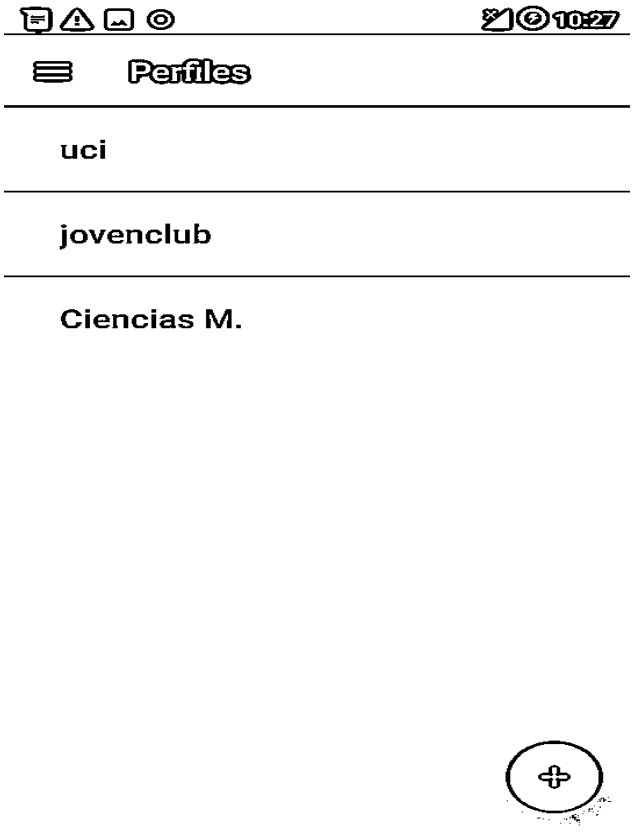
**Descripción:**

Se debe mostrar al usuario el listado de los perfiles de conexión creados hasta el momento, mostrando por cada perfil solamente el nombre. Se debe mostrar además un botón flotante que indique la funcionalidad de adicionar un nuevo perfil de conexión.

La funcionalidad debe ser accedida desde el menú lateral de la aplicación con el nombre “Perfiles”.

**Observaciones:**

**Prototipo elemental de interfaz gráfica de usuario:**



## Anexo 4. HU Listar reglas de cortafuegos.

| **HU: Listar reglas de cortafuegos** | |
|---|---|
| **Número:** 6 | **Nombre del requisito:** Listar reglas de cortafuegos |
| **Programador:** Daniel A. Rodríguez Caballero | **Iteración Asignada:** 2 |
| **Prioridad:** Media | **Tiempo Estimado:** 3 días |
| **Riesgo en Desarrollo:** N/A | **Tiempo Real:** 3 días |

**Descripción:**

Se debe mostrar a usuario el listado de las reglas de cortafuegos creadas hasta el momento, mostrando por cada regla los siguientes datos:

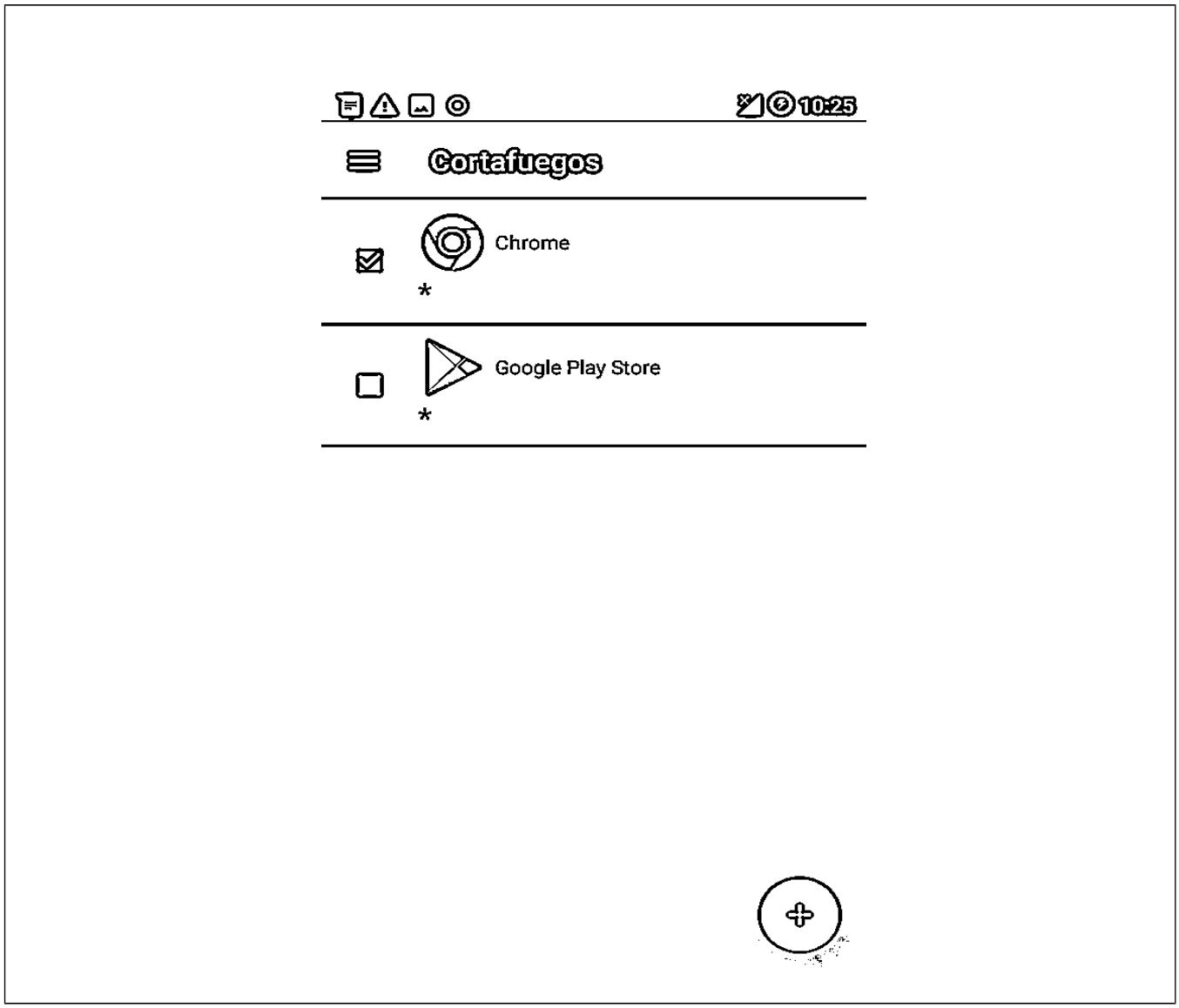
1. Icono de la aplicación a la que se le aplica la regla.
2. Nombre de la aplicación a la que se le aplica la regla.
3. Regla a aplicar.

