UNIVERSIDAD DEL BÍO-BÍO
FACULTAD DE CIENCIAS EMPRESARIALES
DEPARTAMENTO DE INGENIERÍA CIVIL EN INFORMÁTICA

UNIVERSIDAD DEL BÍO-BÍO

"Desarrollo de una aplicación móvil en Android para la transmisión en paralelo de archivos en un entorno Wi-Fi P2P"

Proyecto de título para optar al título profesional de

Ingeniería Civil en Informática

Autoras
Carla Cáceres Valenzuela
Ana Sepúlveda Nieto

Profesor Guía
Patricio Galdames Sepúlveda

11 de marzo de 2016

RESUMEN

El presente proyecto da conformidad a los requisitos exigidos por la Universidad del Bío-Bío en el proceso de titulación para a la carrera de Ingeniería Civil en Informática. El proyecto, titulado "*Desarrollo de una aplicación móvil en Android para la transmisión en paralelo de archivos en un entorno Wi-Fi P2P*", tiene como resultado principal una aplicación en el sistema operativo Android que permite transferir archivos en paralelo entre dos o más dispositivos móviles que cuenten con la aplicación utilizando la tecnología Wi-Fi Direct, en la que los dispositivos pueden conectarse entre ellos de forma rápida, segura y conveniente, sin la necesidad de estar conectado a un red, aprovechando las características o condiciones más favorables de los dispositivos móviles cercanos (tasa de transmisión entre el dispositivo servidor y los dispositivos clientes y nivel de batería) para la transferencia de estos archivos.

La aplicación se desarrolla pensando en un ambiente académico, por ejemplo, una sala de clases, en donde los profesores y alumnos necesiten intercambiar documentos o archivos. La aplicación, al realizar la descarga en paralelo, permite la propagación del archivo entre los dispositivos de manera rápida. De esta manera, si en un comienzo una persona tiene el archivo original, lo transmite a otro usuario, y luego, siendo dos quienes lo tengan, podrán enviar a su vez el archivo paralelamente a otros dispositivos, disminuyendo los tiempos de descarga cada vez que el archivo sea compartido. A grandes rasgos, la aplicación puede ser utilizada en cualquier situación en que se desee compartir archivos entre varios dispositivos.

Dado que los dispositivos móviles son una herramienta y tecnología masiva, es posible que se encuentre un mismo archivo en distintos aparatos, por lo que la aplicación permite transferir la mayor parte, o incluso la totalidad, del archivo desde aquel que tenga una mayor cantidad de energía y tasa de transmisión, y el resto del que cuente con menos recursos. Considerando estas características, la aplicación permite reducir considerablemente los tiempos de descarga en comparación con otras técnicas de envío de datos, distribuyendo de la mejor forma posible la carga de cada dispositivo al momento de transferir estos datos.

ABSTRACT

The present project gives conformity to the requirements of the Universidad del Bío-Bío for the professional title of Civil Engineering in Computer Science. The project, entitled "*Development of a mobile application on Android for parallel transmission of files on a Wi-Fi P2P environment*", has as main result an application on Android OS that allows to transfer files by parallel transmission between two or more mobile devices that have the application, using the Wi-Fi Direct technology wherein devices can connect directly to each other quickly, securely and conveniently, without the need of being connected to a network, taking advantage of the most favorable features or conditions (transmission rate between the server device and a client device, and battery level) of the nearby mobile devices for transferring these files.

The application is developed thinking in an academic environment, for example, a classroom, where teachers and students need to exchange documents or files. The application, doing the parallel download, allows the spread of the file between devices quickly. Thus, if at first a person has the original file, and then they transmitted the file to another user, and then, as two have the file, they can send the file parallely to other devices, reducing download times each time the file is shared. Broadly speaking, the application can be used in any situation where is wanted to share files between multiple devices.

Due that mobile devices are a massive technological tool, it's possible that the same file is in different devices, so that the application transfers most, or even all, of the file from the device that has a greater amount of energy and transmission rate, and the rest of the one that has fewer resources. Considering these characteristics, the application allows to significantly reduce download times compared to other techniques for sending data, distributing in the best possible way the load of each device when transferring data.

Índice General

Índice Tablas

Índice Figuras

1 INTRODUCCIÓN

Este documento se presenta para dar conformidad a los requerimientos establecidos por la Universidad del Bío-Bío, para el proceso de titulación de la carrera de Ingeniería Civil Informática. En él se detalla el desarrollo de una aplicación móvil para el sistema operativo Android (versión 4.0 y posteriores), que permite transferir archivos en forma paralela desde distintos dispositivos móviles, combinando esto con la nueva tecnología "Wi-Fi Direct", anteriormente denominada "Wi-Fi P2P" [1], y así disminuir los tiempos de transferencia.

Hoy en día la capacidad de almacenamiento de los dispositivos móviles ha ido en aumento, lo que permite almacenar archivos de grandes tamaños en ellos. Al momento de querer compartir estos archivos a otros dispositivos, se cuenta con diversos métodos, como por ejemplo, Near Field Communication (NFC) y Bluetooth (detallados en el capítulo 3 de este proyecto), que no requieren una conexión a internet, pero su velocidad de transferencia es muy reducida sobre todo para grandes archivos. Cuando se cuenta con conexión a internet se tienen variadas posibilidades de transferir archivos (a través de redes sociales, páginas web especializadas en la descarga de archivos, aplicaciones que permiten el almacenamiento en la Nube, etc.), pero también se ve limitado por la cantidad de ancho de banda que se presenta y de las características propias de estas opciones. Desde su lanzamiento en 2010, la tecnología Wi-Fi Direct se ha ido dando a conocer, dado que permite la conexión directa entre dispositivos inalámbricos [2]. Por ejemplo, imprimir un archivo desde un notebook a una impresora inalámbrica o enviar imágenes de un celular a otro, esto sin necesidad de conexión a internet.

Como diversos dispositivos pueden contar con un mismo archivo, se ha propuesto implementar una técnica de descarga en paralelo desde los estos dispositivos, con el fin de reducir aún más los tiempos de transferencia de un dispositivo a otro. La idea principal es realizar una estimación del tiempo de transferencia de una muestra y añadir el factor nivel de batería y con estos datos calcular la cantidad de bytes a transferir desde cada dispositivo. Mientras más demore en transferir dicha muestra, menos cantidad de bytes se transferirá desde aquel dispositivo, pero si este presenta un alto nivel de batería, aumentará un tanto porciento dicha cantidad a transferir.

Utilizando la tecnología Wi-Fi Direct para la conexión de los dispositivos, y con la implementación de la descarga en paralelo, los tiempos de transferencia disminuyen

notablemente en relación a otras técnicas de transferencia de archivos, ya que actualmente no se han encontrado aplicaciones móviles que utilicen una combinación de estas técnicas.

El documento cuenta de 10 capítulos, los cuales están constituidos de acuerdo a la documentación de proyectos de desarrollo de software. El contenido de cada capítulo se describe a continuación.

- ✓ **Capítulo 1 "INTRODUCCIÓN".** Se presenta al lector cual es el propósito de este documento y se detalla el contenido de cada uno de sus capítulos.

- ✓ **Capítulo 2 "DEFINICIÓN DEL PROYECTO".** Este capítulo incluye lo relativo al proyecto en sí, los objetivos generales y específicos, la metodología utilizada y la descripción de la técnica utilizada para cumplir tales objetivos.

- ✓ **Capítulo 3 "ESTADO DEL ARTE".** Se describe las tecnologías que se ven implicadas en el proyecto, tales como los dispositivos móviles, el sistema operativo Android, las tecnologías de envío de datos existentes y finalmente, la tecnología base utilizada en la aplicación, Wi-Fi Direct.

- ✓ **Capítulo 4 "ESPECIFICACIÓN DE REQUERIMIENTOS DEL SOFTWARE".** Se detalla lo relativo a la aplicación, como sus alcances, objetivos generales y específicos, los requisitos para el correcto funcionamiento de la aplicación y la descripción global del software.

- ✓ **Capítulo 5 "ANÁLISIS".** En este capítulo se presenta el comportamiento y funcionamiento del software utilizando diversas herramientas de modelado.

- ✓ **Capítulo 6 "UBBiFi Direct".** En este capítulo se explican los aspectos generales de la aplicación "UBBiFi Direct", tales como; tipos de usuario, la técnica de transferencia de datos utilizada y la conexión de dispositivos mediante Wi-Fi Direct.

- ✓ **Capítulo 7 "PRUEBAS".** En este capítulo se especifican las pruebas que se realizan al software para comprobar la correcta funcionalidad de la aplicación.

- ✓ **Capítulo 8 "EXPERIMENTOS".** En este capítulo se realizan mediciones considerando distintas condiciones para la aplicación. También se realizan comparaciones con las demás tecnologías de envío de datos para comprobar la eficiencia sobre ellas.

- ✓ **Capítulo 9 "CONCLUSIONES".** En este capítulo se realiza la contrastación de los objetivos del proyecto y del sistema planteados y alcanzado al final del proyecto. Se incluyen conclusiones generales del proyecto desde los puntos de vista académico y personal.

- ✓ **Capítulo 10 "BIBLIOGRAFÍA".** Se indican las fuentes de información utilizadas durante el desarrollo del software.

2 DEFINICIÓN DEL PROYECTO

2.1 Objetivos del Proyecto

2.1.1 Objetivo General

Desarrollar una aplicación móvil para sistemas operativos Android que permita la transferencia en paralelo de archivos y que sea consciente del desempeño y batería de los dispositivos, utilizando la tecnología Wi-Fi Direct, llamada antiguamente Wi-Fi P2P.

2.1.2 Objetivos Específicos

- ✓ Investigar y conocer ampliamente la nueva tecnología Wi-Fi Direct para la conexión de dispositivos.
- ✓ Investigar técnicas de medición de la batería de los dispositivos.
- ✓ Conocer técnicas que permitan la correcta transferencia de los archivos entre dispositivos.
- ✓ Desarrollar un algoritmo que permita medir la porción de archivo a transferir desde los distintos dispositivos.
- ✓ Diseñar e implementar un algoritmo que permita encontrar un equilibrio entre la medición de la batería y la tasa de transferencia.
- ✓ Obtener los tiempos de transferencia del archivo en cada dispositivo

2.2 Ambiente de Ingeniería de Software

2.2.1 Metodología de desarrollo

En el desarrollo de nuestro proyecto utilizamos la metodología incremental, modelo que consiste en entregar pequeños avances llamados incrementos en los que se trabaja en base al incremento anterior.

El modelo incremental se compone en las siguientes etapas:

- ✓ Análisis: En esta etapa se analizan los requerimientos para determinar los objetivos de más importancia que se deben cubrir.
- ✓ Diseño: En esta fase se realiza el diseño de la aplicación.
- ✓ Código: En esta fase se implementa el código fuente para la realización del diseño de la aplicación.
- ✓ Pruebas: En esta etapa se comprueba si los componentes del sistema por cada iteración realizada funcionan correctamente y cumplen con los requisitos validando los resultados obtenidos, luego se ensamblan todas las iteraciones y se realiza una prueba general de la aplicación.

Escogimos esta metodología ya que necesitamos hacer pequeños avances o aplicaciones independientes por cada tema específico en el que nos enfocaremos y realizar las 4 etapas para cada uno de ellos.

2.2.2 Herramientas de apoyo al desarrollo de software

Para el desarrollo del proyecto se utilizarán las siguientes herramientas:

- ✓ **Eclipse Luna 4.4.2:** Eclipse es un entorno de desarrollo integrado (IDE). Contiene un área de trabajo y un sistema de plug-in para personalizar el entorno. Escrito principalmente en Java, Eclipse se puede utilizar para el desarrollo aplicaciones.
- ✓ **SDK Android (Android Developer Kit):** Provee las bibliotecas y herramientas de desarrollo necesarias para crear, probar y depurar aplicaciones para Android.
- ✓ **ADT 23.0.4 (Android Development Tools):** Es una extensión para el entorno de desarrollo Eclipse. Amplía las capacidades de Eclipse para que pueda: configurar rápidamente nuevos proyectos de Android; crear una interfaz de usuario; depurar aplicaciones utilizando las herramientas del SDK de Android; e incluso exportar una aplicación firmada o no, para distribuirla [3].
- ✓ **Smartphone Samsung Galaxy S5:** Teléfono móvil entregado por la Universidad del Bío-Bío para el desarrollo y realización de pruebas de la aplicación (5 Dispositivos).
- ✓ **Power Designer 16:** Software utilizado para la creación de los distintos diagramas presentados en el informe.

2.3 Definiciones, Siglas y Abreviaciones

2.3.1 Definiciones

- ✓ **Archivo**: Colección de datos o información medidos en bits. Un archivo es identificado por un nombre y la descripción de la carpeta o directorio que lo contiene. Hay distintos tipos de archivos: archivos de datos, archivos de texto, archivos de programa [4].
- ✓ **Checksum:** Un dígito que representa la suma de los dígitos correctos en una hoja de datos digitales almacenados o transmitidos, contra la que las comparaciones posteriores se pueden hacer para detectar errores en los datos [5].
- ✓ **Smartphone**: Término (en inglés) utilizado para referirse a un teléfono móvil inteligente dentro del presente proyecto.
- ✓ **Tablet**: Término (en inglés) utilizado dentro del presente proyecto para referirse a un computador portátil táctil que no requiere ratón ni teclado físico [6].
- ✓ **UBBiFi Direct**: Aplicación desarrollada en el presente proyecto.
- ✓ **Wi-Fi:** Mecanismo de conexión de dispositivos electrónicos de forma inalámbrica**.** Los dispositivos habilitados con Wi-Fi pueden conectarse a internet a través de un punto de acceso de red inalámbrica [7].
- ✓ **Wi-Fi Direct:** Tecnología que permite que varios dispositivos Wi-Fi se conecten entre sí sin necesidad de un punto de acceso intermedio [1].

2.3.2 Siglas y Abreviaciones

- ✓ **AP:** *Access Point*; Punto de Acceso en español. Un dispositivo de red que interconecta equipos de comunicación alámbrica para formar una red inalámbrica que interconecta dispositivos móviles o con tarjetas de red inalámbricas [8].
- ✓ **DHCP**: *Dynamic Host Configuration Protocol*; Protocolo de Configuración Dinámica de Host en español. Protocolo de red de tipo cliente/servidor en el que generalmente un servidor posee una lista de direcciones IP dinámicas y las va asignando a los clientes de la red [9].
- ✓ **FTP:** *File Transfer Protocol*; Protocolo de Transferencia de Archivos en español. Protocolo de red para la transferencia de archivos entre sistemas conectados a una red TCP, basado en la arquitectura cliente-servidor [10].

- ✓ **HTTP**: *Hypertext Transfer Protocol*; Protocolo de Transferencia de Hipertexto en español. Protocolo de comunicación cliente-servidor que permite los intercambios de información entre los clientes Web y los servidores HTTP [11].
- ✓ **IDE**: *Integrated development environment*; Ambiente de Desarrollo Integrado en español. Programa informático que consiste en un conjunto de herramientas para facilitar la edición de código para uno o varios lenguajes de programación, como también la compilación y depuración del código [12].
- ✓ **IEEE**: *Institute of Electrical and Electronics Engineers*; Instituto de Ingeniería Eléctrica y Electrónica en español. Asociación mundial de ingenieros dedicada a la estandarización y el desarrollo en áreas técnicas [13].
- ✓ **NFC**: *Near Field Communication*; Comunicación de Campo Cercano en español. Tecnología de comunicación inalámbrica, de corto alcance y alta frecuencia que permite el intercambio de datos entre dispositivos [14].
- ✓ **OS:** *Operating System*; Sistema Operativo en español. Conjunto de órdenes y programas que controlan los procesos básicos de un computador y permiten el funcionamiento de otros programas [15].
- ✓ **OSI**: *Open System Interconnection Model*; Modelo de Interconexión de Sistemas Abiertos en español. Modelo de referencia para los protocolos de la red de arquitectura en capas [16].
- ✓ **P2P**: *Peer-to-Peer*; Red de pares en español. Red de ordenadores en la que todas actúan como nodos, sin clientes ni servidores, para compartir archivos [17].
- ✓ **RFID**: *Radio Frequency IDentification*; IDentificación por Radiofrecuencia en español. Sistema de almacenamiento y recuperación de datos remoto [18].
- ✓ **SIG:** *Special Interest Group*; Grupo con Especial Interés en español. Comunidad dentro de una organización mayor con un interés compartido en el progreso de un área en específico de conocimiento, aprendizaje o tecnología [19].
- ✓ **TCP:** *Transmission Control Protocol*; Protocolo de Control de Transmisión en español. Protocolo fundamental de Internet, que aporta transporte confiable y bidireccional de datos [20].
- ✓ **WLAN**: *Wireless Local Area Network*; Red de Área Local Inalámbrica en español. Red inalámbrica local cuyo dispositivos no necesitan estar vinculados a través de cables para conectarse [21].
- ✓ **WPS:** *Wi-Fi Protected Setup*. Estándar para facilitar la creación de redes WLAN [22].

2.4 Descarga en paralelo

La descarga en paralelo es una técnica que permite la descarga simultánea de distintas partes de un archivo, estableciendo conexiones en paralelo con distintos servidores que contengan el mismo archivo en su totalidad. Al recibir las partes del archivo, se debe realizar la unión de estos, lo que permite recuperar completamente el archivo original. Este método contrasta con el método convencional, donde el cliente descarga archivos desde una sola fuente. La descarga en paralelo permite mejorar el rendimiento, disminuyendo los tiempos de descarga de archivos de gran tamaño y permitiendo una mayor capacidad de resistencia frente a la congestión o fallos en el servidor, ya que cada acceso al archivo es independiente [23].

2.4.1 Trabajos relacionados

Existe un proyecto de título de la Universidad del Bío-Bío, llamado *"Descarga en paralelo de archivos sensible a variaciones del ancho de banda*", desarrollado por Cristopher Barrientos Matamala, el cual implementa la descarga en paralelo de archivos [24], cuyas principales características son:

- ✓ La aplicación original utiliza el protocolo FTP para el intercambio de archivos.
- ✓ Determina el tamaño de las partes a descargar mediante mediciones de ancho de banda.
- ✓ Usa el protocolo TCP para la descarga confiable de archivos y sin pérdidas.
- ✓ Solo puede ser ejecutada en computadores Windows o Linux.