Se debe mostrar además un botón flotante que indique la funcionalidad de adicionar una nueva regla de cortafuegos.

La funcionalidad debe ser accedida desde el menú lateral de la aplicación con el nombre "Cortafuegos".

**Observaciones:**

**Prototipo de interfaz gráfica de usuario**




**Anexo 5. HU Detener servicio de *proxy* local.**

| **HU: Detener servicio de *proxy* local** | |
|---|---|
| **Número:** 7 | **Nombre del requisito:** Detener servicio de *proxy* local |
| **Programador:** Daniel A. Rodríguez Caballero | **Iteración Asignada:** 1 |

| | |
|---|---|
| **Prioridad:** Alta | **Tiempo Estimado:** 2 días |
| **Riesgo en Desarrollo:** N/A | **Tiempo Real:** 2 días |

**Descripción:**

Se debe permitir que el usuario detenga el servicio de *proxy* local.

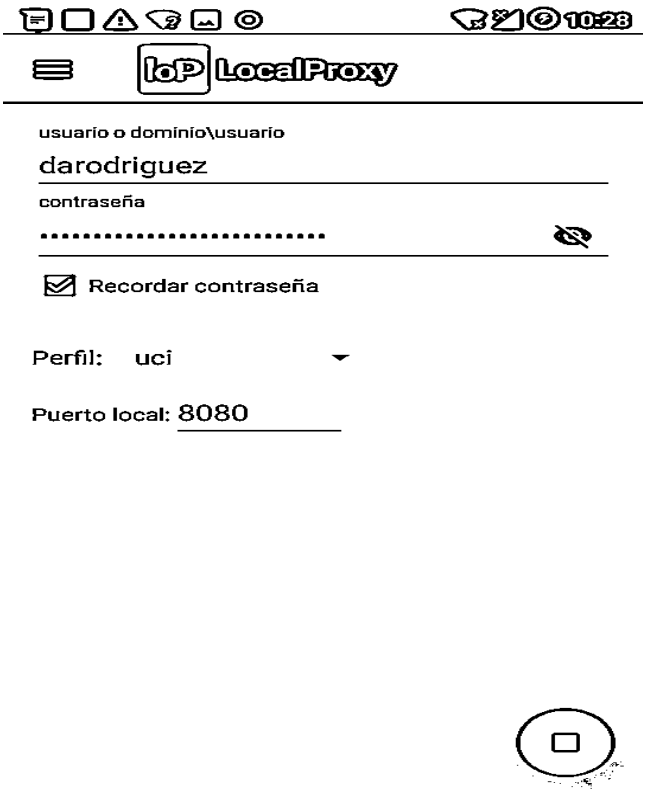
Para su ejecución debe haberse iniciado previamente el servicio de *proxy* local.

La funcionalidad tiene lugar en la vista inicial de la aplicación junto con la funcionalidad de “Autenticar usuario”. Debe mostrarse un botón que indique la acción de detener el servicio.

Luego de ejecutar la funcionalidad se debe remplazar el botón flotante que indica la acción de detener por un botón flotante que indique la acción de iniciar el servicio, esto relativo a la historia de usuario ”Iniciar servicio de *proxy* local”.

**Observaciones:**

**Prototipo elemental de interfaz gráfica de usuario:**



## Anexo 6. HU Restablecer configuración.

| **HU: Restablecer configuración** | |
|---|---|
| **Número:** 8 | **Nombre del requisito:** Restablecer configuración |
| **Programador:** Daniel A. Rodríguez Caballero | **Iteración Asignada:** 1 |
| **Prioridad:** Alta | **Tiempo Estimado:** 2 días |
| **Riesgo en Desarrollo:** N/A | **Tiempo Real:** 1 día |
| **Descripción:**<br>Al iniciarse la aplicación se debe restablecer la última configuración con la que se ejecutó el servicio de *proxy* local. Los campos a restablecer son:<br>**1.** Perfil de conexión seleccionado<br>**2.** Puerto local de escucha<br>**3.** Nombre de usuario<br>**4.** Contraseña (si fue marcada la opción “Recordar contraseña”). | |
| **Observaciones:** | |
| **Prototipo elemental de interfaz gráfica de usuario:** | |

## Anexo 7. HU Activar o desactivar regla de cortafuegos.

| **HU: Activar o desactivar regla de cortafuegos** | |
|---|---|
| **Número:** 9 | **Nombre del requisito:** Activar regla de cortafuegos, Desactivar regla de cortafuegos |
| **Programador:** Daniel A. Rodríguez Caballero | **Iteración Asignada:** 2 |
| **Prioridad:** Baja | **Tiempo Estimado:** 8 días |
| **Riesgo en Desarrollo:** N/A | **Tiempo Real:** 10 días |

**Descripción:**

Se deberá permitir al usuario que active o desactive determinada regla de cortafuegos. Se debe mostrar un checkbox al lado de cada regla de cortafuegos en el listado de las reglas de cortafuegos. La funcionalidad tiene lugar al marcar o desmarcar el checkbox mostrado, donde el servicio de *proxy* local debe tener o no tener en cuenta la regla para realizar el filtrado de las peticiones.

**Observaciones:**

**Prototipo elemental de interfaz gráfica de usuario:**

## Anexo 8. HU Adicionar traza de navegación.

| **HU: Adicionar traza de navegación** | |
|---|---|
| **Número:** 10 | **Nombre del requisito:** Adicionar traza de navegación |
| **Programador:** Daniel A. Rodríguez Caballero | **Iteración Asignada:** 3 |
| **Prioridad:** Media | **Tiempo Estimado:** 4 días |
| **Riesgo en Desarrollo:** N/A | **Tiempo Real:** 5 días |

**Descripción:**

Se deberá crear una traza de navegación por cada petición realizada en el dispositivo. Los datos a almacenar por cada traza son:

1. Nombre de la aplicación que realizó la petición
2. Fecha de la petición.
3. Consumo en bytes de la petición.
4. Dirección URL que se requirió

La funcionalidad es llevada a cabo por la aplicación luego de resolverse una petición por alguna aplicación del dispositivo.

**Observaciones:**

**Prototipo elemental de interfaz gráfica de usuario:**

## Anexo 9. HU Listar trazas de navegación.

| **HU: Listar trazas de navegación** | |
|---|---|
| **Número:** 11 | **Nombre del requisito:** Listar trazas de navegación |
| **Programador:** Daniel A. Rodríguez Caballero | **Iteración Asignada:** 3 |
| **Prioridad:** Media | **Tiempo Estimado:** 3 días |
| **Riesgo en Desarrollo:** N/A | **Tiempo Real:** 3 días |

**Descripción:**

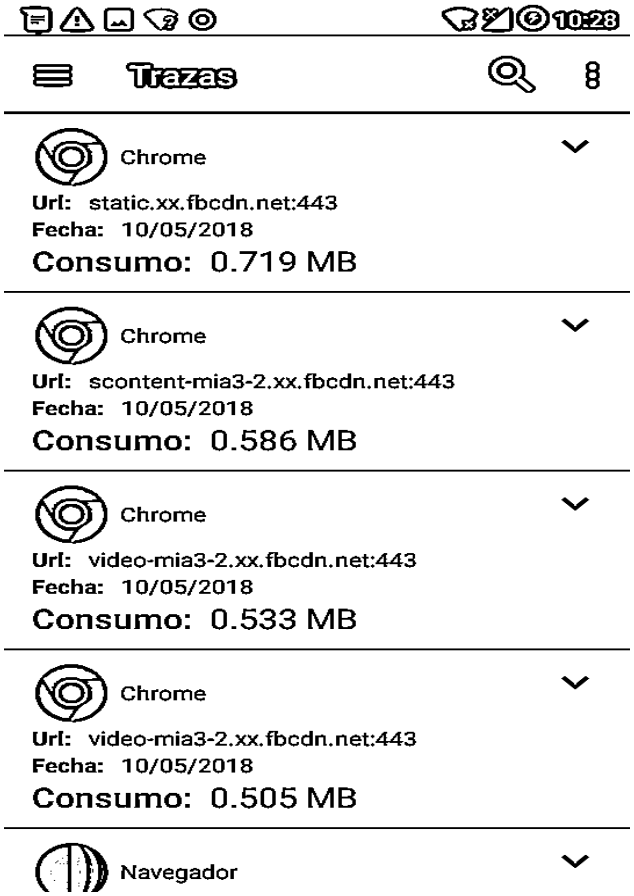
Se deberá mostrar un listado con las trazas de navegación creadas por cada petición. Los datos a mostrar por cada traza son:

1. Icono de la aplicación que realizó la petición.
2. Nombre de la aplicación que realizó la petición.
3. URL que se requirió en la petición.
4. Fecha de realización de la petición.
5. Consumo de la petición en MB.

La funcionalidad tiene ligar al seleccionar la opción de “Trazas” en el menú lateral de la aplicación.