La descarga en paralelo ha sido empleada en Redes de Distribución de Contenido, redes Peer-to-Peer (P2P), con el fin de mejorar el rendimiento en las descargas [23]. La idea de utilizar múltiples servidores en paralelo para mejorar el rendimiento de descarga es bastante reciente.

Un novedoso enfoque de descarga paralelo fue propuesto por *Byers* en 1999, para minimizar el tiempo de la descarga y entrega de los datos. En su esquema, un archivo de tamaño F es dividido en $n = k + 1$ partes de tamañop, donde $n * p > k * p > F$. Para minimizar el número de partes duplicadas, cada server entrega los paquetes en un orden aleatorio. Tan pronto como se reciben las partes, el archivo original puede ir siendo reconstruido inmediatamente [25].

En 2002, *Rodríguez* y *Biersack* estudiaron la dinámica de descarga en paralelo donde un cliente descargaba archivos desde varios servidores que residían en una red. Ellos muestran que el ancho de banda óptimo se determina basándose en el tiempo de transmisión óptima. El tiempo de transmisión es óptimo cuando todos los servidores envían información útil hasta que el archivo sea recibido por completo y no hay tiempos ociosos entre la recepción de dos bloques consecutivos. Propusieron almacenar la información histórica de los servidores y probar cuál de ellos era el mejor. El cliente especificaba qué parte descargar de cada servidor. También propusieron la descarga dinámica, donde el cliente divide el archivo en partes más pequeñas y descarga dichas partes de distintos servidores [26].

En 2006, *Sohail* implementó la descarga en paralelo basadas en FTP, donde el servidor, mediante monitoreo dinámico, calcula la división óptima, no importa cuán abruptamente cambien las características de red y del servidor [27].

En 2004, *Nabat* y *Sidi* consideraron el escenario donde los archivos son solicitados desde varios servidores, y a su llegada analizan la cantidad de buffer necesaria [28].

3 ESTADO DEL ARTE

En este punto se definen las tecnologías usadas para el desarrollo de la aplicación "UBBiFi Direct", como también las tecnologías que existen actualmente para la transferencia de archivos. Uno de los puntos más importantes es la tecnología "Wi-Fi Direct", la cual permite la conexión entre dispositivos y que se explica más a fondo a continuación.

3.1 Dispositivos móviles inteligentes

Un dispositivo móvil inteligente, desde ahora smartphone (contraparte en inglés), es un teléfono con características muy avanzadas, tales como una pantalla táctil de alta resolución, conectividad Wi-Fi, capacidad de navegar por la Web, y la habilidad de aceptar sofisticadas aplicaciones [29]. Es por esto que cada vez se están requiriendo dispositivos con mayores capacidades de almacenamiento, procesamiento y hardware, que permita a los usuarios acceder a los distintos tipos de aplicaciones con una mayor fluidez. Los smartphones hoy en día se asemejan a verdaderos computadores ya que no sirven solo para hablar, sino que son un medio de comunicación, información y diversión para el usuario.

Estos smartphones funcionan bajo cualquiera de estos sistemas operativos móviles: iOS, Symbian, BlackBerry OS, Windows Mobile y Android OS [29]. Es en este último sistema operativo móvil en donde se desarrollará la aplicación "UBBiFi Direct".

3.2 Sistema Operativo Android

Android OS, o simplemente Android, es un sistema operativo móvil actualmente desarrollado por Google, basado en el kernel (núcleo de un sistema operativo) de Linux y está diseñado principalmente para dispositivos móviles táctiles, tales como smartphones y tablets, lo que permite un ambiente de desarrollo de aplicaciones móviles con altas características.

Android ofrece nuevas posibilidades para aplicaciones móviles al proveer un entorno de desarrollo abierto, basado en un kernel de código abierto Linux a diferencia de los otros sistemas operativos que están construidos sobre sistemas propietarios que a menudo dan prioridad a las aplicaciones nativas por sobre los creados por terceros y restringen la comunicación entre las aplicaciones y los datos nativos del smartphone.

El código fuente de Android es liberado por Google bajo licencias de código abierto. Android es altamente popular debido a que compañías tecnológicas muchas veces requieren un sistema operativo listo para ser utilizado, de bajo costo y personalizable para dispositivos de alta tecnología. Su naturaleza abierta ha alentado a una gran comunidad de desarrolladores y entusiastas a usar el código como base para distintos proyectos, lo que permite a usuarios avanzados añadir nuevas características, ya que pueden extraer y sustituir cualquier aplicación nativa por una alternativa creada por terceros, puesto que tiene acceso al hardware del dispositivo (Conjunto de elementos físicos que constituyen el dispositivo) a través de las mismas APIs (Interfaz de programación de aplicaciones). Por lo tanto, aplicaciones nativas y alternativas tienen igual jerarquía y son ejecutadas en un ambiente totalmente compatible [30].

3.2.1 Arquitectura Android

Para conocer el desarrollo de aplicaciones en Android es importante saber cómo está estructurado. La arquitectura de Android está formada por varias capas que facilitan al desarrollador la creación de aplicaciones. Además, esta distribución permite acceder a las capas más bajas mediante el uso de librerías para que el desarrollador no tenga que programar a bajo nivel las funcionalidades necesarias para que una aplicación haga uso de los componentes de hardware de los smartphones.

Cada una de las capas utiliza elementos de la capa inferior para realizar sus funciones, es por esto que a este tipo de arquitectura se le conoce también como pila. A continuación se muestra el diagrama que explica dicha arquitectura:

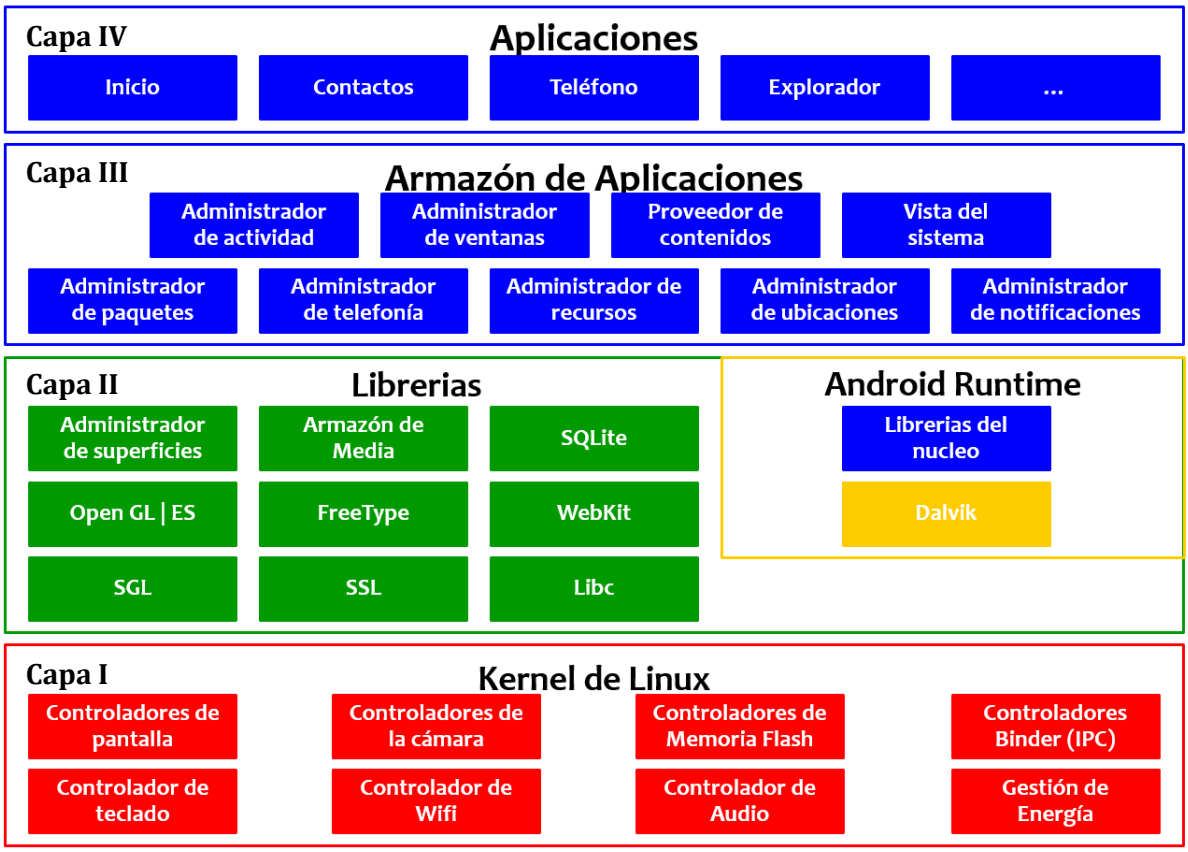

Figura 1: Arquitectura del Sistema Operativo Android

El núcleo del sistema operativo Android está basado en el "kernel" de Linux versión 2.6 (figura 1, capa I), adaptado a las características del hardware en el que se ejecutará Android. El núcleo actúa como una capa de abstracción entre el hardware y el resto de las capas de la arquitectura. El desarrollador no accede directamente a esta capa, sino que debe utilizar las librerías disponibles en capas superiores. Para cada elemento de hardware del smartphone existe un controlador (o driver) dentro del kernel que permite utilizarlo desde el software.

La capa II (figura 1), que se sitúa justo sobre el kernel, la componen las bibliotecas nativas de Android, también llamadas "librerías". Están escritas en C o C++ y compiladas para la arquitectura hardware específica del teléfono. Estas normalmente están hechas por el fabricante, quien también se encarga de instalarlas en el dispositivo antes de ponerlo a la venta. El objetivo de las librerías es proporcionar funcionalidad a las aplicaciones para tareas que se repiten con frecuencia, evitando tener que codificarlas cada vez y garantizando que se lleven a cabo de la forma más eficiente.

El entorno de ejecución de Android, o "Android Runtime", (figura 1, capa II) no se considera una capa en sí, dado que también está formado por librerías. Aquí encontramos las librerías con las funcionalidades habituales de Java así como otras específicas de Android.

El componente principal de Android Runtime es la máquina virtual Dalvik [31]. Las aplicaciones se codifican en Java y son compiladas en un formato específico para que esta máquina virtual las ejecute. La ventaja de esto es que las aplicaciones se compilan una única vez y de esta forma estarán listas para distribuirse con la total garantía de que podrán ejecutarse en cualquier dispositivo Android que disponga de la versión mínima del sistema operativo que requiera la aplicación.

La capa III (figura 1) está formada por todas las clases y servicios que utilizan directamente las aplicaciones para realizar sus funciones. La mayoría de los componentes de esta capa son librerías Java que acceden a los recursos de las capas anteriores a través de la máquina virtual Dalvik.

En la última capa, la capa IV (figura 1), se incluyen todas las "aplicaciones" del dispositivo, tanto las que tienen interfaz de usuario como las que no, las nativas (programadas en C o C++) y las administradas (programadas en Java), las que vienen preinstaladas en el dispositivo y aquellas que el usuario ha instalado.

Como podemos ver, Android nos proporciona un entorno sumamente poderoso para que podamos programar aplicaciones [31].

3.2.2 Niveles API y versiones de Android

Cada dispositivo Android soporta exactamente un nivel API, y se garantiza que este nivel API sea único para cada versión de la plataforma Android. El nivel API es la versión de librerías a la que una aplicación puede llamar; identifica la combinación de elementos del manifiesto, permisos, etc. contra los que un desarrollador codifica. El sistema de niveles API ayuda a Android a determinar si una aplicación es compatible con una imagen del sistema Android antes de instalar una aplicación en un dispositivo. Cuando se construye una aplicación, contiene la siguiente información de nivel API:

- ✓ El nivel API de Android objetivo en el que la aplicación está construida para ejecutarse.
- ✓ El mínimo nivel API de Android que se requiere para ejecutar la aplicación.

Esta configuración se utiliza para asegurar que la funcionalidad necesaria para ejecutar la aplicación está disponible en el dispositivo Android en el momento de la instalación. Si no, la aplicación es bloqueada para ejecutarse en ese dispositivo. Por ejemplo, si el nivel API de un dispositivo Android es más bajo que el nivel API mínimo requerido por una aplicación, el dispositivo Android va a prevenir al usuario de instalar esa aplicación [32].

Las actualizaciones para Android se han desarrollado bajo nombres inspirados en "postres" (Cupcake, Donut, Eclair, Gingerbread, Honeycomb, Ice Cream Sandwich, Jelly Bean, KitKat, Lollipop, Marshmallow). Cada versión, las cuales aparecen en orden alfabético y con nuevas mejoras, es diseñada para ser compatible con las versiones anteriores, para que una aplicación pueda siempre ser compatible con versiones futura de Android mientras se usen las APIs documentadas de Android. Además se agregan nuevas funcionalidades, se deprecian otras que ya no son relevantes para la nueva versión (no se eliminan, sino que se etiquetan como obsoleta, pero se pueden continuar utilizando) y se corrigen fallos para mejorar el rendimiento [22].

Debido a que los usuarios instalan aplicaciones tanto en versiones antiguas como en más recientes, las aplicaciones deben ser diseñadas para trabajar con múltiples niveles API de Android.

Cada versión de Android puede representarse por múltiples nombres:

- ✓ La versión Android, como "Android 4.4".
- ✓ El nivel API, como "API nivel 19".
- ✓ Un nombre de postre, como "KitKat".

Las nuevas versiones de Android son lanzadas aproximadamente una vez por año, y a la vez estas versiones cuentan con actualizaciones, las que son liberadas varias veces por año. Esto da como resultado que un amplio universo de dispositivos Android, con una variedad de versiones antiguas o nuevas, puedan correr una aplicación en específico. Entonces, ¿cómo asegurar que una aplicación se ejecute de manera consistente y confiable en tantas versiones diferentes de Android? Es aquí donde las APIs de Android ayudan a manejar este problema [32].

3.3 Tecnologías de envío de datos

Debido al creciente aumento de la tecnología móvil inalámbrica, se han promovido nuevas y revolucionarias formas de transferencia de datos entre dispositivos.

Actualmente existen diferentes tecnologías disponibles en el mercado, con rangos que se extienden desde unos pocos centímetros hasta cientos de metros. Cada una de ellas ofrece un conjunto único de rendimiento, rango, seguridad, costo, etc., lo que los hace adecuados para distintos campos de aplicación.

Cuando se trata de dispositivos móviles, hay tres tecnologías inalámbricas que son usadas más frecuentemente que otras para la transferencia de datos, las que se muestran en la figura 2. Wireless Fidelity (Wi-Fi), Bluetooth y Near Field Communication (NFC) son hoy en día ampliamente soportados por la mayoría de los dispositivos móviles. Estas tecnologías son por lo tanto altamente relevantes de tener en cuenta cuando se investiga sobre Wi-Fi Direct.

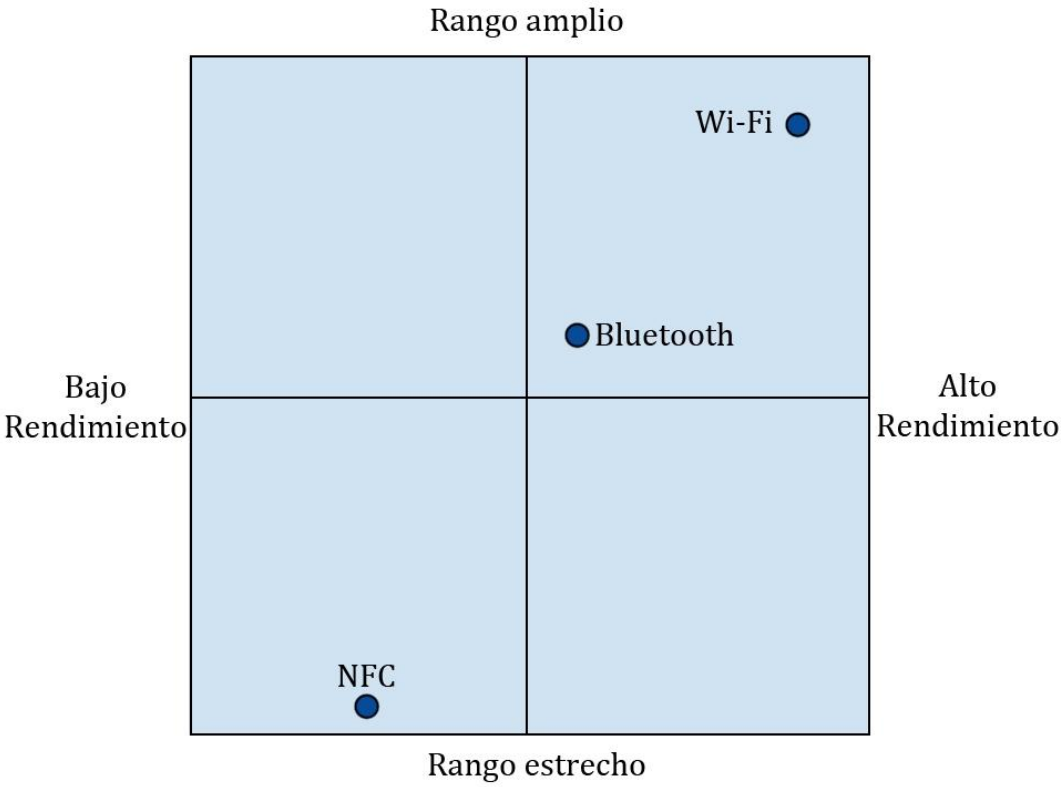

Figura 2: Tecnologías inalámbricas usadas en dispositivos móviles

3.3.1 Wi-Fi (IEEE 802.11 WLAN)

Wi-Fi es una marca comercial de Wi-Fi Alliance, organización que adopta y certifica los equipos que cumplen con los estándares 802.11 de las redes inalámbricas de área local. Wi-Fi Alliance define Wi-Fi como un dispositivo de red de área local inalámbrica (WLAN) que implementa el estándar 802.11 por el IEEE [33].