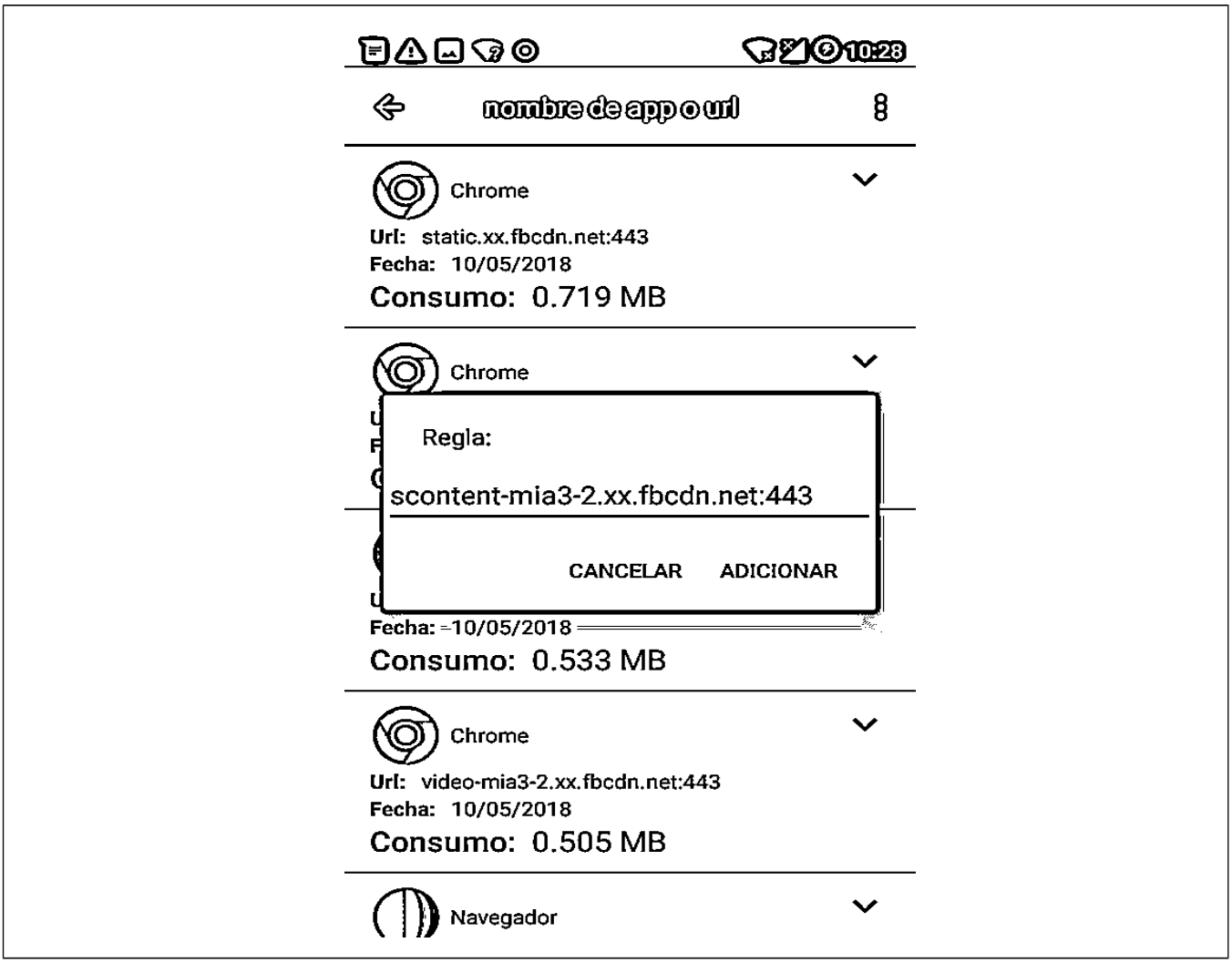
**Observaciones:**

**Prototipo elemental de interfaz gráfica de usuario:**



## Anexo 10. HU Añadir traza de navegación como regla de cortafuegos.

| **HU: Añadir traza de navegación como regla de cortafuegos** | |
|---|---|
| **Número:** 12 | **Nombre del requisito:** Añadir traza de navegación como regla de cortafuegos |
| **Programador:** Daniel A. Rodríguez Caballero | **Iteración Asignada:** 3 |
| **Prioridad:** Media | **Tiempo Estimado:** 3 días |
| **Riesgo en Desarrollo:** N/A | **Tiempo Real:** 3 días |
| **Descripción:**<br>Se deberá permitir al usuario añadir una traza de navegación como regla de cortafuegos. Se debe tener en cuenta el nombre de la aplicación que realizó la petición y la URL que solicitó.<br>La funcionalidad tiene lugar al realizar una pulsación extendida sobre una traza de navegación en el listado de trazas, luego se mostrará un diálogo con el objetivo de que el usuario modifique la URL de la petición a su gusto y la adapte a la regla que desea aplicar. | |
| **Observaciones:** | |
| **Prototipo elemental de interfaz gráfica de usuario:** | |




**Anexo 11. HU Buscar trazas de navegación.**

| **HU: Buscar trazas de navegación** | |
|---|---|
| **Número:** 13 | **Nombre del requisito:** Buscar trazas de navegación |
| **Programador:** Daniel A. Rodríguez Caballero | **Iteración Asignada:** 3 |
| **Prioridad:** Media | **Tiempo Estimado:** 2 días |

| **Riesgo en Desarrollo:** N/A | **Tiempo Real:** 3 días |
|---|---|

**Descripción:**

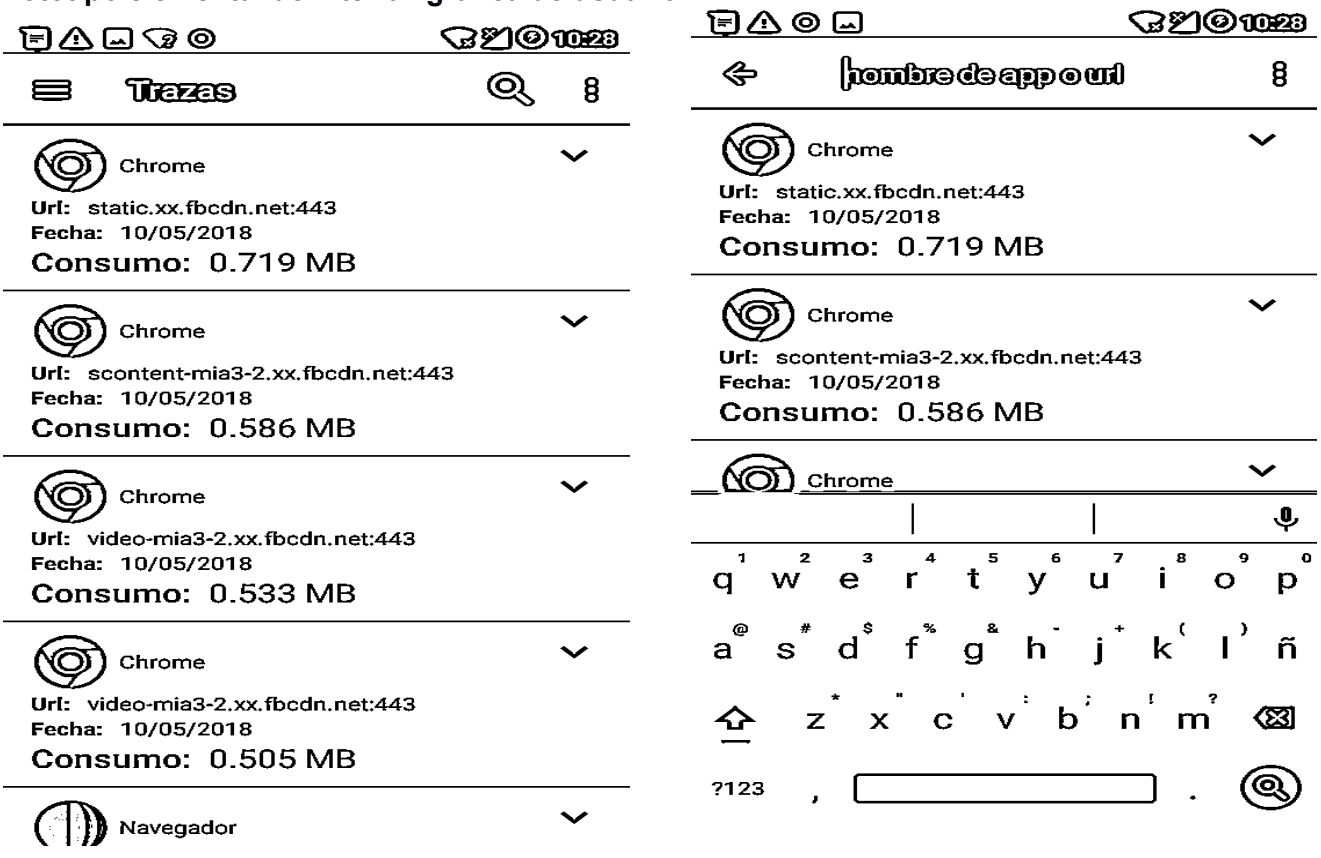
Se deberá permitir al usuario buscar trazas de navegación a partir de un criterio de búsqueda insertado por este. Para realizar la búsqueda se tienen en cuenta los campos de:

- Nombre de la aplicación que realizó la petición.
- URL que se requirió en la petición.

La funcionalidad tiene lugar en la barra de herramientas de la vista del listado de trazas de navegación donde se muestra un botón que sugiere la opción de buscar. Los resultados de la búsqueda se mostrarán en el mismo listado de trazas. Deberá poseer además un botón que indique la acción de cancelar la búsqueda donde al presionarlo se vuelven a mostrar todas las trazas creadas.

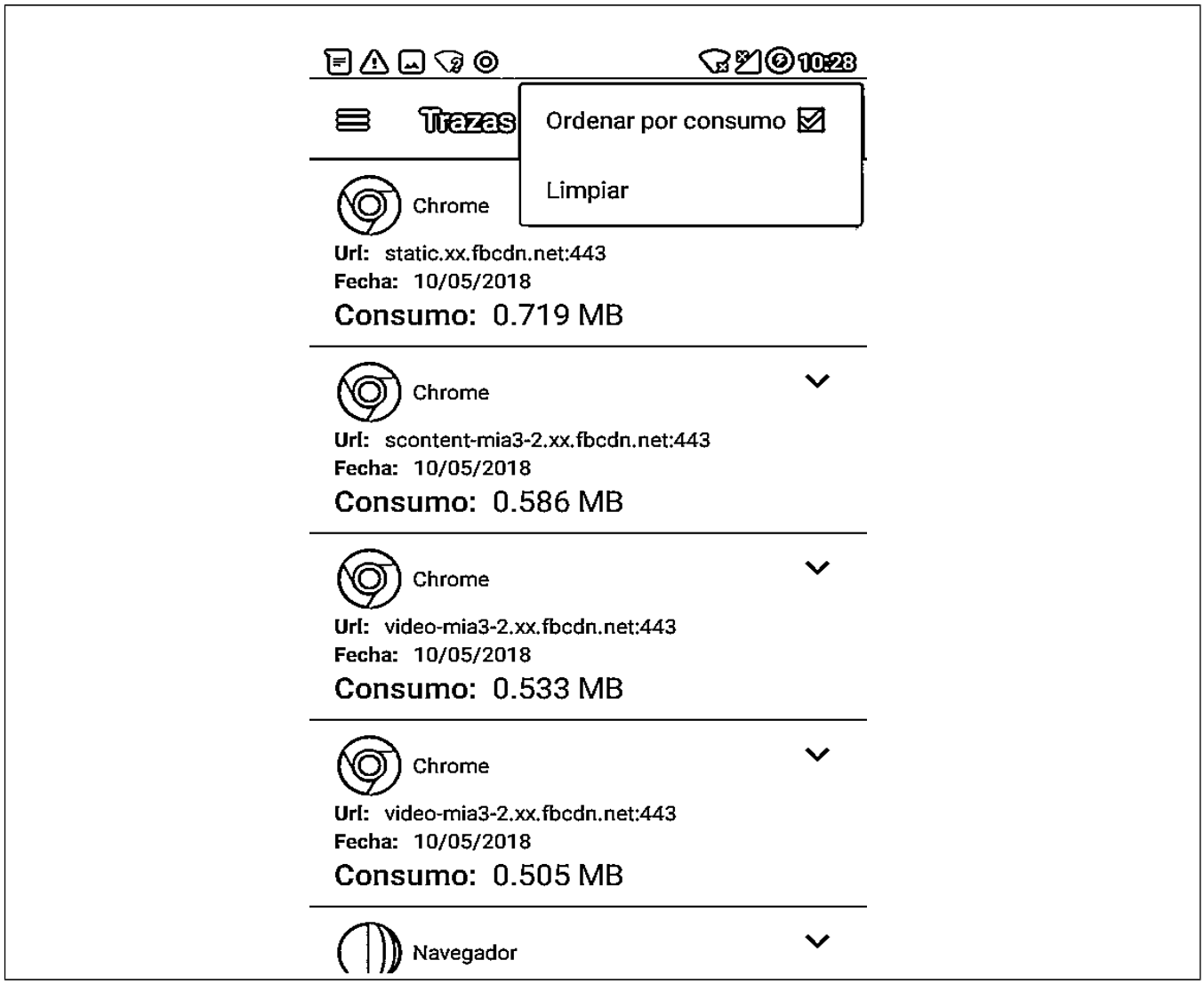
**Observaciones:**

**Prototipo elemental de interfaz gráfica de usuario:**



**Anexo 12. HU Ordenar trazas de navegación.**

| **HU: Ordenar trazas de navegación** | |
|---|---|
| **Número:** 14 | **Nombre del requisito:** Ordenar trazas de navegación |
| **Programador:** Daniel A. Rodríguez Caballero | **Iteración Asignada:** 3 |
| **Prioridad:** Media | **Tiempo Estimado:** 1 día |
| **Riesgo en Desarrollo:** N/A | **Tiempo Real:** 1 día |
| **Descripción:**<br>Se deberá permitir al usuario ordenar el listado de las trazas de navegación atendiendo al criterio de cantidad *bytes* consumidos por la petición. La funcionalidad tiene lugar en el menú de la barra de herramientas de la vista del listado de trazas de navegación. Se muestra un checkbox que indica la acción, este, al activarse se ordenan las trazas atendiendo al consumo en bytes y se muestran los resultados en el listado de la vista. Al desactivarse se muestran las trazas en su orden por defecto. | |
| **Observaciones:** | |
| **Prototipo elemental de interfaz gráfica de usuario:** | |


10:28
Trazas
Ordenar por consumo
Limpiar
Chrome
Url: static.xx.fbcdn.net:443
Fecha: 10/05/2018
Consumo: 0.719 MB
Chrome
Url: scontent-mia3-2.xx.fbcdn.net:443
Fecha: 10/05/2018
Consumo: 0.586 MB
Chrome
Url: video-mia3-2.xx.fbcdn.net:443
Fecha: 10/05/2018
Consumo: 0.533 MB
Chrome
Url: video-mia3-2.xx.fbcdn.net:443
Fecha: 10/05/2018
Consumo: 0.505 MB
Navegador