Para que se entienda mejor, Wi-Fi es una metodología de conexión que permite interconectar dispositivos y acceder a Internet sin usar cables ni realizar configuraciones avanzadas, lo que permite una gran simpleza de uso y movilidad. Los dispositivos que poseen esta posibilidad, tales como computadores, computadores portátiles, consolas de videojuegos, smartphones o tablets, entre otros, se pueden conectar a internet a través de un punto de acceso de red inalámbrica. Este punto de acceso tiene un alcance de 32 metros en interiores, mientras que al aire libre su distancia de alcance es mayor, aproximadamente de 95 metros [34].

Una WLAN es una red inalámbrica de dispositivos que se comunican con señales de frecuencia de radio. La mayoría los dispositivos modernos WLAN basados en el estándar IEEE 802.11 son certificados Wi-Fi, lo cual es la razón de porqué el término Wi-Fi se ha convertido en sinónimo de IEEE 802.11 WLAN. Todos los dispositivos con una WLAN caen en dos categorías, "puntos de acceso" y "estaciones" (también conocidos como clientes en Wi-Fi):

- ✓ **Punto de Acceso (AP o WAP - Wireless Access Point):** Es el dispositivo central que interconecta equipos de comunicación inalámbrica para formar una red que permite transportar datos entre ellos. Un AP puede también actuar como un dispositivo puente para sus estaciones hacia otras redes, por ejemplo, cuando un router permite a sus clientes acceso inalámbrico a Internet.
- ✓ **Estaciones (STA):** Llamados clientes, es un dispositivo que soporta IEEE 802.11, usado en la infraestructura WLAN y el modo Ad hoc. Una estación puede ser un teléfono móvil, un computador portátil o un computador de escritorio [35].

WLAN opera en dos modos básicos: el modo "Infraestructura" y el modo "Ad hoc" [36]. Sin embargo, la mayoría de las redes WLAN hoy en día están usando el modo infraestructura. Cuando hablamos de Wi-Fi tradicional, usualmente se refiere al modo infraestructura.

3.3.1.1 Modo Infraestructura

El modo Infraestructura es una red tipo cliente-servidor, que consiste en un dispositivo AP central y múltiples dispositivos estaciones (clientes) conectados a él. Aunque WLAN provee un alto rendimiento, tiene en contra el consumo de energía (un AP típico consume 150 mA (miliamperios) con una fuente de alimentación de 15 V (voltios)). Se requiere que el AP esté constantemente transmitiendo datos para mantener el enlace con cada estación aún si no hay transmisión [37].

3.3.1.2 Modo Ad hoc

Ad hoc es un tipo de red descentralizada que consiste solo de estaciones (clientes), como se puede ver en la figura 3, donde cada cliente se comunica directamente con los demás, sin necesidad de un AP (punto de acceso).

Las configuraciones Ad hoc son comunicaciones punto a punto. Solo los dispositivos dentro de un rango definido pueden comunicarse entre sí para formar una red peer to peer (de igual a igual) [38].

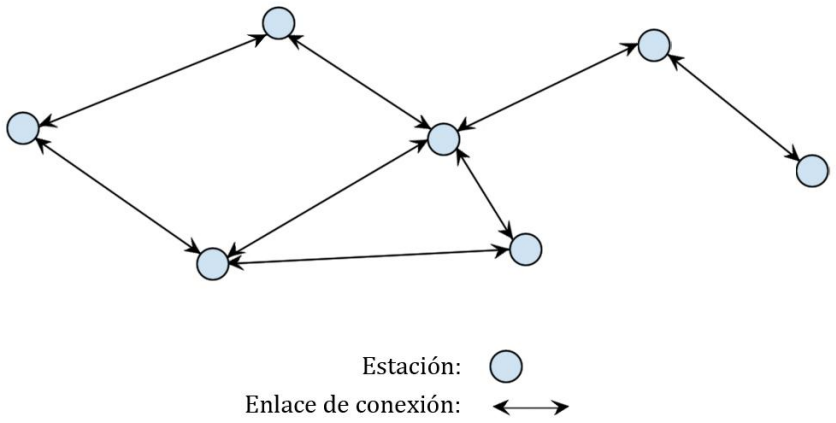

Figura 3: Estructura de una Red Ad hoc

La ventaja del modo Ad hoc es que es una solución que ofrece flexibilidad y movilidad. Comparado con el modo Infraestructura, Ad hoc no está limitado por un punto físico dado que la ubicación de cada estación no está limitado al rango del AP. Pero el nivel de rendimiento también es más bajo. Los dispositivos Wi-Fi en modo Ad hoc tiene una tasa de transferencia de datos máxima de 11 Mb/s (megabits por segundo). El consumo de energía de cada estación en

Ad hoc es también más grande que el de las estaciones en modo infraestructura, consumiendo 156 mA (miliamperios) en modo inactivo [39].

3.3.2 Bluetooth

La tecnología inalámbrica Bluetooth es una tecnología de ondas de radio de corto alcance que opera en la banda de frecuencia de 2,4 GHz (gigahercios), cuyo objetivo es el simplificar las comunicaciones entre dispositivos informáticos, como computadores portátiles, smartphones, tablets, entre otros. También pretende simplificar la sincronización de datos entre los dispositivos y otros computadores.

La topología de red de Bluetooth es 1:N, donde un dispositivo "maestro" puede conectarse a uno o muchos dispositivos "esclavos". La red formada es llamada una "Piconet", donde el dispositivo con rol maestro es responsable del control de la red [38].

Existen equipos Bluetooth clase 1, 2 y 3. Estas clases se diferencian entre sí solo por el rango de alcance de la comunicación inalámbrica. Los dispositivos clase 1 llegan a 100 metros, los de clase 2 lo hacen a 20 metros, mientras que los dispositivos Bluetooth de tercera clase, poseen apenas un metro de alcance y son los que casi no se usan [40].

3.3.3 Near Field Communication (NFC)

NFC es una tecnología inalámbrica usada para la comunicación de corto alcance entre dispositivos. El rango de transmisión es aproximadamente de unos pocos centímetros, y la comunicación es inmediatamente iniciada cuando dos dispositivos NFC son puestos uno junto a otro, movimiento llamado "one-touch" (un toque en español) o "tapping". NFC opera en la banda 13.56 MHz (megahercios) y soporta un máximo rendimiento de 424 Kb/s (kilobits por segundo) [41].

Hay tarjetas SIM con tecnología NFC, denominadas "tag", en las que no hace falta cambiar de móvil para utilizarlas. Funcionan tanto con smartphones como con los teléfonos móviles tradicionales. La mayoría de los nuevos dispositivos móviles ofrecen ahora soporte NFC. En varios casos, NFC es combinado con otra tecnología inalámbrica, por ejemplo, Bluetooth [42].

3.4 Wi-Fi Direct

Wi-Fi Direct, inicialmente llamado Wi-Fi peer-to-peer (P2P), es una tecnología desarrollada por Wi-Fi Alliance que soporta los estándares IEEE 802.11 a/b/g/n/ [43], la cual permite a los dispositivos que cuenten con la tecnología Wi-Fi conectarse directamente sin necesidad de un AP (punto de acceso) que permita la comunicación entre ellos. Los dispositivos que soportan Wi-Fi Direct no tienen necesidad de acceder a internet, los dispositivos simplemente establecen un "Grupo P2P" para comunicarse entre ellos (comunicación Peer-to-Peer). Como se muestra en la figura 4, un dispositivo que pertenece a un Grupo P2P es llamado Group Owner P2P (GO P2P) o cliente P2P.

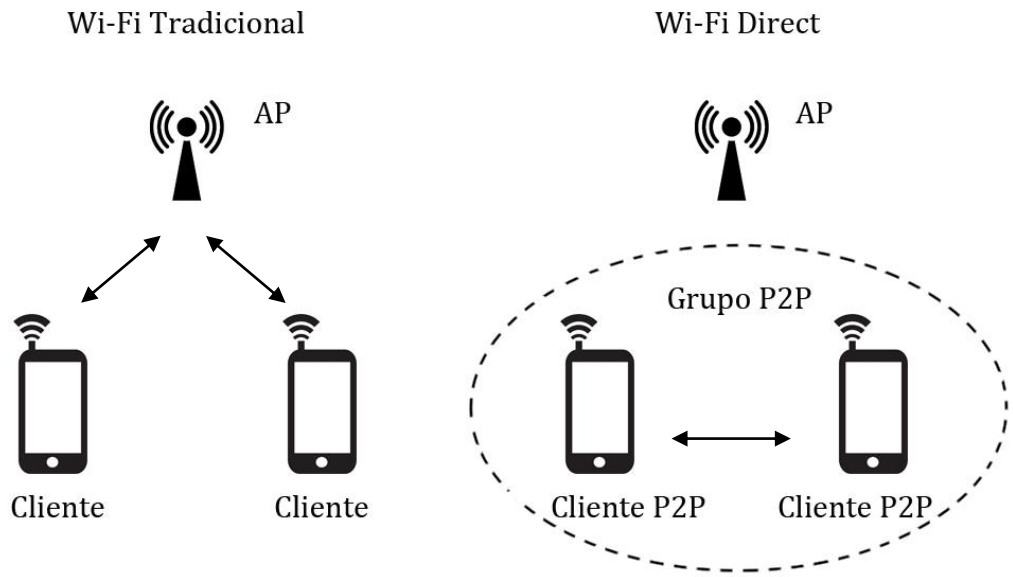

Figura 4: Conexión Wi-Fi comparada con Wi-Fi Direct

La topología en un Grupo P2P es 1:N, donde un GO P2P (Group Owner P2P), que en este caso actúa similar a un AP (punto de acceso) normal, puede conectarse a múltiples clientes. Es similar a la topología WLAN pero sin la comunicación cliente-cliente a través del GO P2P. Wi-Fi Direct se diferencia de Ad hoc debido a que sus estructuras de red son completamente diferentes, aunque ambos son soluciones peer-to-peer.

Wi-Fi Direct es compatible con dispositivos certificados Wi-Fi. Dado que Wi-Fi Direct usa los mismos estándares IEEE 802.11 que el Wi-Fi tradicional, puede operar a distancias de hasta 200 metros y alcanzar velocidades de transferencia de datos de hasta 250 Mb/s (megabits por segundos).

Actualmente hay cerca de 5000 dispositivos electrónicos certificados con Wi-Fi Direct, que van desde smartphones y tablets a televisores y dispositivos de juegos. Wi-Fi Direct ha sido una característica obligatoria para los nuevos dispositivos Android comenzando con la versión 4.0 "Ice Cream Sandwich". Dispositivos con Android inferior a la versión 4.0 no soportan esta tecnología. El primer smartphone en pasar la certificación fue el Samsung Galaxy S [1].

Los dispositivos que usan Wi-Fi Direct, oficialmente denominados "dispositivos P2P", interactúan con otros dispositivos estableciendo Grupos P2P. Un dispositivo en un Grupo P2P asumirá el rol de Group Owner P2P y los otros dispositivos en el grupo serán referidos como clientes P2P [44].

3.4.1 APIs Wi-Fi Direct

Wi-Fi Direct permite a dispositivos Android 4.0 (API Nivel 14) o posteriores con el hardware apropiado conectarse directamente entre ellos vía Wi-Fi sin un AP (Punto de Acceso) intermediario. El framework Wi-Fi P2P de Android cumple con el programa de certificación Wi-Fi Direct™ de Wi-Fi Alliance [45].

Usando estas APIs, se puede detectar y conectar a otros dispositivos que soporten Wi-Fi Direct, para luego comunicarse mediante de una conexión rápida a través de distancias mucho mayores que la conexión Bluetooth. Esto es útil para aplicaciones que comparten datos entre los usuarios, tales como juegos multijugador o aplicaciones para compartir imágenes.

Las APIs Wi-Fi Direct contienen las siguientes partes principales:

- ✓ Métodos que permiten descubrir, solicitar, y conectar a dispositivos P2P, los que son definidos en la clase "WifiP2pManager" [45].
- ✓ Listeners, métodos llamados por la aplicación cuando el usuario interactúa con la UI (interfaz de usuario), que permiten notificar del éxito o fallo de las llamadas de algún método de la clase "WifiP2PManager" [45].

- ✓ Intents (objetos que contienen la información necesaria para lanzar la ejecución de algún componente de la aplicación), los que notifican cuando eventos específicos son detectados por el framework Wi-Fi P2P, tales como la caída de la conexión a un peer recién descubierto [45].

3.4.2 Dispositivo P2P

Los dispositivos Wi-Fi Direct, conocidos formalmente como dispositivos P2P, pueden soportar el rol tanto como de Group Owner P2P como de Cliente P2P, ya que sus roles son dinámicos.

3.4.2.1 Group Owner P2P (GO P2P)

El rol GO P2P es equivalente a la función del AP tradicional en las redes WLAN, ya que funciona como un Software de Punto de Acceso (Soft AP) habilitado para los clientes P2P en su Grupo P2P. Solo el GO P2P puede conectarse a todos los clientes P2P. Este tipo de conexión es realizado en la capa de red OSI [46]. El GO P2P actuará entonces como servidor DHCP con lo que proveerá a sus clientes P2P con direcciones IP.

3.4.2.2 Cliente P2P

Los clientes P2P funcionan como clientes Wi-Fi tradicionales, conectados al GO P2P. Un cliente P2P está diseñado para comunicarse con el GO P2P en el Grupo P2P.

3.4.3 Detección de Dispositivos y Formación del Grupo

Hay tres formas en que dos dispositivos pueden establecer un Grupo P2P; formación Estándar, formación Autónoma o formación Persistente. Los tres procedimientos comienzan con un escaneo activo Wi-Fi tradicional (IEEE 802.11), pero difieren en las fases posteriores [47], como se observa en la figura 5.

- ✓ **Formación Estándar**

La formación Estándar es usada cuando dos dispositivos P2P desconocidos establecen un Grupo P2P. Este procedimiento es típicamente realizado cuando dos dispositivos establecen un grupo por primera vez y necesitan negociar quién tomará el rol de GO P2P. Además del escaneo activo, la fase de detección también incluye una fase de hallazgo donde las peticiones de sondeo y las respuesta de sondeo son usadas para detectar otros dispositivos [44].

- ✓ **Formación Autónoma**

La formación autónoma es utilizada cuando un dispositivo P2P ya ha decidido convertirse en GO P2P y autónomamente crea su propio grupo. Otros dispositivos pueden detectar el grupo establecido y unirse como clientes, y por lo tanto saltarse la fase de negociación de GO P2P. La fase de detección también está simplificada, dado que el dispositivo que establece el grupo no necesita pasar por la fase de hallazgo [44].

- ✓ **Formación Persistente**

Cuando durante la formación de grupo los dispositivos P2P usan una bandera para declarar al grupo como persistente, pueden recordar las credenciales de la red y asignar roles P2P en el Grupo P2P. La próxima vez que los dispositivos se deseen conectar, uno de ellos envía una petición de invitación para re-establecer rápidamente el grupo persistente con sus roles originales [44].

Formación Estándar	**Formación Autónoma**	**Formación Persistente**
1) Detección de dispositivos	1) Detección de dispositivos	1) Detección de dispositivos
Fase de escaneo	Fase de escaneo	Fase de escaneo
Fase de hallazgo	-	-
2) Negociación del GO P2P	-	2) Invitación
3) Aprovisionamiento WPS	2) Aprovisionamiento WPS	3) Aprovisionamiento WPS
Fase 1	Fase 1	Fase 1
Fase 2	Fase 2	Fase 2

Figura 5: Procedimientos de detección P2P y formación de Grupo P2P

3.4.3.1 Detección de Dispositivos

- ✓ **Fase de escaneo**

Un dispositivo comienza la fase de escaneo al realizar un escaneo Wi-Fi tradicional a través de todos los canales soportados para encontrar y recopilar información sobre los Grupos P2P próximos. Al usar el escaneo Wi-Fi, el dispositivo P2P localiza el mejor canal potencial para luego establecer un grupo P2P. Durante esta fase, el dispositivo P2P no debe responder a ninguna petición de sondeo [48].

- ✓ **Fase de hallazgo**

Una vez que la fase de escaneo termina, el dispositivo procede con la fase de hallazgo. El propósito es asegurar rápidamente que dos dispositivos P2P en "detección de dispositivo" estén ubicados en el mismo canal para intercambiar información del dispositivo y determina si una conexión debería ser intentada. Esto es logrado dejando a los dispositivos P2P alternar entre los estados de escucha y de búsqueda.

El estado de escucha es cuando un dispositivo P2P espera por una petición de sondeo en un canal fijo, y el estado de búsqueda es cuando el dispositivo P2P envía peticiones de sondeo a un conjunto de canales fijos. Si un dispositivo P2P adecuado recibe la petición de sondeo, este puede responder con una respuesta de sondeo para proceder con la siguiente fase.

La probabilidad de que dos dispositivos se encuentren entre ellos depende del número de canales y el tiempo que demoren en cada uno de los dos estados. Esto se optimiza escogiendo un conjunto de canales recomendados para Wi-Fi Direct, los canales 1, 6 y 11, en la banda de 2.4 GHz (gigahercios), como se muestra en la figura 6 [48].

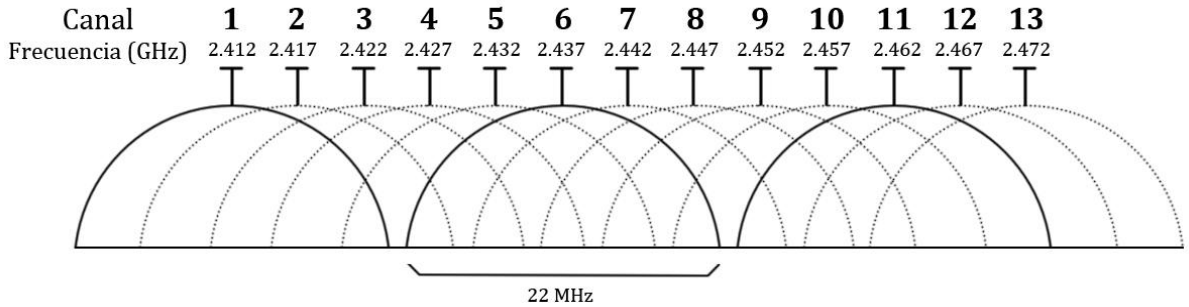

Figura 6: Canales en la banda 2.4 GHz

3.4.3.2 Negociación del Group Owner P2P (GO P2P)

Cuando la información del dispositivo ha sido intercambiada, la negociación de GO P2P toma lugar. Esto se realiza por un handshake de tres sentidos (palabra inglesa cuyo significado es apretón de manos y que es utilizada en tecnologías informáticas. Handshake (o handshaking, dependiendo del contexto) es un proceso automatizado de negociación que establece de forma dinámica los parámetros de un canal de comunicaciones establecido entre dos entidades antes de que comience la comunicación normal por el canal [49]) entre los dos dispositivos P2P (negociación GO/respuesta/confirmación).