## Anexo 13. Fragmento del diagrama de clases del diseño.

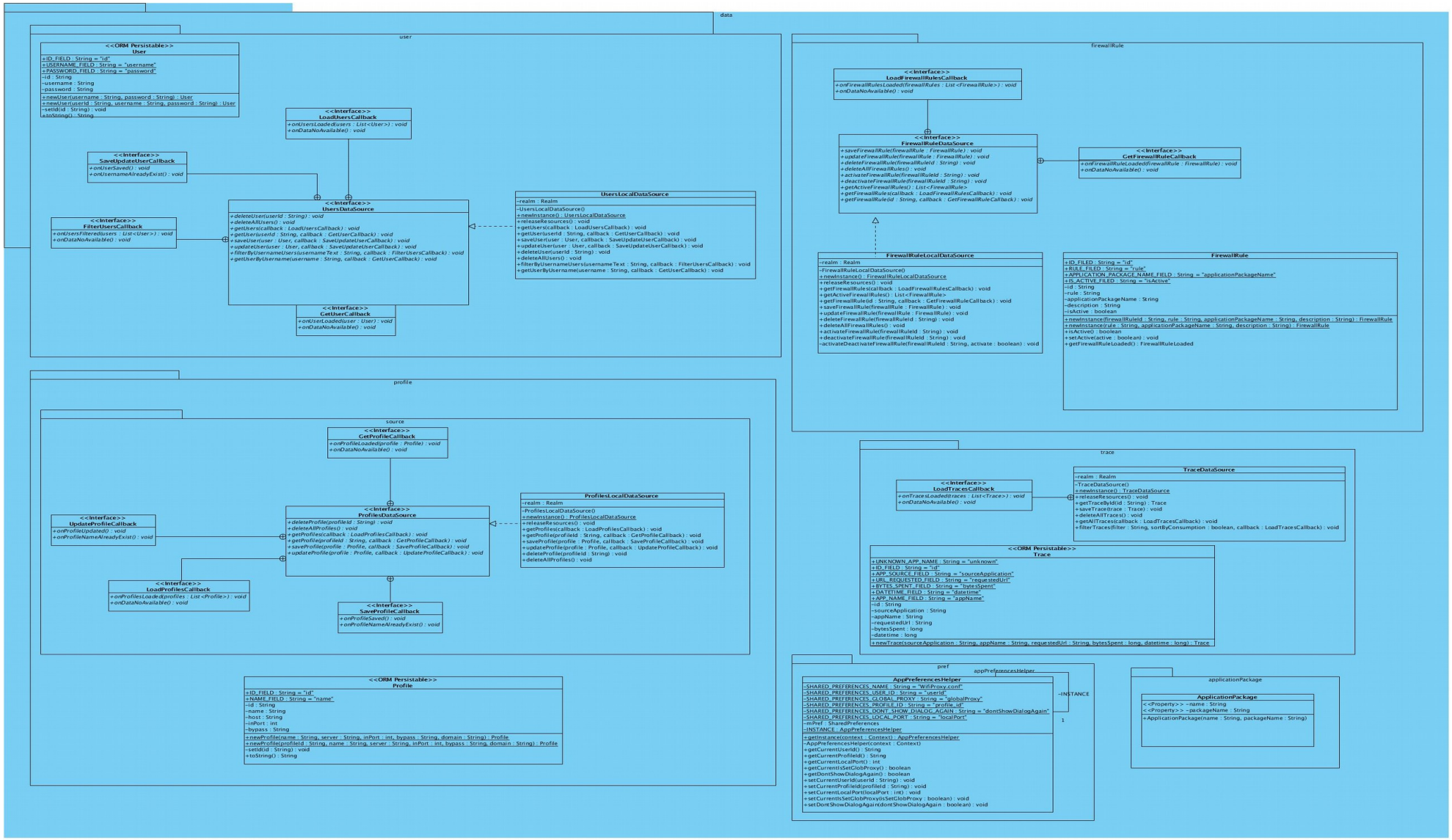



*Figura 19: Fragmento del diagrama de clases del diseño relativo a la persistencia y lógica de acceso a datos.*

## Anexo 14. Caso de Prueba para la HU Gestionar regla de cortafuegos.

| **Descripción general** | | | | | | |
|---|---|---|---|---|---|---|
| El caso de prueba inicia cuando el usuario selecciona la opción de *Cortafuegos* en el menú lateral de la aplicación. Se presenta entonces al usuario el listado de las reglas de cortafuegos creadas hasta el momento y además la opción de adicionar una nueva regla. El caso de prueba termina cuando el usuario abandona la sección de Cortafuegos. | | | | | | |
| **Condiciones de ejecución** | | | | | | |
| Condiciones para SC2, SC3, SC4: debe existir al menos una regla de cortafuegos creada previamente. | | | | | | |
| **SC1 Adicionar regla de cortafuegos** | | | | | | |
| **Escenario** | **Descripción** | **Aplicación** | **Regla** | **Descripción** | **Respuesta del sistema** | **Flujo central** |
| EC 1.1 Adicionar regla de cortafuegos. | El usuario introduce los datos necesarios para insertar una nueva regla de cortafuegos | N/A | V<br>*uci.cu | V<br>Bloquo de Google | Se introduce la nueva regla de cortafuegos y se regresa a la vista del listado de reglas de cortafuegos donde se observa la regla recién creada | Seleccionar la opción "Cortafuegos" del menú lateral de la aplicación, luego pulsar el botón flotante que indica la opción de adicionar, después introducir los datos y pulsar el botón flotante que indicala opción de aceptar. |
| | | N/A | V<br>*uci.cu | V | | |
| EC 1.2 Adicionar regla de cortafuegos con datos incorrectos | El usuario introduce los datos necesarios para insertar una nueva regla de cortafuegos y la aplicación verifica que todos los datos sean correctos | N/A | I | V | La aplicación verifica que todos los datos introducidos sean correctos, si un dato no es válido no se adiciona la regla y se muestra un mensaje debajo del campo no válido indicando el error. | Seleccionar la opción "Cortafuegos" del menú lateral de la aplicación, luego pulsar el botón flotante que indica la opción de adicionar, después introducir los datos y pulsar el botón flotante que indicala opción de aceptar. |
| **SC2 Modificar Regla de cortafuegos** | | | | | | |
| **Escenario** | **Descripción** | **Aplicación** | **Regla** | **Descripción** | **Respuesta del** | **Flujo central** |
| EC 2.1 Modificar Regla de cortafuegos | El usuario selecciona una regla previamente creada y modifica los datos que desee | N/A | V<br>*uci.cu | V<br>Bloquo de Google | Se modifica la regla de cortafuegos, se regresa a la vista de visualizar la regla y se muestra un mensaje indicando que se modificó correctamente | Seleccionar la opción "Cortafugos" del menú lateral de la aplicación, luego seleccionar una regla previamente creada de la lista mostrada, después pulsar el botón flotante que indica la acción de modificar, entonces modificar los datos y pulsar el botón flotante que indicala opción de aceptar. |
| | | N/A | V<br>*uci.cu | V | | |
| EC 2.2 Modificar regla de cortafuegos con con datos incorrectos | El usuario selecciona una regla previamente creada, modifica los datos que desee y la aplicación verifica que estos datos sean correctos | N/A | I | V | La aplicación verifica que todos los datos introducidos sean correctos, si un dato no es válido no se modificala regla y se muesta un mensaje debajo del campo no válido indicando el error. | Seleccionar la opción "Cortafugos" del menú lateral de la aplicación, luego seleccionar una regla previamente creada de la lista mostrada, después pulsar el botón flotante que indica la acción de modificar, entonces modificar los datos y pulsar el botón flotante que indicala opción de aceptar. |
| **SC3 Mostrar Regla de cortafuegos** | | | | | | |
| **Escenario** | **Descripción** | **Aplicación** | **Regla** | **Descripción** | **Respuesta del** | **Flujo central** |
| EC 3.1 Mostrar la regla de cortafuegos. | Se muestran los datos de una regla de cortafuegos previamente creada. | | | | Se muestra en una nueva vista todos los datos de una regla de cortafuegos previamente creada | Seleccionar la opción "Cortafuegos" del menú lateral de la aplicación y luego seleccionar una regla previamente creada de la lista mostrada. |
| **SC4 Eliminar Regla de cortafuegos** | | | | | | |
| **Escenario** | **Descripción** | **Aplicación** | **Regla** | **Descripción** | **Respuesta del** | **Flujo central** |
| EC 4.1 Eliminar regla de cortafugos. | Se elimina una regla de cotafuegos previamente creada | | | | Se se elimina la regla de cortafuegos y se regresa a la vista del listado de reglas. | Seleccionar la opción "Cortafuegos" del menú lateral de la aplicación, luego seleccionar una regla previamente creado de la lista mostrada y después pulsar la opción de "eliminar" mostrada en la parte izquierda de la barra de herramientas de la vista |