Con el fin de llegar a un acuerdo en el que se defina cuál dispositivo actuará como GO P2P, los dispositivos P2P envían un parámetro numérico, llamado GO Intent, y el dispositivo cuyo GO Intent tenga el valor más alto se convertirá en GO P2P. Para prevenir conflictos entre dos dispositivos declarados con el mismo GO Intent (valor que indica la inclinación de un dispositivo de ser GO P2P), un bit "tie-breaker" (bit que se encarga de "romper el empate", si este se presenta) es incluido en la petición de negociación de GO, el cual es aleatoriamente definido cada vez que una petición de negociación de GO es enviada [48].

3.4.3.3 Aprovisionamiento WPS

Cuando los respectivos roles han sido negociados la parte de autentificación se lleva a cabo. Los dispositivos Wi-Fi Direct usan el protocolo de seguridad, Wi-Fi Protected Setup (WPS), para establecer y apoyar la conexión. AL utilizar el protocolo WPS se requiere que el Grupo P2P implemente dos partes, "Matriculador" y "Matriculado" (GO P2P y cliente P2P respectivamente) [48].

- ✓ **Fase 1**

En la primera fase el matriculador genera y emite las credenciales de red para el matriculado mediante el intercambio de marcos de información.

- ✓ **Fase 2**

Durante la segunda fase el matriculado se vuelve a conectar al matriculador usando sus nuevas credenciales de autentificación.

3.4.3.4 Invitación P2P

La invitación es un procedimiento opcional que se utiliza durante la re-invocación de un grupo persistente, en el cual dos dispositivos P2P (uno de ellos GO) han sido previamente aprovisionado. Una solicitud de invitación puede ser enviada tanto desde el cliente como del GO, y el grupo persistente es re-invocado una vez que se recibe a una respuesta de invitación exitosa.

4 ESPECIFICACIÓN DE REQUERIMIENTOS DEL SOFTWARE

4.1 Alcances

El proyecto a desarrollar tiene como objetivo crear una aplicación que permita la correcta transferencia en paralelo de archivos implementando la técnica de descarga en paralelo explicada en el capítulo 2 “DEFINICIÓN PROYECTO”.

La aplicación permite:

- ✓ la descarga de partes de un archivo desde 1 hasta 4 dispositivos móviles que cuenten con la aplicación, permitiendo que el dispositivo GO (Group Owner) pueda unir dichas partes y obtener un archivo completo. El usuario debe indicar la cantidad de clientes desde los cuales se descarga el archivo.

La aplicación no permite:

- ✓ la comunicación escrita entre los dispositivos, estilo mensajería instantánea.
- ✓ la ejecución o reproducción de los archivos descargados.
- ✓ el envío de más de un archivo al mismo tiempo desde un mismo dispositivo.
- ✓ navegar entre las carpetas de archivos de otros dispositivos.
- ✓ que un dispositivo actúe como cliente y GO a la vez, es decir, que no puede estar recibiendo un archivo y enviando otro al mismo tiempo.
- ✓ que los clientes seleccionen archivos distintos para ser transferidos.

4.2 Objetivo del software

4.2.1 Objetivo General

La correcta transferencia de archivos en paralelo entre dispositivos Android mediante Wi-Fi Direct aplicando la técnica de “descarga en paralelo adaptativa P2P".

4.2.2 Objetivos Específicos

- ✓ La aplicación detecta otros dispositivos cercanos que hayan iniciado una búsqueda por la aplicación "UBBiFi Direct".
- ✓ La aplicación permite establecer conexión entre el GO (Group Owner) y los clientes.
- ✓ La aplicación permite seleccionar los archivos a enviar entre las distintas carpetas del dispositivo.
- ✓ La aplicación utiliza correctamente la técnica de "descarga en paralelo adaptativa P2P" para gestionar la descarga de archivos.
- ✓ La aplicación reconstruye el archivo luego de recibidas las partes en el GO.
- ✓ La aplicación consigue menores tiempos de transferencia de archivos en comparación a otras tecnologías de envío de datos.

4.3 Requisitos mínimos del Software

En este punto se muestran los requisitos mínimos que requiere la aplicación para su perfecto funcionamiento.

4.3.1 Requisitos del Sistema Operativo

Debido a que la aplicación se encuentra codificada en el lenguaje de programación Java y además utiliza la tecnología Wi-Fi Direct, funciona sobre la siguiente versión del SO Android y posteriores:

Nombre: Android

Desarrollador: Google.

Versión: Android 4.0 "Ice Cream Sandwich" (API nivel 14) y posteriores.

4.3.2 Requisitos de Software

Dado que el sistema operativo Android viene con todo lo necesario para ejecutar la aplicación, no se requiere la instalación de algún software en específico.

4.3.3 Requisitos de Hardware

Es necesario un dispositivo móvil compatible con la versión Android descrita en el punto 4.3.1, debido a que los requisitos para soportar el sistema operativo son los mismos que soportan la aplicación. Para el desarrollo de este proyecto se contó con 5 dispositivos que contaban con este requisito.

4.4 Descripción Global del Producto

4.4.1 Interfaz de hardware

La aplicación utiliza la conexión de sockets para la transferencia de archivos, utilizando en específico el puerto 8988.

4.4.2 Interfaz de software

La aplicación no necesita el uso de otros software, ni interfaces con otros sistemas. La aplicación es autónoma en ese sentido.

4.4.3 Interfaces de comunicación

En este ítem se mencionan los protocolos que son utilizados por la aplicación para comunicarse entre los dispositivos.

Nombre: *Transmission Control Protocol*

Abreviación: TCP

Es el protocolo de la capa de transporte que se usa para el envío de los segmentos de una manera confiable libre de errores y sin pérdidas [20].

Nombre: HyperText Transfer Protocol

Abreviación: HTTP

Protocolo de la capa de aplicación que permite la comunicación entre el servidor y cliente a través de sockets TCP [11].

Nombre: *Dynamic Host Configuration Protocol*

Abreviación: DHCP

Protocolo que utiliza el GO (Group Owner) para ejecutar un servidor DHCP que permite proveer a los clientes con direcciones IP [10].

4.5 Requerimientos Específicos

4.5.1 Requerimientos Funcionales del sistema

ID	NOMBRE	DESCRICPCIÓN
1	Activar Wi-Fi	La aplicación debe activar el Wi-Fi si es que este se encuentra inhabilitado.
2	Buscar dispositivos	Permite la búsqueda de dispositivos cercanos que cuenten con la aplicación que hayan iniciado una búsqueda.
3	Lista de dispositivos	La aplicación debe mostrar la lista de dispositivos cercanos encontrados.
4	Conectar dispositivos	Permite la conexión entre los dispositivos clientes y Group Owner mediante la tecnología Wi-Fi Direct.
5	Selección clientes	La aplicación permite al usuario GO seleccionar la cantidad de clientes que enviarán parte del archivo, desde 1 a 4 clientes.
6	Búsqueda archivo	La aplicación permite al usuario seleccionar un archivo desde el directorio del dispositivo.
7	Progreso transferencia	La aplicación debe mostrar el proceso de transferencia del archivo mediante una barra de progreso en el dispositivo cliente.
8	Cancelar descarga	Permite al GO cancelar la transferencia en curso de un archivo.
9	Descarga finalizada	La aplicación debe dar aviso cuando el archivo se ha almacenado totalmente en el GO.

ID	NOMBRE	DESCRICPCIÓN
10	Fallo dispositivo	La aplicación debe dar aviso cuando la conexión a un dispositivo ha fallado.
11	Desconectar dispositivo cliente	La aplicación permite al usuario desconectar un dispositivo cliente del GO (Group Owner), sin alterar la conexión de los otros clientes.
12	Desconectar dispositivo GO	La aplicación permite al usuario desconectar a todos los clientes desde el GO.
13	Unión archivo	La aplicación realiza la unión de las partes del archivo transferidas desde cada cliente al GO.

Tabla 1: Requerimientos funcionales del sistema

4.5.2 Interfaces externas de entrada

Cada interfaz de entrada indica todos los grupos de datos que serán ingresados al sistema, independiente del medio de ingreso.

ID	NOMBRE DEL ÍTEM	DETALLE DE DATOS CONTENIDO EN ÍTEM
1	Cantidad de clientes	Número de clientes que van a enviar el archivo.

Tabla 2: Interfaces externas de entrada

5 ANÁLISIS

5.1 Procesos de negocio futuros

En este punto se presenta el diagrama de "Modelado de procesos de negocio", conocido por BPMN (*Business Process Modeling Notation*) [50], el cual se encarga de mostrar la secuencia de procesos que ejecuta la aplicación y los mensajes que fluyen entre los actores, en este caso los clientes y el GO (Grupo Owner). Este diagrama se basa en la técnica de flujos de datos para una mejor comprensión del lector (figura 7).

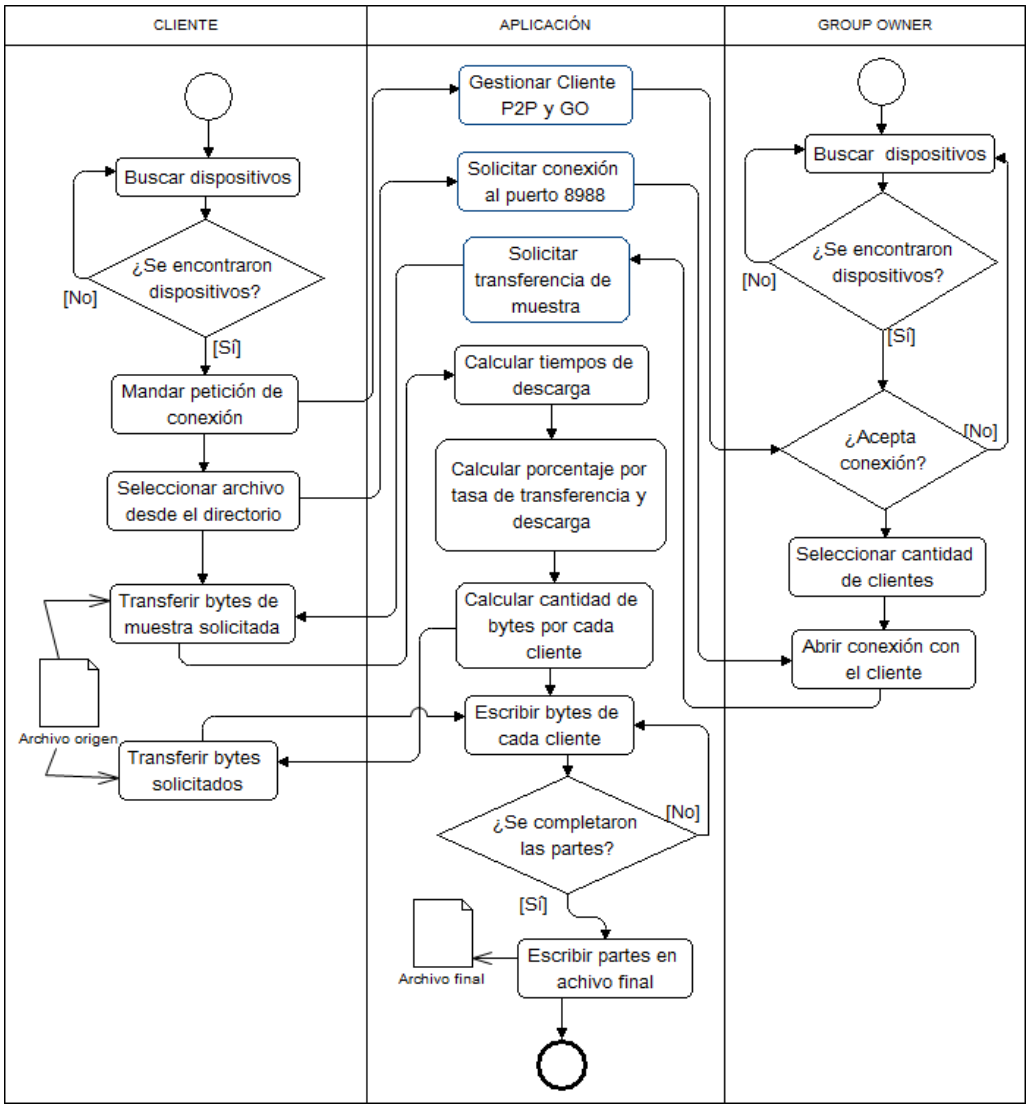

Figura 7: Procesos de Negocio futuros

5.2 Diagrama de flujo de datos (DFD)

En este punto se presentan los diagramas de flujo de datos, los que dan a conocer el flujo de información que entra y sale del sistema, la interacción del sistema con las entidades externas y el flujo de datos interno.

Para representar ampliamente el flujo de datos, un DFD consta de 3 niveles; el primer nivel, o nivel 0, es el "diagrama de flujo de datos de contexto" (DFD de contexto), que presenta una visión más general del sistema con la interacción de las entidades externas y el sistema; el nivel 1 es el "diagrama de flujo de datos superior" (DFD superior), donde se detallan los procesos internos del sistema que describen el proceso principal; y el nivel 2 "diagrama de flujo de datos de detalle" (DFD de detalle), es el que especifica cada uno de los procesos detallados en el DFD Superior.

5.2.1 DFD de contexto: Nivel 0

En este primer nivel se presenta un solo proceso, el "Proceso 0", llamado "Gestionar transferencia de archivos", el cual representa al sistema en general. Se muestra el intercambio de datos de las entidades externas, en este caso "cliente" y "Group Owner", con el sistema, omitiendo los flujos internos que se manejan en este y permitiendo observar los datos que entran y salen de este. Este DFD se detalla en la figura 8.

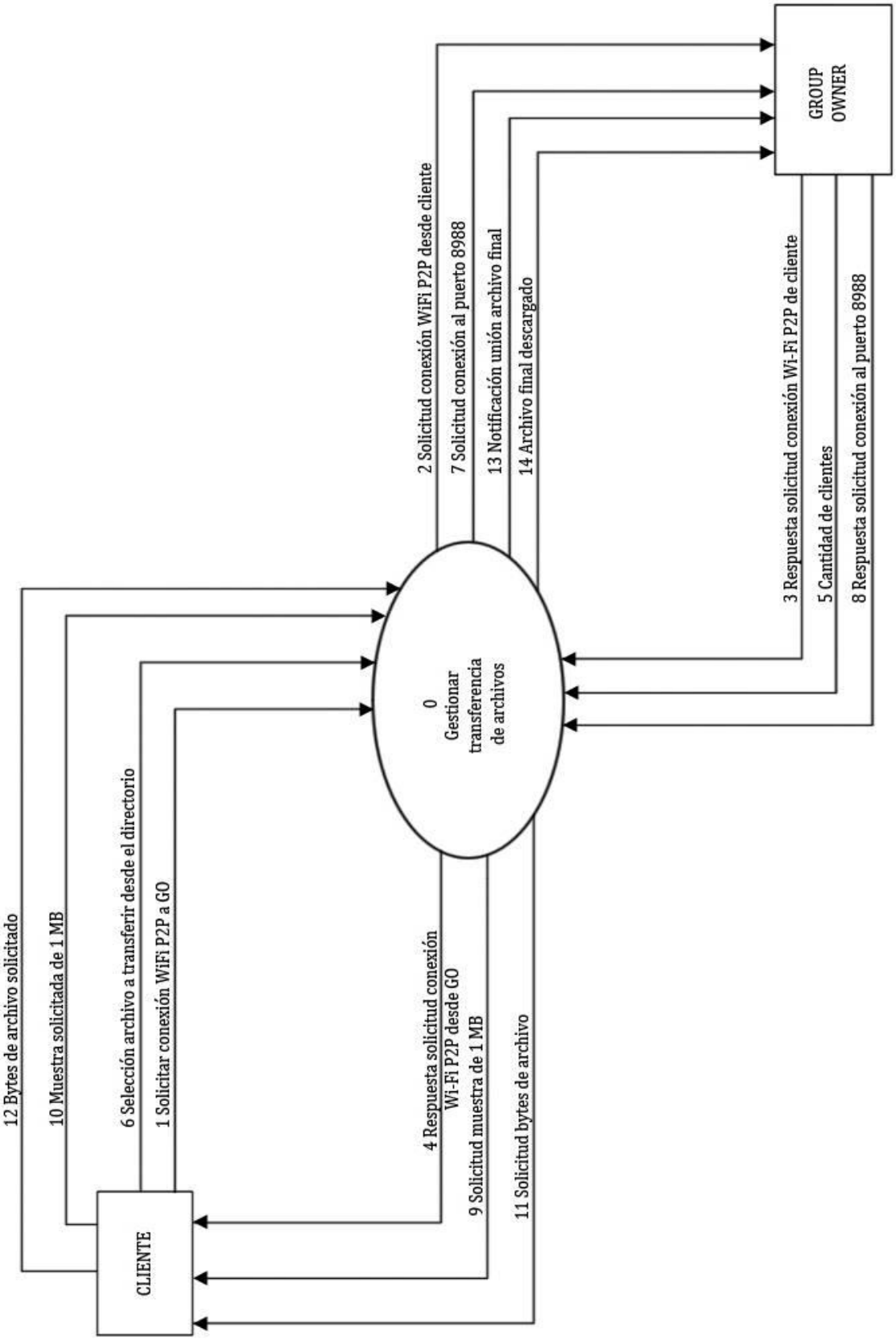

Figura 8 : DFD de Contexto

5.2.2 DFD superior: Nivel 1

Este diagrama, representado en la figura 9, muestra todos los procesos principales que describen al DFD de contexto.

1. **Gestionar Grupo P2P:** Este proceso permite gestionar la conexión del cliente y el GO (Group Owner), mediante Wi-Fi Direct. El cliente se encarga de solicitar al sistema, mediante la interfaz gráfica, la creación de un Grupo P2P o la unión a un grupo ya creado, siendo el GO el encargado de la formación de este grupo.
2. **Gestionar detalles del archivo:** Este proceso recibe el archivo seleccionado por el cliente para su transferencia y se encarga de extraer los datos necesarios para realizar la descarga.
3. **Gestionar bytes del archivo:** Este proceso se encarga de calcular los porcentajes de descarga del archivo y transformarlos a la cantidad de bytes finales a descargar desde cada cliente.
4. **Gestionar transferencia de datos:** Este proceso es el encargado de conectar el cliente con el GO a través de un socket del puerto 8988, para permitir la transferencia de datos. También calcula los tiempos de transferencia de la muestra y realiza las descargas del archivo al GO.

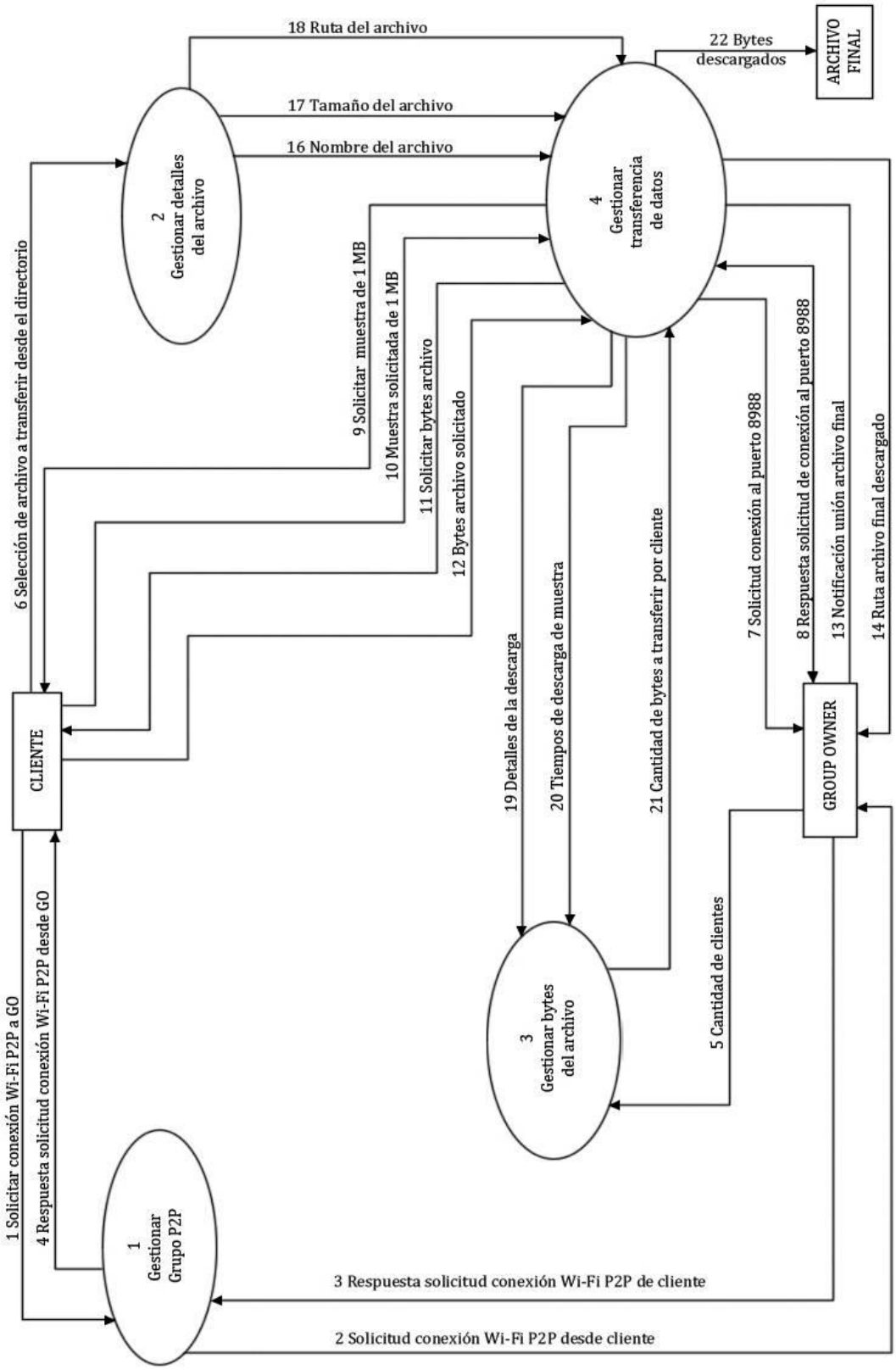

Figura 9: DFD Superior

5.2.3 DFD de detalle: Nivel 2

En el DFD de detalle, cada proceso descrito en el DFD Superior se divide en subprocesos, con el fin de detallar aún más los procesos del sistema.

Proceso 1

Se detalla el proceso 1 "Gestionar Grupo P2P", representado en la figura 10.

1.1 *Gestionar Cliente*: Este proceso permite establecer el rol de cliente dado que ha solicitado la creación de un Grupo P2P o conectarse al GO (Group Owner).

1.2 *Gestionar Group Owner*: Dado que se ha gestionado y establecido el cliente, se crea un Grupo P2P con el Dispositivo de destino como GO o dueño del grupo.

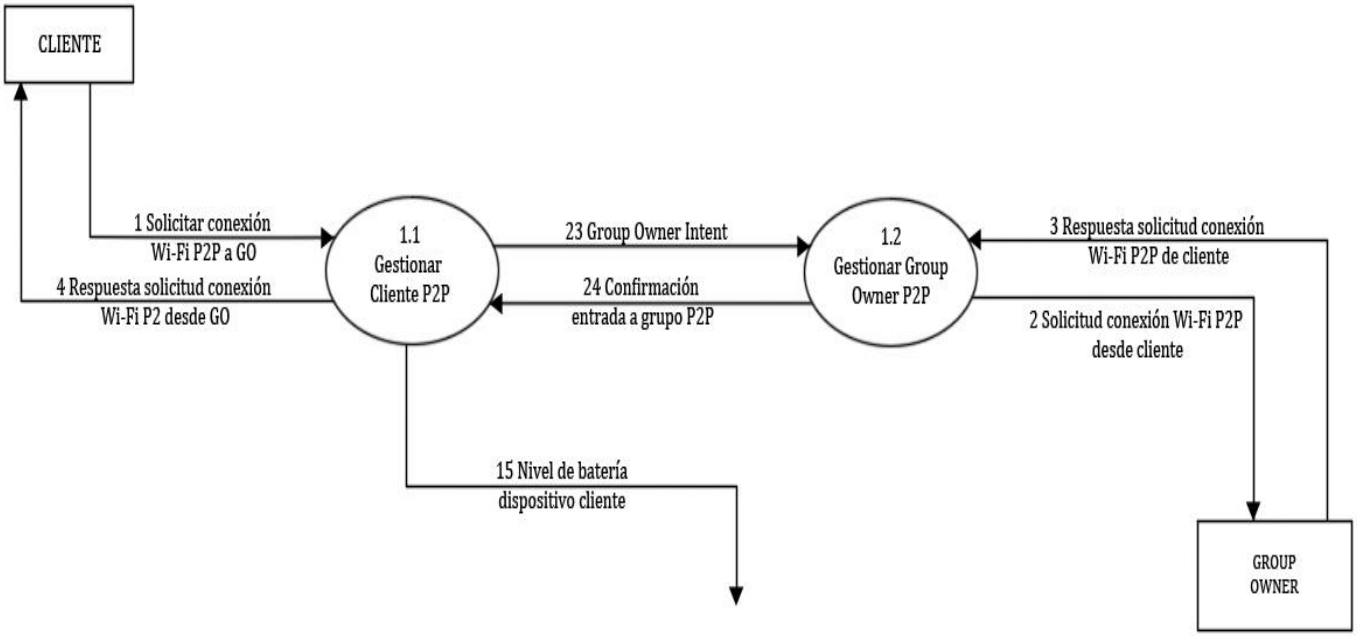

Figura 10: DFD de detalle proceso 1

Proceso 2

Se detalla el proceso 2 "Gestionar detalles del archivo", representado en la figura 11.

2.1 Comprobar archivo: Este proceso se encarga de verificar que el archivo escogido por el cliente para la transferencia sea válido y pueda ser transferido por el cliente.

2.2 Extraer detalles del archivo: Este proceso permite la identificación de las distintas partes del archivo, como; el nombre, tamaño y ruta del archivo.

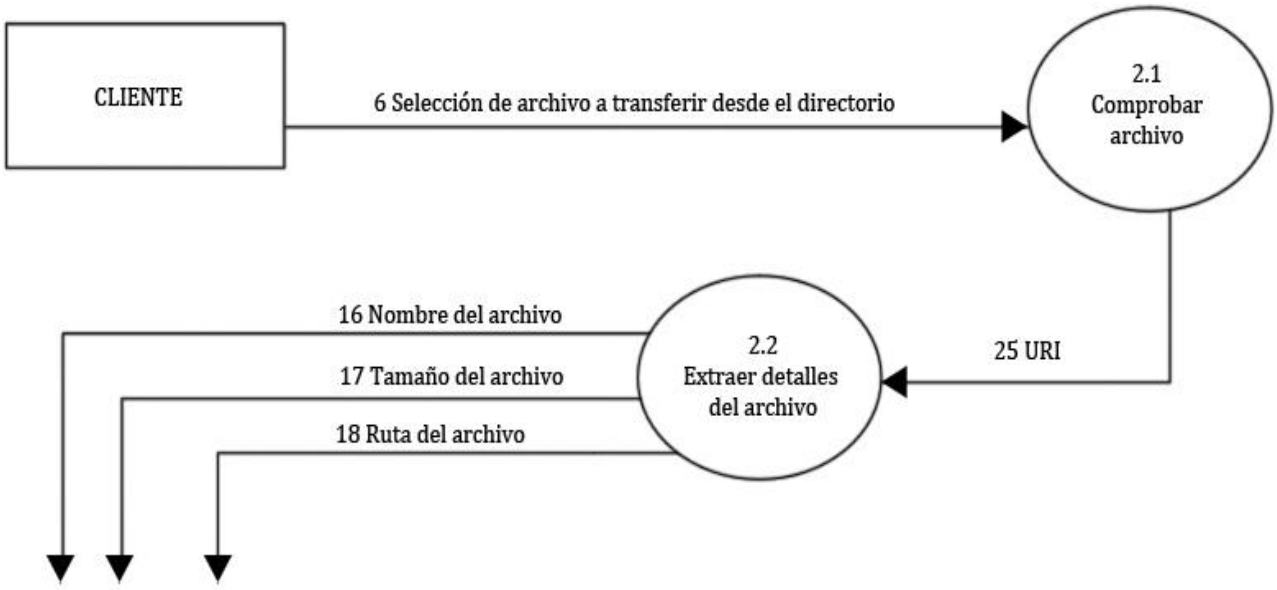

Figura 11: DFD de detalle proceso 2

Proceso 3

Se detalla el proceso 3 "Gestionar bytes del archivo", representado en la figura 12.

3.1 Calcular porcentaje de bytes por tasa de transferencia: Este proceso permite recibir los tiempos de descarga de la muestra de 1 MB y calcular con ellos el porcentaje de bytes a descargar desde cada cliente.

3.2 Calcular porcentaje de bytes por nivel de batería: Este proceso recibe el nivel de batería (1 a 100) del dispositivo cliente y con ello calcula el porcentaje de bytes a descargar con respecto a su nivel de batería.

3.3 *Calcular cantidad de bytes a descargar:* Este proceso se encarga de recibir los porcentajes de bytes a descargar dependiendo de la tasa de transferencia y nivel de batería y con ellos calcula la cantidad total a transferir desde cada cliente.

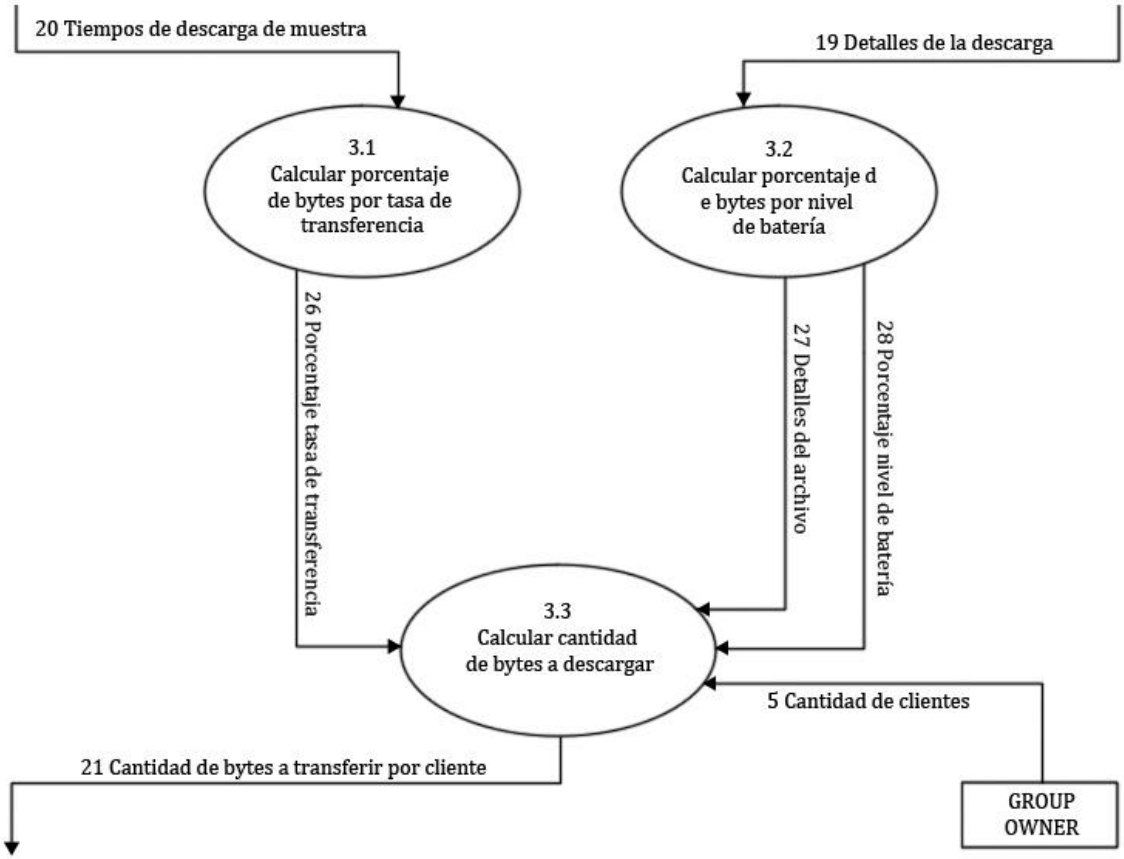

Figura 12: DFD de detalle proceso 3

Proceso 4

Se detalla el proceso 4 "Gestionar transferencia de datos", representado en la figura 13.

4.1 *Solicitar conexión con el Group Owner:* Este proceso se encarga de abrir la conexión al puerto 8988 del GO (Group Owner), creando un nuevo socket para la transferencia de datos, con el puerto que el GO le asigne a cada cliente.

4.2 *Gestionar detalles de la descarga:* Este proceso permite gestionar los detalles para asignarlos a la descarga de las muestras.

4.3 *Gestionar descarga de muestra y medición:* Este proceso permite la transferencia de la muestra al cliente y calcula el tiempo que demora en cada descarga al GO (Group Owner).

4.4 *Gestionar descarga de datos:* Este proceso es el encargado de recibir la cantidad de bytes finales a descargar, además de solicitar y recibir los bytes finales a transferir desde cada cliente. Los bytes recibidos los almacena en el archivo de destino que se encuentra en el GO.