**Anexo 15. Caso de Prueba para la HU Listar perfiles de conexión.**

| Descripción general | | | |
|---|---|---|---|
| El caso de prueba inicia cuando el usuario selecciona la opción de *Perfiles* en el menú lateral de la aplicación. Se presenta entonces al usuario el listado de los perfiles de conexión creados hasta el momento y además la opción de adicionar un nuevo perfil. El caso de prueba termina cuando el usuario abandona la lista *Perfiles*. | | | |
| **Condiciones de ejecución** | | | |
| | | | |
| **SC1 Listar perfiles de conexión** | | | |
| **Escenario** | **Descripción** | **Respuesta del sistema** | **Flujo central** |
| EC 1.1 Mostrar el listado de perfiles de conexión | Se muestran el listado de los perfiles de conexión previamente creados. | Se muestra en una vista la lista con los perfiles de conexión previamente creados | Seleccionar la opción "Perfiles" del menú lateral de la aplicación. |

**Anexo 16. Caso de Prueba para la HU Listar reglas de cortafuegos.**

| Descripción general | | | |
|---|---|---|---|
| El caso de prueba inicia cuando el usuario selecciona la opción de *Cortafuegos* en el menú lateral de la aplicación. Se presenta entonces al usuario el listado de las reglas de cortafuegos creadas hasta el momento. El caso de prueba termina cuando el usuario abandona la lista Reglas de cortafuegos. | | | |
| **Condiciones de ejecución** | | | |
| | | | |
| **SC1 Listar reglas de cortaguegos** | | | |
| **Escenario** | **Descripción** | **Respuesta del sistema** | **Flujo central** |
| EC 1.1 Mostrar el listado de reglas de cortafuegos. | Se muestran el listado de las reglas de cortafuegos previamente creadas. | Se muestra en una vista la lista con las reglas de cortafuegos previamente creadas | Seleccionar la opción "Cortafuegos" del menú lateral de la aplicación. |

**Anexo 17. Caso de Prueba para la HU Detener servicio de *proxy* local.**

| Descripción general | | | |
|---|---|---|---|
| El caso de prueba inicia después de que el usuario inicia el servicio de proxy local, se muestra entonces un botón flotante indicando la opción de parar el servicio. | | | |
| **Condiciones de ejecución** | | | |
| | | | |
| **SC1 Detener servicio de proxy local** | | | |
| **Escenario** | **Descripción** | **Respuesta del sistema** | **Flujo central** |
| EC 1.1 Detenenr Servicio de proxy local | Se detiene el servicio de proxy local | Se detiene el servicio de proxy local y se cambia el botón flotante que indica la acción de para por el que indica la acción de iniciar | Iniciar el servicio de proxy local. |

**Anexo 18. Caso de Prueba para la HU Restablecer configuración.**

| Descripción general | | | |
|---|---|---|---|
| El caso de prueba inicia cuando el usuario inicia la aplicación luego de haber inciado el servicio de proxy local anteriormente | | | |
| **Condiciones de ejecución** | | | |
| | | | |
| **SC1 Restablecer configuración** | | | |
| **Escenario** | **Descripción** | **Respuesta del sistema** | **Flujo central** |
| EC 1.1 Restablecer configuración | Se restablece la última configuración con la que se inició el servicio de proxy local. | Se autocompletan todos los tados necesarios para iniciar el servicio de proxy local, teniendo en cuenta los datos utilizados la última ocasión que se inició el servicio. | Iniciar la aplicación luego de haber iniciado y detenido el servicio de proxy local. |

**Anexo 19. Caso de Prueba para la HU Activar o desactivar regla de cortafuegos.**

| **Descripción general** | | | |
|---|---|---|---|
| El caso de prueba tiene lugar en el listado de reglas de cortafuegos en los checkbox que se muestran a la izquierda de cada regla | | | |
| **Condiciones de ejecución** | | | |
| Condiciones para SC1 y SC2: debe existir al menos una regla de cortafuegos creada anteriormente. | | | |
| **SC1 Activar regla de cortafuegos** | | | |
| **Escenario** | **Descripción** | **Respuesta del sistema** | **Flujo central** |
| EC 1.1 Activar regla de cortafuegos | Se activa una regla de cortafuegos que previamente estaba desactivada | Se activa la regla y se muestra un mensaje indicando que la regla fue activada. El sistema en consecuencia tiene en cuenta dicha regla para denegar peticiones. | Seleccionar la opción de "Cortafuegos" en el menú lateral desplegable de la aplicación |
| **SC2 Desactivar regla de cortafuegos** | | | |
| **Escenario** | **Descripción** | **Respuesta del sistema** | **Flujo central** |
| EC 1.1 Desactivar regla de cortafuegos | Se desactiva una regla de cortafuegos que previamente estaba activada | Se desactiva la regla y se muestra un mensaje indicando que la regla fue desactivada. El sistema en consecuencia no tiene en cuenta dicha regla para denegar peticiones. | Seleccionar la opción de "Cortafuegos" en el menú lateral desplegable de la aplicación |

**Anexo 20. Caso de Prueba para la HU Adicionar traza de navegación.**

| **Descripción general** | | | |
|---|---|---|---|
| El caso de prueba inicia cuando el usuario realiza una petición a Internet a través de una aplicación cliente o de un navegador. | | | |
| **Condiciones de ejecución** | | | |
| Condiciones para SC1: debe haberse iniciado el servicio de proxy local | | | |
| **SC1 Adicionar traza de navegación** | | | |
| **Escenario** | **Descripción** | **Respuesta del sistema** | **Flujo central** |
| EC 1.1 Adicionar traza de navegación | Se adiciona una traza de navegación por cada petición realizada en el dispositivo | Se adiciona una traza de navegación por cada petición realizada, esta traza puede ser visible desde el listado de trazas de navegación mostrado al selecionar la opción "Trazas" del menú desplegable. | Realizar una petición después de haber iniado el servicio de proxy local. |