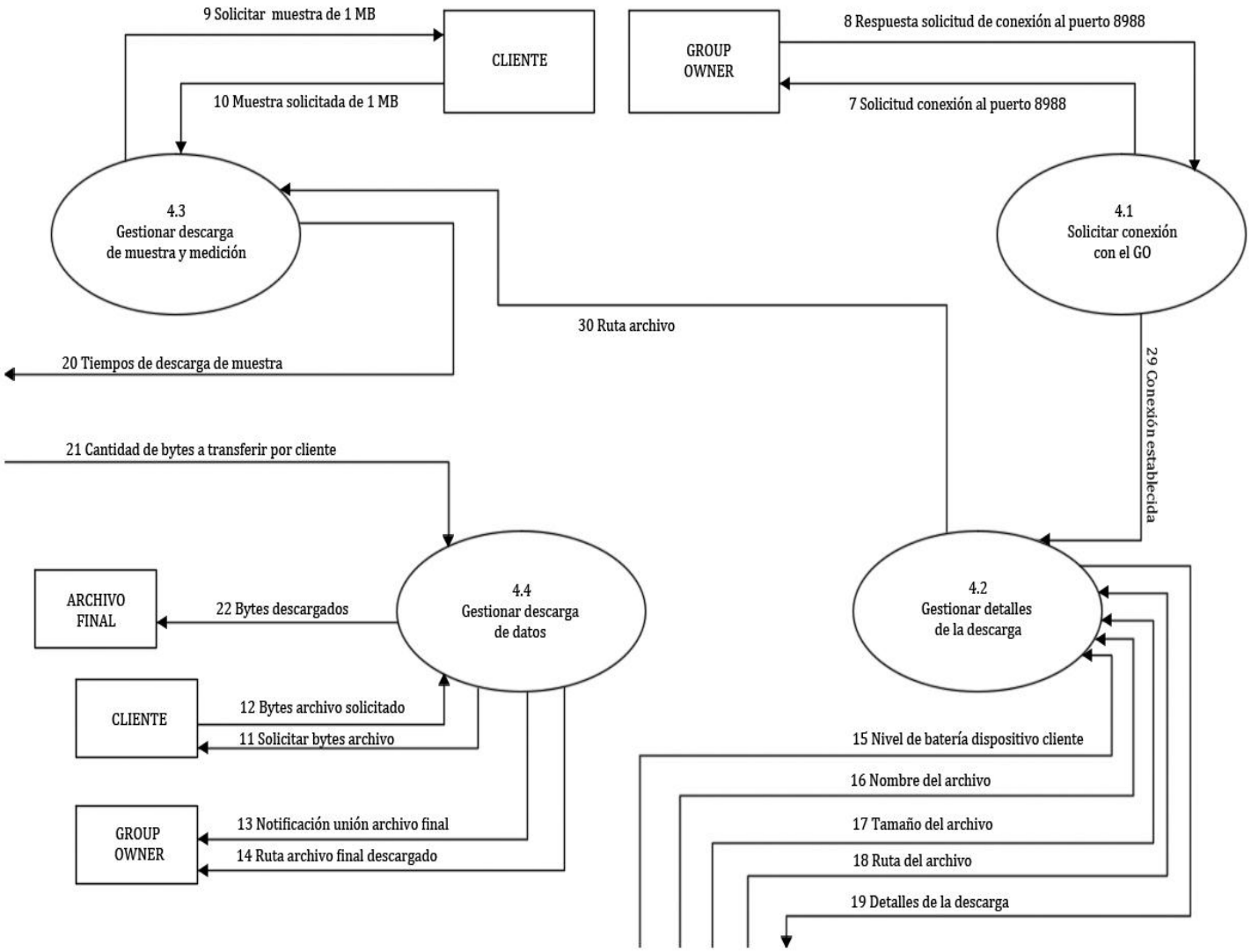

Figura 13: DFD de detalle proceso 4

6 UBBIFI DIRECT

En este punto se especifican los aspectos generales de la aplicación "UBBiFi Direct", la cual ha sido creada con la tecnología Wi-Fi Direct.

6.1 Tipos de usuarios

En la aplicación se pueden distinguir 2 tipos de usuarios:

Usuario (rol)	Características
General	✓ Dispositivo que no ha asumido rol aún.
Group Owner (GO)	✓ Dispositivo que actúa como receptor de los archivos transferidos. ✓ No puede solicitar conexión a otro usuario hasta volver a ser usuario General.
Cliente	✓ Dispositivo que solicita la conexión al usuario Group Owner, para poder transferir archivos a este. ✓ No puede conectarse a otro usuario cliente. ✓ No puede conectarse a otro GO mientras ya pertenezca a un grupo formado.

Tabla 3: Tipos de usuario

6.2 Conexión de dispositivos

La aplicación permite conectar varios dispositivos entre sí a través de la tecnología Wi-Fi Direct, formando un Grupo P2P, como está explicado en el punto "3.4 Wi-Fi Direct".

La interfaz de usuario general, como se muestra en la figura 14, muestra el estado del cliente dependiendo de la conexión con el GO (disponible, conectando, conectado, fracaso, no disponible, desconocido), y el nombre del dispositivo. En la barra superior se visualiza el ícono de búsqueda de dispositivos y el ícono para cerrar la aplicación.

Los dispositivos que deseen conectarse a un GO, deben iniciar la búsqueda de dispositivos desde la barra superior, lo que permite detectar los dispositivos cercanos que cuenten con la tecnología Wi-Fi Direct. La figura 15 muestra el resultado de esta acción.

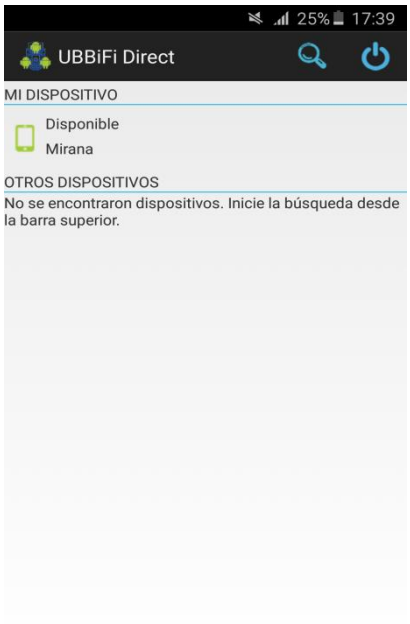

Figura 14: Interfaz de usuario general

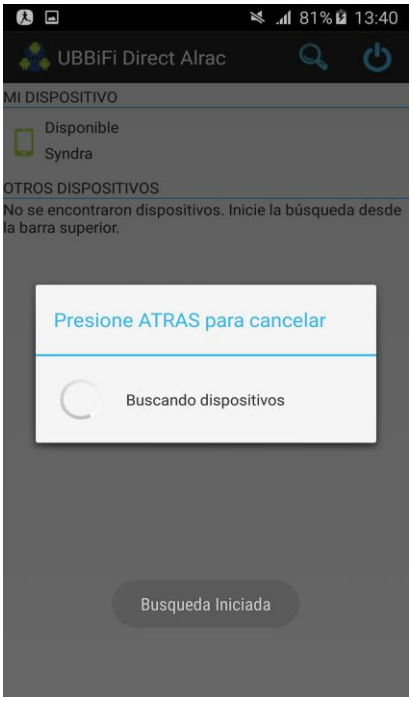

Figura 15: Búsqueda de Dispositivos

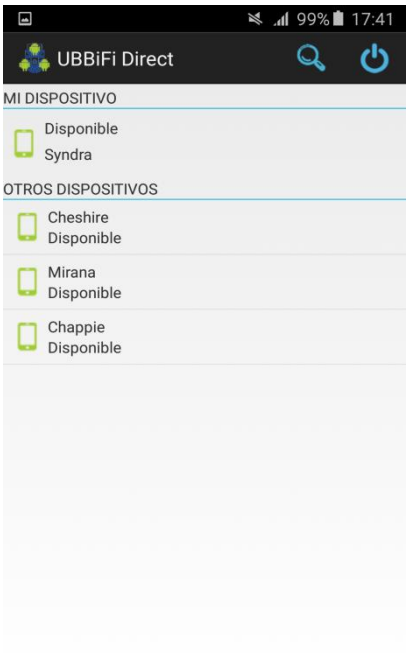

Figura 16 : Lista de dispositivos encontrados

Luego de generar la lista de dispositivos (figura 16), se selecciona el dispositivo que se desee establecer como Group Owner (figura 17). El botón "Conectar" permite mandar una solicitud de conexión a dicho dispositivo. La aplicación permite que el usuario que pide la conexión sea definido como cliente (figura 18), ya que el valor del GO Intent (identificador de negociación de GO) así lo define. El GO solo debe esperar a la petición de conexión (figura 19). El máximo de clientes que pueden pedir petición de conexión es de 4, ya que los dispositivos con los que se realizó este proyecto solo aceptan hasta este número de conexiones.

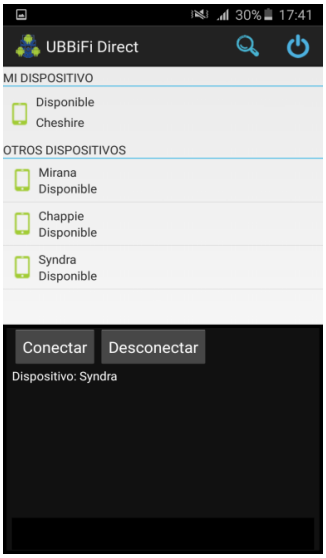

Figura 17: Selección de dispositivo a conectar

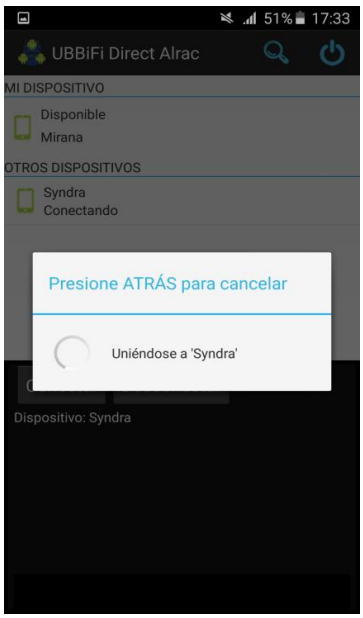

Figura 18: Solicitud de conexión por el cliente

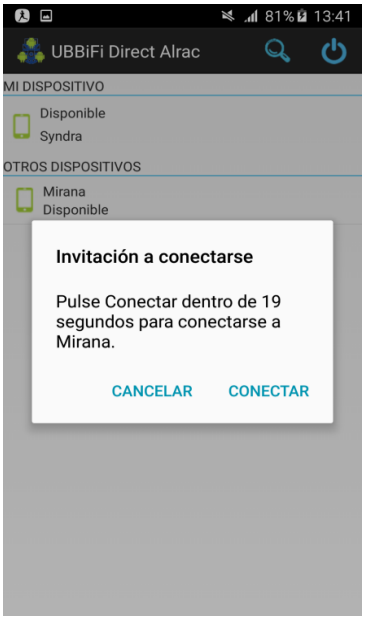

Figura 19: Petición de conexión del cliente al GO

6.3 Selección archivo

Cuando se han establecido los roles de cliente y de GO (Group Owner), ambas interfaces de usuario se modifican. Luego de que el GO seleccione la cantidad de clientes que envían el archivo, los clientes deben seleccionar el archivo a enviar. Por ejemplo, en la figura 20, se ha seleccionado que 3 dispositivos clientes enviarán una archivo. Independiente de la cantidad de clientes conectados al Grupo P2P, solo podrán enviar archivos la cantidad de clientes seleccionada por el GO.

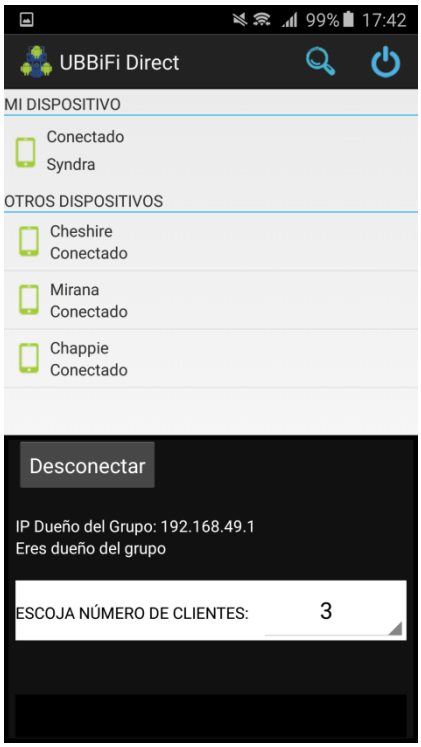

Figura 20: Selección número de clientes en el GO

Los clientes tendrán la opción de acceder al directorio de archivos mediante el botón "Buscar Archivo". Al seleccionar un archivo, se pedirá una confirmación si es ese el archivo que desea enviar (figura 21).

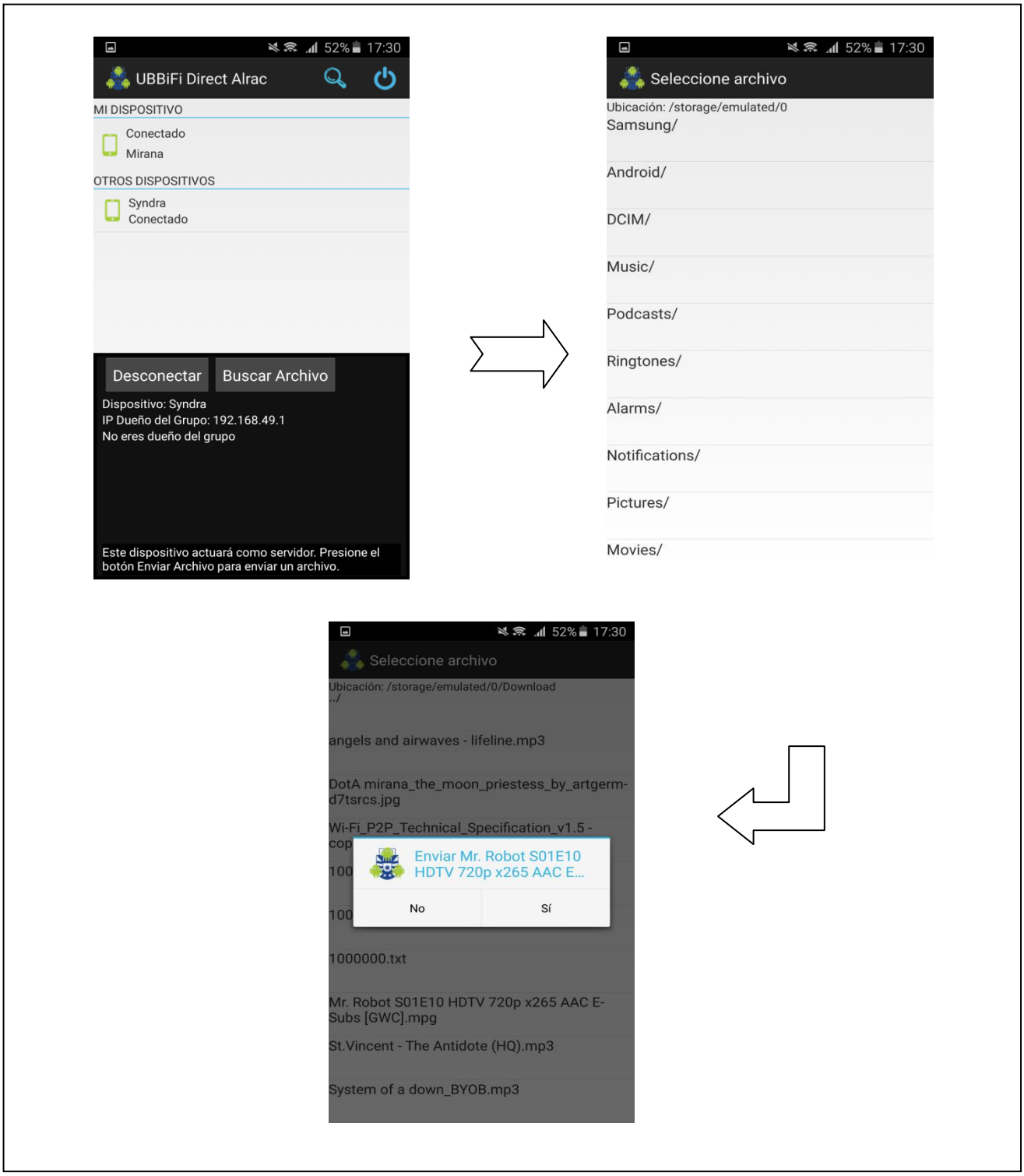

Figura 21: Selección y envío de archivo

Cuando se haya confirmado la selección del archivo, se iniciará la descarga una muestra del archivo y la ejecución de la técnica de descarga como se explica a continuación.

6.4 Técnica propuesta: Descarga en Paralelo Adaptativa P2P

En este punto se detalla una técnica de descarga en paralelo adaptada al sistema P2P (Wi-Fi Direct), que llamaremos *Descarga en Paralelo Adaptativa P2P*. En esta técnica se calcula un porcentaje a descargar de acuerdo al tiempo de transferencia de una muestra del archivo entre los dispositivos y al nivel de batería de los clientes. Luego de haber obtenido estos valores, se hace una combinación de estos dos porcentajes para obtener una óptima cantidad de bytes de transferencia.

Cada cliente debe transferir una parte del archivo total, llamada "muestra". Mientras más demore en enviarse dicha muestra, menos bytes del archivo debe transferir ese cliente. Este tamaño dependerá del tiempo en que demore la transferencia desde el cliente al GO (Group Owner), valor denominado “tasa de transferencia”.

6.4.1 Transferencia en paralelo de acuerdo a la tasa de transferencia

Si el archivo a enviar es mayor a 1 MB (Mega byte), o sea, 1048576 bytes, se toma una muestra de ese tamaño (1 MB) para calcular los tiempos que demora en transferir dicha muestra. Con estos datos se obtiene la tasa de transferencia, que son los bytes por milisegundos (B/ms) entre cada cliente-GO.

Sea tc_i el tiempo que demora el GO en descargar la muestra m desde cada cliente, la tasa de transferencia de cada cliente x_i está dada por

$$x_i = \frac{m}{tc_i} \tag{1}$$

Luego, sea L el tamaño total del archivo y $\sum x_i$ la suma de las tasas de transferencia x_i de cada cliente, se define:

$$y = \frac{L}{\sum x_i} \tag{2}$$

Para conocer la cantidad de bytes que transfiere cada cliente, se multiplica la variable y (Ecuación 2) por cada tasa de transferencia x_i, resultado que indica la cantidad de bytes a transferir por cada cliente, C_i (Ecuación 3).

$$C_i = y * x_i \tag{3}$$

Por lo tanto, la sumatoria de todas estas cantidades C_i da como resultado el total del archivo, L (Ecuación 4):

$$\sum C_i = L \tag{4}$$

Por ejemplo, se tiene un archivo de tamaño 169 MB (177209344 bytes) y considerando la muestra m de 1048576 bytes, se desea transferir un archivo desde 3 clientes.

Los tiempos de descarga tc_i desde cada cliente son:

Cliente	tc_i (milisegundos)
Cliente 1	1665 ms
Cliente 2	1050 ms
Cliente 3	580 ms

Luego, utilizando la Ecuación 1, obtenemos la tasa de transferencia x_i de cada cliente:

Cliente	x_i (bytes/milisegundos)
Cliente 1	629.77 B/ms
Cliente 2	998.64 B/ms
Cliente 3	1807.88 B/ms

Utilizando la Ecuación 2, con $L = 177209344$ y $\sum x_i = 2716.29$, se calcula la variable y, la que da como resultado:

$$y = \frac{177209344}{2716.29}$$

$$y = 65239.478$$

Finalmente, utilizando la Ecuación 3, se calcula la cantidad de bytes a descargar C_i desde cada cliente:

Cliente	C_i (bytes)
Cliente 1	41085866 B
Cliente 2	65150452 B
Cliente 3	70972723 B

La suma de cada uno de estos datos debe ser igual a la cantidad del archivo total (Ecuación 4), pero en este caso, el resultado es de 177209041 bytes. Esto es debido a que, a nivel de programación, una cierta cantidad de bytes no puede ser tratada de forma decimal. Es por esto que los 303 bytes que faltan son agregados al último cliente que envíe su parte, para así obtener el archivo completo. En este caso, $C_3 = 70973026$ bytes.

6.4.2 Transferencia en paralelo de acuerdo al nivel de batería

Dado que en la transferencia se toma en cuenta la cantidad de batería que posee el dispositivo, se establece que, mientras más bajo sea el nivel de batería, el dispositivo envía una menor cantidad de bytes, con el fin de que el nivel de batería no descienda a niveles más críticos.

Las fórmulas utilizadas para calcular los bytes descargados de acuerdo al nivel de batería son muy similares a los de la tasa de transferencia.

La aplicación se encarga de obtener los niveles de batería b_i de cada cliente. Luego se puede obtener directamente la variable que, multiplicada por cada nivel de batería b_i, nos dará la cantidad de bytes a descargar desde cada cliente.

Sea L el tamaño total del archivo y $\sum b_i$ la sumatoria de las baterías, se tiene que:

$$y = \frac{L}{\sum b_i} \tag{5}$$

Luego, la transferencia en bytes que debería realizar cada cliente de acuerdo a su nivel de batería está dada por:

$$Cb_i = y * b_i \tag{6}$$

Como fue en el caso de la tasa de transferencia, la sumatoria de todos los Cb_i debe dar el tamaño total del archivo:

$$\sum C_i = L \tag{7}$$

Por ejemplo, suponga que tiene los mismos 3 dispositivos clientes mencionados en el punto 6.4.1, cada uno con sus respectivos niveles de batería; b_1 = 50, b_2 = 10 y b_3 = 80. Se pretende descargar el mismo archivo de 169 MB (177209344 bytes).

Utilizando la Ecuación 5, con $L = 177209344$ y $\sum b_i = 140$, se calcula la variable y, la que da como resultado:

$$y = \frac{177209344}{140}$$
$$y = 1265781.03$$

Luego, utilizando la Ecuación 6, se obtienen los bytes descargados desde cada cliente:

Cliente	Cb_i (bytes)
Cliente 1	63289051 B
Cliente 2	12657810 B
Cliente 3	101262482 B

Si se tomara en cuenta solo el nivel de batería de los dispositivos cliente, cada uno de los estos debería transferir la cantidad de bytes que se mostraron anteriormente. La mayor carga de bytes es asignada al cliente 3 debido a que es el que cuenta con el mayor nivel de batería, a diferencia del cliente 2, que descarga una mínima parte del archivo.

La sumatoria de los Cb_i da como resultado 177209343, como falta 1 byte por completar el tamaño total del archivo (177209344 bytes), este se sumará al último cliente, $Cb_3 =$ 101262483.

6.4.3 Combinación de porcentajes

Como la transferencia del archivo depende de la tasa de transferencia y del nivel de batería, cada parte descargada representa un porcentaje del total del archivo.

Siendo L el tamaño total del archivo, y teniendo los bytes que transfiere cada cliente, tanto por la tasa de transferencia C_i (Ecuación 3), como por el nivel de batería Cb_i (Ecuación 6), el porcentaje asignado a cada uno de estos valores estará dado por:

Porcentaje de tasa de transferencia

$$Pt_i = \frac{C_i}{L} \tag{8}$$

Porcentaje de nivel de batería

$$Pb_i = \frac{Cb_i}{L} \tag{9}$$

Tanto Pt_i como Pb_i son los porcentajes que debe transferir cada cliente del archivo, pero en relación a la tasa de transferencia y al nivel de batería, respectivamente. Para unificar estos valores, el nivel de batería representará un 5 % en la descarga total del archivo y la tasa de transferencia un 95 %, ya que el nivel de batería no es un factor tan importante a la hora de la de transferencia. Esto si todos los dispositivos cliente tiene sobre un 15 % de nivel de batería.

Si existe un dispositivo cuyo nivel de batería es 15 % o menor, el nivel de batería representa un 10 % en la descarga total del archivo y la tasa de transferencia un 90 %. Se le ha agregado más valor al nivel de batería, ya que, como existe un dispositivo con nivel de batería bajo 15 %, este necesita descargar menos bytes con el fin de que la batería no disminuya a niveles más críticos.

Entonces, para cada Pt_i correspondiente al porcentaje de la tasa de transferencia, y para cada Pb_i correspondiente al nivel de batería, se calcula su porcentaje respecto del total del archivo.

Para nivel de batería mayor a 15%, se utiliza un 95 % (0.95) del porcentaje de la tasa de transferencia Pt_i y un 5 % (0.05) del porcentaje de nivel de batería Pb_i:

$$P_i = Pt_i * 0.95 \; + \; Pb_i * 0.05 \tag{10}$$

Para nivel de batería igual o menor a 15%, se utiliza un 90 % (0.9) del porcentaje de la tasa de transferencia Pt_i y un 10 % (0.1) del porcentaje de nivel de batería Pb_i:

$$P_i = Pt_i * 0.9 \; + \; Pb_i * 0.1 \tag{11}$$

Cada P_i representa el porcentaje final del archivo que debe descargar cada cliente. Luego, para calcular los bytes a transferir por cada cliente, se debe multiplicar el tamaño total del archivo L por cada porcentaje P_i (Ecuación 12).

$$Tc_i = L * \; P_i \tag{12}$$

Por ejemplo. Suponga que tiene las mismas tasas de transferencias y niveles de batería desde los 3 clientes mencionados en los puntos anteriores (puntos 6.4.1 y 6.4.2) y la descarga del mismo archivo de 169 MB (177209344 bytes). Se tienen los siguientes valores:

Bytes a descargar por tasa de transferencia

Cliente	C_i (bytes)
Cliente 1	41085866 B
Cliente 2	65150452 B
Cliente 3	70973026 B

Bytes a descargar por nivel de batería

Cliente	Nivel de batería (porcentaje)	Cb_i (bytes)
Cliente 1	50 %	63289051 B
Cliente 2	10 %	12657810 B
Cliente 3	80 %	101262483 B

Se calculan los porcentajes Pt_i correspondientes a la tasa de transferencia de cada cliente, utilizando la Ecuación 8:

Cliente	Pt_i
Cliente 1	0.23
Cliente 2	0.36
Cliente 3	0.40

Lo ideal es que la suma de estas variables de 1, ya que esta suma representa el 100% del tamaño del archivo. En este caso, como falta un 0,01 para completar el 1, se agrega al último cliente. Así, $Pt_3 = 0.41$.

Luego, se calcula los porcentajes con respecto al nivel de batería, utilizando la Ecuación 9:

Cliente	Pb_i
Cliente 1	0.35
Cliente 2	0.07
Cliente 3	0.57

Igual que en el caso anterior, la sumatoria de los Pb_i es igual a 0.99. El 0.01 faltante se agrega al último cliente, dando como resultado $Pb_3 = 0.58$.

Luego, como existe un dispositivo cliente con nivel de batería bajo 15 %, el porcentaje de nivel de batería calculado anteriormente representará un 10 % de la transferencia por cada cliente y el porcentaje de la tasa de transferencia un 90 %.

Por lo tanto, utilizando la Ecuación 11, se obtiene:

Cliente	P_i
Cliente 1	$0.23 * 0.9 + 0.35 * 0.1 = 0.242$
Cliente 2	$0.36 * 0.9 + 0.07 * 0.1 = 0.331$
Cliente 3	$0.41 * 0.9 + 0.58 * 0.1 = 0.427$

Finalmente, utilizando la Ecuación 12, se calculan los bytes a transferir por cada cliente:

Cliente	Porcentaje final (porcentaje)	Tc_i (bytes)
Cliente 1	$0.242 = 24.2\ \%$	42884661 B
Cliente 2	$0.331 = 33.1\ \%$	58656292 B
Cliente 3	$0.427 = 42.7\ \%$	75668389 B

La sumatoria de los valores Tc_i da como resultado 177209342 bytes. Los 2 bytes faltantes para completar el tamaño total del archivo (177209344 bytes) son agregados al último cliente que comenzó la transferencia. Por lo tanto $Tc_3 = 75668391$ bytes.

Podemos observar que, por tasa de transferencia, los 3 clientes descargarían una cantidad similar entre ellos, pero por nivel de batería, el Cliente 2 se le asignaría una menor carga de transferencia, dado que presenta un nivel bajo de batería. Pero debido a que el nivel de batería no es un factor que influya de manera considerable en la transferencia (a diferencia de la tasa de transferencia), es que se asignan los porcentajes anteriormente establecidos. Así, al observar la tabla 4, que compara los bytes de transferencia, se puede observar como la cantidad de bytes final del cliente 2 disminuyen a favor de menor consumo de batería.

Cliente	Bytes por tasa de transferencia (bytes)	Bytes por nivel de batería (bytes)	Bytes descarga final (bytes)
Cliente 1	41085866 Bytes	63289051 bytes	42884661 bytes
Cliente 2	65150452 Bytes	12657810 bytes	58656292 bytes
Cliente 3	70973026 Bytes	101262483 bytes	75668391 bytes

Tabla 4 : Comparación de bytes

6.5 Transferencia y unión de archivos

Luego de que cada cliente haya seleccionado el archivo a transferir, se envía la "muestra" y se calculan los tiempos de transferencia de dicha muestra, valor al que llamamos "tasa de transferencia". Con este valor, más el nivel batería, se calcularán los bytes a transferir desde cada dispositivo cliente y se habilitará la transferencia automática de las partes del archivo.

Los clientes mostrarán en su interfaz el proceso de transferencia del archivo mediante una barra de progreso de 0-100 %, la cual desaparecerá cuando se haya transferido el archivo al GO (Group Owner). Como ejemplo, en la figura 22 se muestra a 3 clientes transfiriendo sus partes del archivo al GO.

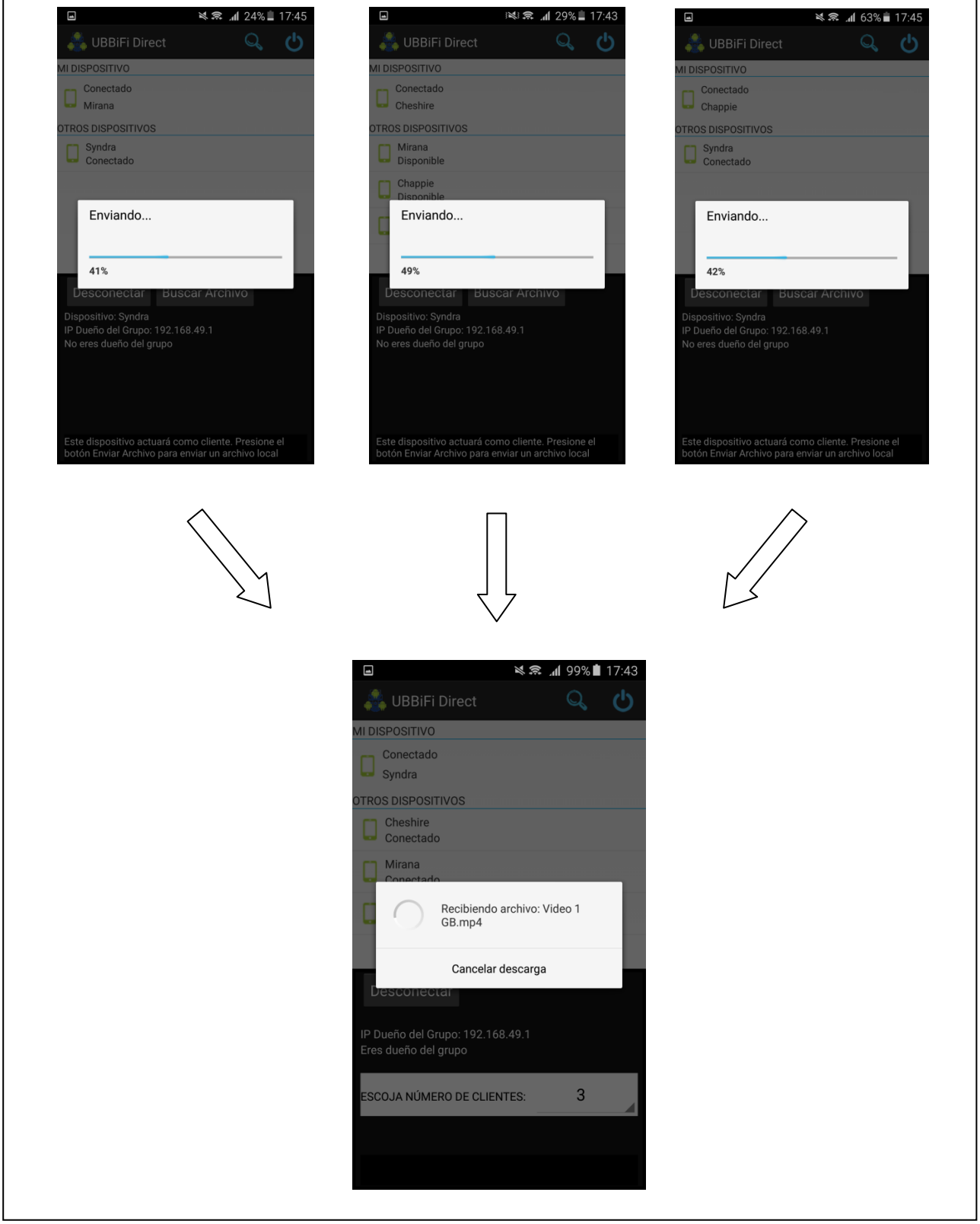

Figura 22: Transferencia en paralelo del archivo por "UBBiFi Direct"

El GO (Group Owner) puede cancelar la descarga en curso (figura 23), lo que detendrá la transferencia de todos los clientes que estén enviando partes del archivo.

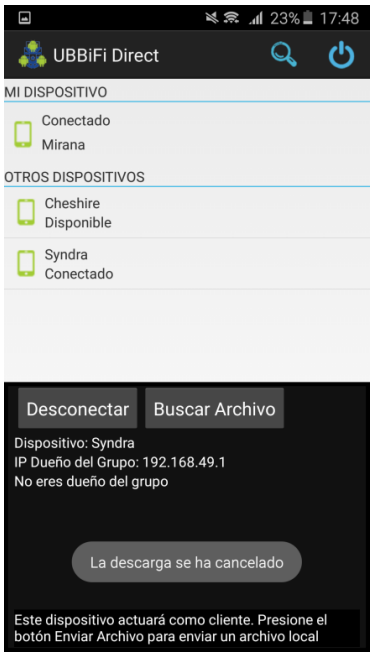

Figura 23: Notificación de transferencia cancelada

Si el GO no ha cancelado la descarga y esta se ha completado con éxito, se procederá a la unión de las partes del archivo, leyéndolas y escribiéndolas en el archivo final (figura 24).

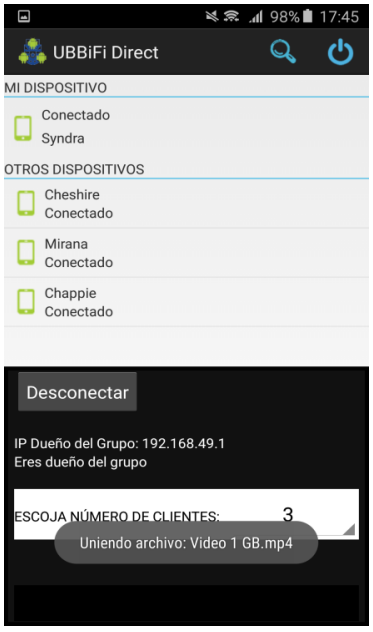

Figura 24: Notificación unión de las partes transferidas

Luego de unidas las partes, el GO (Group Owner) es notificado de que el archivo final se ha completado. La notificación se puede ver al pie de la aplicación (figura 25).

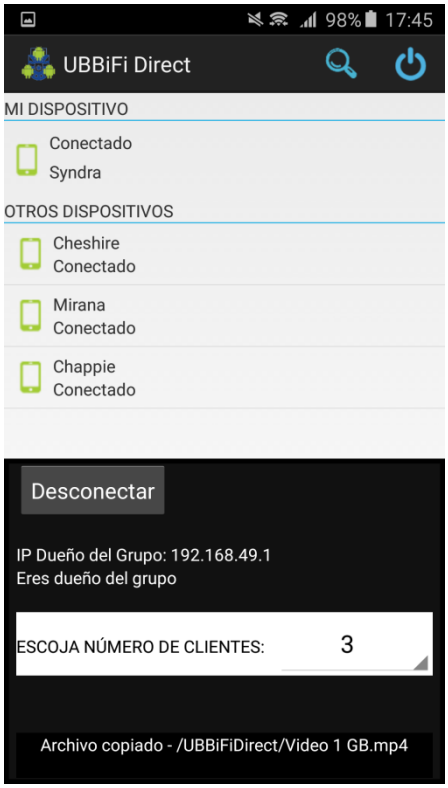

Figura 25: Notificación de creación del archivo final

Finalmente, el GO podrá disolver el Grupo P2P al presionar "Desconectar", o aceptar nuevas transferencias de archivos desde los clientes.

7 PRUEBAS

En este punto se llevan a cabo pruebas sobre la aplicación desarrollada durante este proyecto para corroborar el correcto funcionamiento de esta. El desarrollo de estas pruebas se realiza usando 3 dispositivos en total: 1 dispositivo GO (Group Owner) y 3 dispositivos clientes, posicionando los clientes a una distancia de 20 cm aproximadamente del GO.

7.1 Elementos de pruebas

- ✓ **Descarga completa del archivo**: En este elemento de prueba se verifica que el tamaño del archivo descargado es igual al tamaño del archivo original.
- ✓ **Validación número de clientes**: En este elemento de prueba se verifica que el número de clientes que envían el archivo es igual al número de Clientes definido por el GO.
- ✓ **Validación de la integridad del archivo final**: Luego de que el archivo es descargado se realiza una suma de verificación (*checksum*) con el fin de comprobar que el archivo descargado es igual al archivo original.