**Anexo 21. Caso de Prueba para la HU Listar trazas de navegación.**

| **Descripción general** | | | |
|---|---|---|---|
| El caso de prueba inicia cuando el usuario selecciona la opción "Trazas" del menú desplegable de la aplicación. | | | |
| **Condiciones de ejecución** | | | |
| | | | |
| **SC1 Listar traza de navegación** | | | |
| **Escenario** | **Descripción** | **Respuesta del sistema** | **Flujo central** |
| EC 1.1 Listar trazas de navegación | Se listan las trazas de navegación creadas por las peticiones realizadas en el dispositivo | Se muestra el listado de las trazas de navegación creadas por las peticiones realizadas en el dispositivo. | Seleccionar la opción "Trazas" del menú lateral desplegable de la aplicación. |

**Anexo 22. Caso de Prueba para la HU Añadir traza de navegación como regla de cortafuegos.**

| **Descripción general** | | | | |
|---|---|---|---|---|
| El caso de prueba inicia cuando el usuario realiza una pulsación larga sobre una traza de navegación. | | | | |
| **Condiciones de ejecución** | | | | |
| Condiciones para SC1: debe existir al menos una traza de navegación creada. | | | | |
| **SC1 Adicionar traza de navegación como regla de cortafuegos.** | | | | |
| **Escenario** | **Descripción** | **Regla** | **Respuesta del sistema** | **Flujo central** |
| EC 1.1 Adicionar traza de navegación como regla de cortafuegos | El usuario añade una traza de navegación como regla de cortafuegos | V<br>*uci.cu | Se introduce la nueva regla de cortafuegos. | Seleccionar la opción "Trazas" del menú lateral de la aplicación, luego realizar una pulsación larga sobre la traza quese desea añadir como regla, presionar entonces el boton que muestra la acción de añadir como regla de cortafuegos y luego modificar la regla, si se desea, del diálogo que se muestra. |
| | | V<br>*uci.cu | | |
| EC 1.2 Adicionar traza de navegación como regla de cortafuegos con datos incorrectos | El usuario añade una traza de navegación como regla de cortafuegos con datos incorrectos | I | La aplicación verifica que todos los datos introducidos sean correctos, si un dato no es válido no se adiciona la regla de cortafuegos y se muesta un mensaje debajo del campo no válido indicando el error. | Seleccionar la opción "Trazas" del menú lateral de la aplicación, luego realizar una pulsación larga sobre la traza quese desea añadir como regla, presionar entonces el boton que muestra la acción de añadir como regla de cortafuegos y luego modificar la regla, si se desea, del diálogo que se muestra. |

**Anexo 23. Caso de Prueba para la HU Buscar trazas de navegación.**

| **Descripción general** | | | | |
|---|---|---|---|---|
| El caso de prueba inicia cuando el usuario selecciona la opción de buscar en la barra de herramientas de la vista que muestra el listado de las trazas de navegación. | | | | |
| **Condiciones de ejecución** | | | | |
| Condiciones para SC1: debe existir al menos una traza de navegación creada. | | | | |
| **SC1 Buscar trazas de navegación** | | | | |
| **Escenario** | **Descripción** | **Regla** | **Respuesta del sistema** | **Flujo central** |
| EC 1.1Buscar trazas de navegación | El usuario inserta un criterio de búsqueda. | V<br>facebook<br>V<br>google | Se muestra en el listado las trazas que coninciden con el criterio de busqueda, estas son buscadas por nombre de la aplicación y por la URL que se requirió. | Seleccionar la opción "Trazas" del menú lateral de la aplicación y luego selecccionar la opción de buscar en la barra de herramientas. |

**Anexo 24. Caso de Prueba para la HU Ordenar trazas de navegación.**

| **Descripción general** | | | |
|---|---|---|---|
| El caso de prueba inicia cuando el usuario selecciona la vista del listado de trazas de navegación | | | |
| **Condiciones de ejecución** | | | |
| Condiciones para SC1: debe haber al menos dos trazas de navegación creadas | | | |
| **SC1 Ordenar trazas de navegación** | | | |
| **Escenario** | **Descripción** | **Respuesta del sistema** | **Flujo central** |
| EC 1.1 Ordenar trazas de navegación | Se ordena el listado de las trazas de navegación. | Se ordena el listado de las trazas de navegación de mayor a menor según el consumo que provocaron en Mega Bytes. | Seleccionar la opción de "Trazas" en el menú lateral desplegable y luego seleccionar la opción de ordenar en el menú flotante ubicado en la esquina superior derecha de la vista. |

**Anexo 25. Acta de Aceptación.**

Universidad de las Ciencias Informáticas

# Centro de Soluciones Libres

# CESOL

## Acta de Aceptación

**De una parte**, el tesista Daniel Alejandro Rodríguez Caballero y de **otra parte** el Centro de Soluciones Libres de la Universidad de las Ciencias Informáticas, en lo adelante CESOL, representado por su director Mcs. Yoandy Pérez Villazón.

**Primero:** A raíz de la necesidad del CESOL de permitir el acceso a Internet a través de servidor *proxy* web en los dispositivos móviles con sistema operativo NovaDroid, se acuerda el desarrollo de una aplicación móvil que permita la autenticación con servidor *proxy* web en NovaDroid, a ser utilizada principalmente en los entornos institucionales del país.

**Considerando:** Que los hitos realizados han sido desarrollados con la calidad requerida y que los requisitos pactados han sido implementados en su totalidad.

**Considerando:** Que las pruebas realizadas en entornos de trabajo real han arrojado valores y criterios positivos.

**Por tanto:** Las partes acuerdan formalizar mediante la presente acta, la aceptación del producto **Aplicación móvil para la autenticación con servidor *proxy* web en NovaDroid** en su versión 1.0.

Para que así conste se extiende la presente acta en dos ejemplares rubricados por ambas partes.

Daniel Alejandro Rodríguez Caballero
Entrega

Mcs. Yoandy Pérez Villazón
Director CESOL
Recibe

Universidad de las Ciencias Informáticas
Centro CESOL Facultad 1

## Glosario de términos

**Perfil de conexión**: perfil con el cual se desea iniciar el servicio de *proxy* local de la solución, contiene los datos necesarios para establecer la conexión con el servidor *proxy* de la institución.

**Regla de cortafuegos**: regla a aplicar para bloquear las peticiones de determinada aplicación a un destino, varios destinos, un dominio o varios dominios.

**Traza de navegación**: contiene información acerca de cada una de las peticiones realizadas en el dispositivo: aplicación que realizó la petición, recurso que requirió, fecha de la petición y consumo que provocó.