7.2 Especificación de las pruebas

ID	Características a probar	Objetivo de la prueba	Enfoque	Actividades de prueba	Criterios de cumplimiento
01	Resistencia	Verificar que la conexión de los dispositivos clientes con el GO sea estable durante las transferencias de las partes del archivo.	Caja negra	✓ Conectarse a un GO. ✓ Enviar archivo.	La interfaz gráfica debe mostrar que el estado del dispositivo es "Conectado" durante y después del envío.
02	Funcionalidad	Comprobar que la transferencia de las partes por cada cliente se complete correctamente.	Caja negra	✓ Conectar clientes al GO. ✓ Seleccionar archivo a enviar. ✓ Esperar que se completen las transferencias en un 100%.	La app debe notificar al GO cuando se hayan transferido todos los archivos.
03	Validación	Se cancelan las transferencias con éxito desde el GO.	Caja negra	✓ Mientras se realizan las transferencias se cancela la descarga desde el GO.	La app debe notificar a cada cliente y al GO que se ha cancelado la descarga.
04	Funcionalidad	La app debe permitir transferir nuevos archivos luego de cada descarga.	Caja negra	✓ Luego de que se haya completado una descarga, seleccionar nuevamente "Buscar Archivo" para iniciar otra transferencia.	La app debe mostrar nuevamente la barra de progreso para la nueva transferencia.
05	Funcionalidad	La app debe disolver correctamente el Grupo P2P al desconectar el GO.	Caja negra	✓ Habiendo clientes conectados al GO y sin haber iniciado una transferencia, presionar "Desconectar" en el dispositivo GO.	Todos los dispositivos deben volver a la interfaz de usuario general *(Figura)*
06	Integridad	Comprobar que la suma de verificación (*Checksum*) del archivo descargado es igual al *checksum* del archivo original.	Caja negra	✓ Una vez descargado el archivo y unidas las partes, mediante la app "ES File Explorer" se compara la *checksum* con la del archivo original.	El *checksum* del archivo descargado debe ser idéntico al del archivo o original.

Tabla 5: Especificación de las pruebas

7.2.1 Resultados de las pruebas

ID	Datos de entrada	Salida esperada	Salida obtenida	Éxito / Fracaso
01	No hay datos de entrada.	Estado de los dispositivos involucrado se mantenga como "Conectado"	Todos los dispositivos se mantuvieron como "Conectados" durante la transferencia y al finalizar esta.	Éxito
02	Selección "Buscar archivo" en cada dispositivo cliente y seleccionar un archivo del directorio.	Notificación de archivo copiado en GO.	"Archivo copiado */direcciónarchivo*"	Éxito
03	Selección "Buscar archivo" en cada dispositivo cliente y seleccionar un archivo del directorio.	Notificación en el dispositivo cliente que indique que la descarga se ha detenido.	"La descarga se ha cancelado"	Éxito
04	Selección "Buscar archivo" en cada dispositivo cliente y seleccionar un archivo del directorio.	Barra de progreso de la transferencia del archivo.	"Enviando..."	Éxito
05	Presionar "Desconectar" en el dispositivo GO.	Interfaz de Usuario General	Interfaz de Usuario General	Éxito
06	Selección "Buscar archivo" en cada dispositivo cliente y seleccionar un archivo del directorio.	e729cc1b413806ee59167 f39b4b6f315	e729cc1b413806ee59167f39b4 b6f315	Éxito

Tabla 6: Resultado de las pruebas

8 EXPERIMENTOS

En esta sección se realizan distintos experimentos para evaluar el rendimiento de la técnica aplicada, donde se transfieren archivos en diferentes escenarios: Distintos tamaños del archivo a enviar; distinta cantidad de clientes que transfieren un archivo; distintos tamaños de muestra; transferencias a distintas distancias. También se realiza una comparación con dos de los otros medios de transferencia nombrados anteriormente, Bluetooth y SBeam (aplicación NFC de los dispositivos utilizados).

Los distintos datos que se evalúan en el experimento son:

- ✓ *Tiempo total de transferencia*: Tiempo total de transferencia del archivo desde el cliente al GO (Group Owner), que se inicia desde el envío de la "muestra" desde todos los dispositivos cliente, hasta el fin de la unión de las partes del archivo y la formación del archivo final.
- ✓ *Desbalance de carga*: Es la diferencia de tiempo de transferencia del cliente que terminó último la transferencia de su parte con el tiempo de transferencia del cliente que terminó primero. Mientras más cercano a cero sea el valor de este desbalance, se observa un mejor nivel de transferencia en paralelo.

Para la realización de la mayoría de los experimentos, los dispositivos clientes se encuentran a 20 cm. de distancia del GO, a menos que se indique lo contrario, sin algún elemento entre ellos que pudiese intervenir en la velocidad de transferencia.

8.1 "Tiempos" vs. "Número de clientes" con muestra de 1 MB

Este gráfico muestra los resultados de los tiempos de transferencia de un archivo de 1 GB, realizado por distintos clientes, con una muestra de 1 MB.

Número de clientes	Tiempo total (segundos)
1	102.85 s
2	94.81 s
3	97.21 s
4	90.48 s

Tabla 7: Número clientes/Tiempo de transferencia

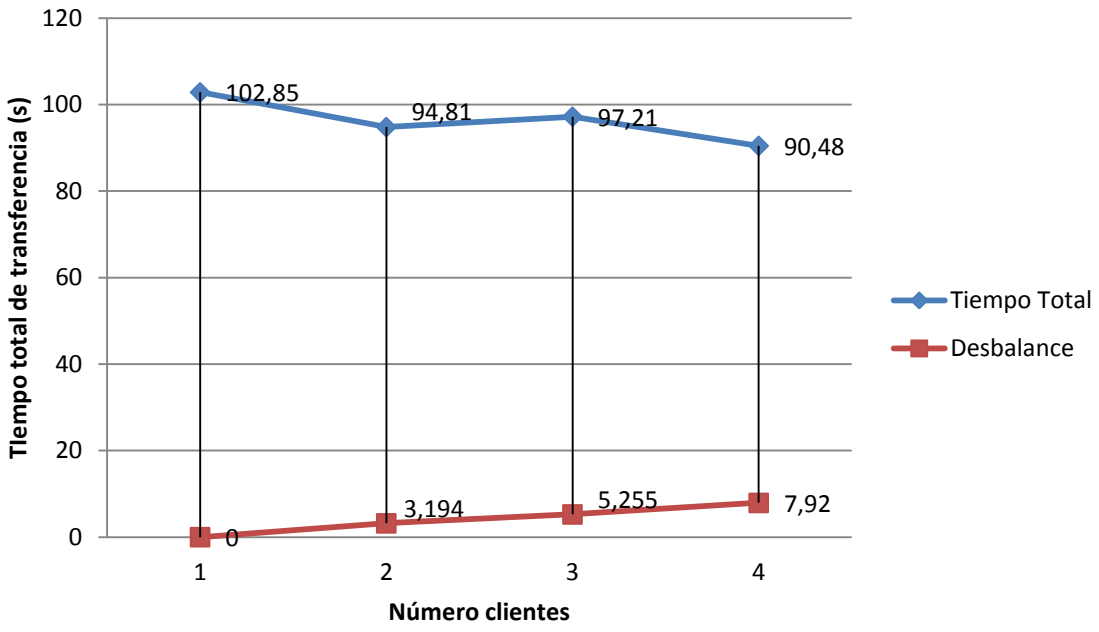

Figura 26: Gráfico "Tiempos" vs. "Número de clientes" con muestra de 1 MB

En el gráfico de la figura 26 se puede observar que a medida que aumenta la cantidad de clientes que transfieran el archivo, el tiempo de transferencia disminuye, excepto con la cantidad de 3 clientes, ya que se ve aumentado en relación a la transferencia realizada por 2 clientes. Con esto se logra en parte el objetivo de la descarga en paralelo, ya que mientras más transferencias concurrentes se ejecuten, en este caso 4 clientes, menos tiempo demora la transferencia total del archivo, con un promedio de 90 segundos. Lamentablemente, esta cantidad de clientes es la que obtuvo un nivel de desbalance más alto. Debido al corto tamaño de la muestra, no se pudo hacer una estimación de tasa de transferencia que se adaptara a todos los clientes, ya que presentaron tiempos de transferencia muy distintos entre sí.

8.2 "Tiempos" vs. "Tamaño archivo"

En este gráfico se realizó con la cantidad óptima de 4 clientes para la transferencia, como se señaló en el gráfico anterior. Se realizaron experimentos con distintos tamaños de archivos, obteniendo los tiempos de transferencias de cada uno de ellos.

Tamaño archivo (megabytes)	Tiempo total (segundos)
107 MB	6.06 s
574 MB	48.58 s
1024 MB	91.98 s
1587.2 MB	152.41 s
2099.2 MB	202.70 s

Tabla 8: Tamaño archivo/Tiempo de transferencia

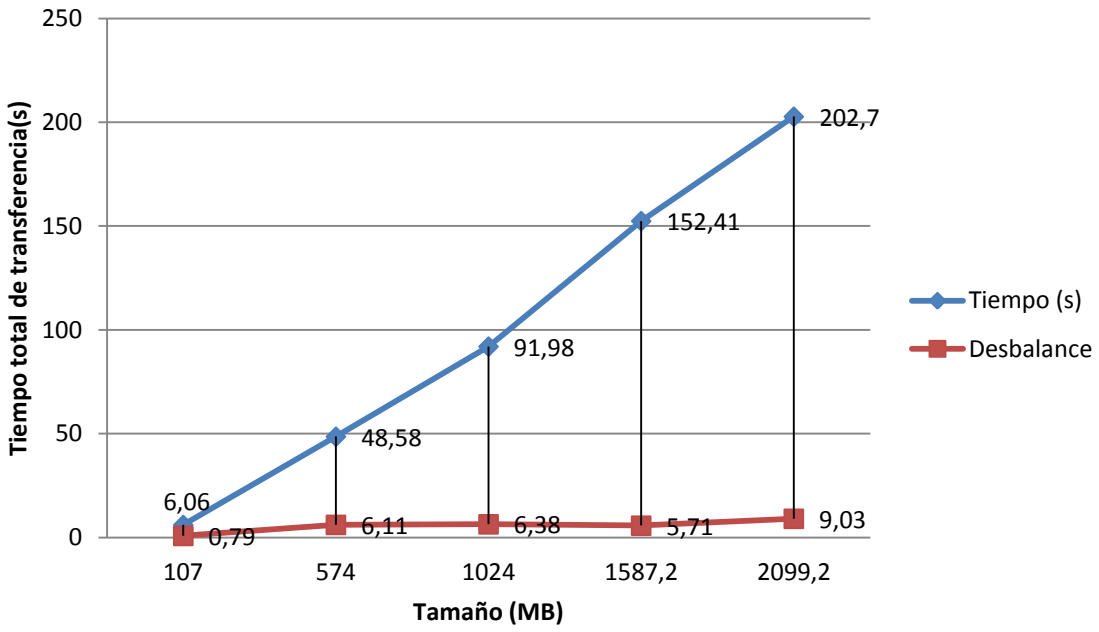

Figura 27: Gráfico "Tiempos" vs. "Tamaño de archivo"

Analizando el gráfico de la figura 27, se puede concluir que la técnica de descarga en paralelo presenta una proporcionalidad casi directa al momento de aumentar el tamaño del archivo transferir. Al igual que el primer gráfico, a medida que aumenta el tamaño del archivo, se puede observar un aumento en el desbalance ya que también se realizó el experimento con la muestra de 1 MB.

8.3 Comparación tiempos de transferencia

En este gráfico se comparan los tiempos de transferencia de la aplicación “UBBiFi Direct” con los 2 tipos de tecnologías más conocidos para la transferencia de archivos, Bluetooth y NFC (Near Field Communication). Se realizan mediciones para distintos tamaños de archivos con el fin de obtener una visión general del rendimiento de cada uno de ellos.

Tamaño (megabytes)	Tiempo de Bluetooth (segundos)	Tiempo S-Beam (NFC) (segundos)	UBBiFi Direct (segundos)
10 MB	48 s	54 s	0.6 s
50 MB	240 s	270 s	2.83 s
100 MB	480 s	540 s	5.66 s
500 MB	2400 s	2700 s	42.32 s
1024 MB (1 GB)	4915.2 s	5529.6 s	91-98 s
1536 MB (1.5 GB)	7372.8 s	8294.4 s	147.49 s
2048 MB (2 GB)	9839.4 s	11059.2 s	197.76 s

Tabla 9: Comparación tiempos de transferencia

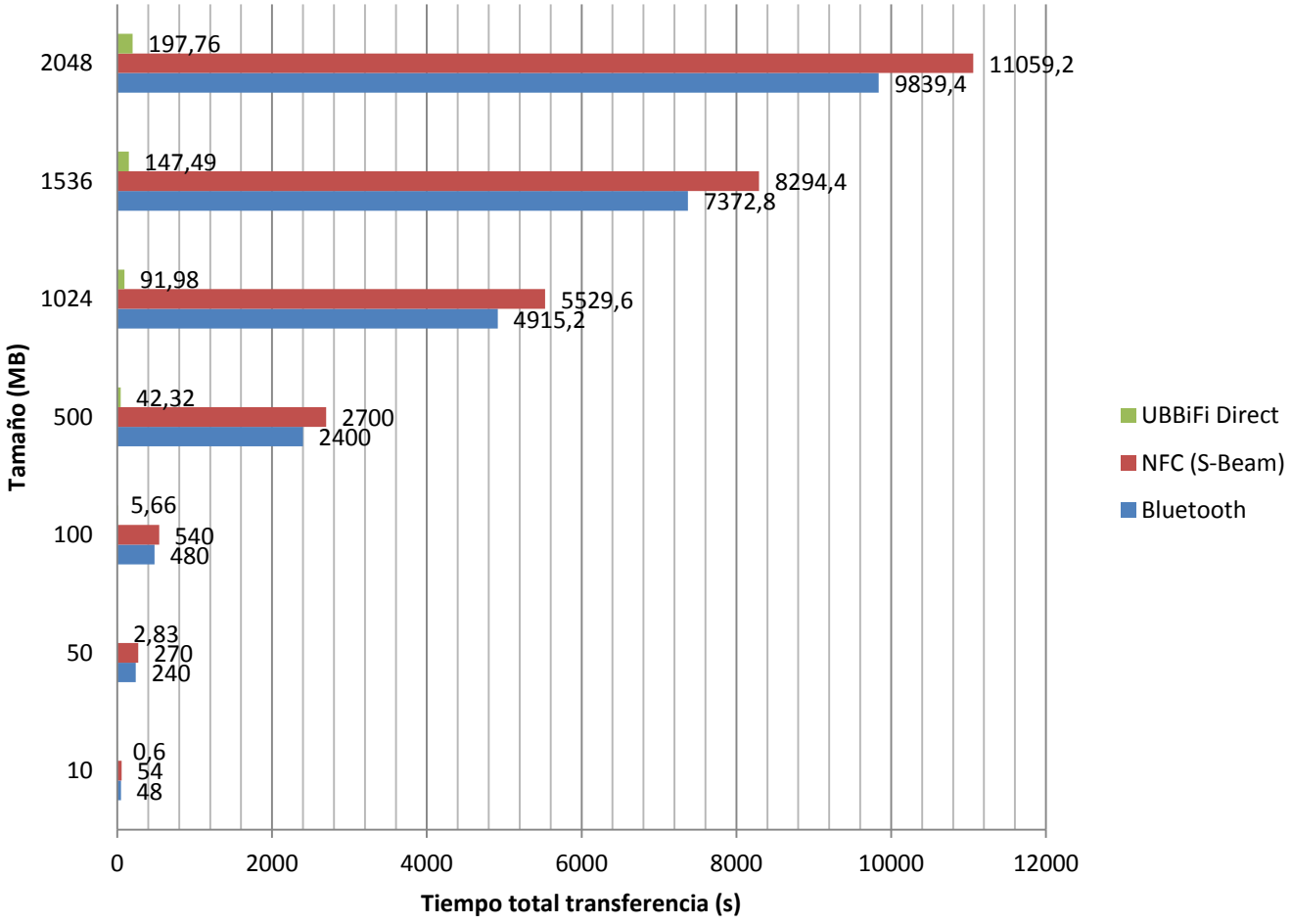

Figura 28: Gráfico "Comparación de tiempos de transferencia"

Como se puede apreciar en el gráfico de la figura 28, Bluetooth y NFC cuentan con una velocidad de transferencia muy similar, pero comparado con la aplicación realizada, esta presenta una velocidad de transferencia muy alta, ya que podemos observar que para un archivo de casi 2 GB, Bluetooth demora cerca de 9889 segundos, o sea, casi 2.5 horas; NFC demora cerca de 3 horas y con la aplicación UBBiFi Direct, la descarga se completa en tan solo 197 segundos, es decir, solo 3 min. Se puede concluir que UBBiFi Direct posee una gran superioridad en cuanto a la velocidad de transferencias de archivos y tiene una mayor ventaja, que es la descarga en paralelo.

8.4 "Tiempos" vs. "Tamaño muestra"

En esta sección se miden los tiempos de transferencia de archivos de 500 MB, 1 GB y 2 GB, desde 4 clientes, cambiando el tamaño de la muestra utilizando valores de 1 MB, 20 MB, 30 MB, 50 MB, 75 MB y 100 MB.

Al ir aumentando el tamaño de esta muestra, la tasa de transferencia puede medirse con mayor exactitud. Es por esto que se puede observar que los tiempos de desbalance (diferencia entre el tiempo del cliente que demoró más en transferir el archivo y el tiempo del cliente que demoró menos transferir) disminuyen a medida que el tamaño de la muestra aumenta. Sin embargo, esto conlleva a que el tiempo total de transferencia aumente, ya que este incluye el tiempo en que se descarga dicha muestra (que al ir aumentando, demora más en ser descargada).

8.4.1 Archivo de 500 MB

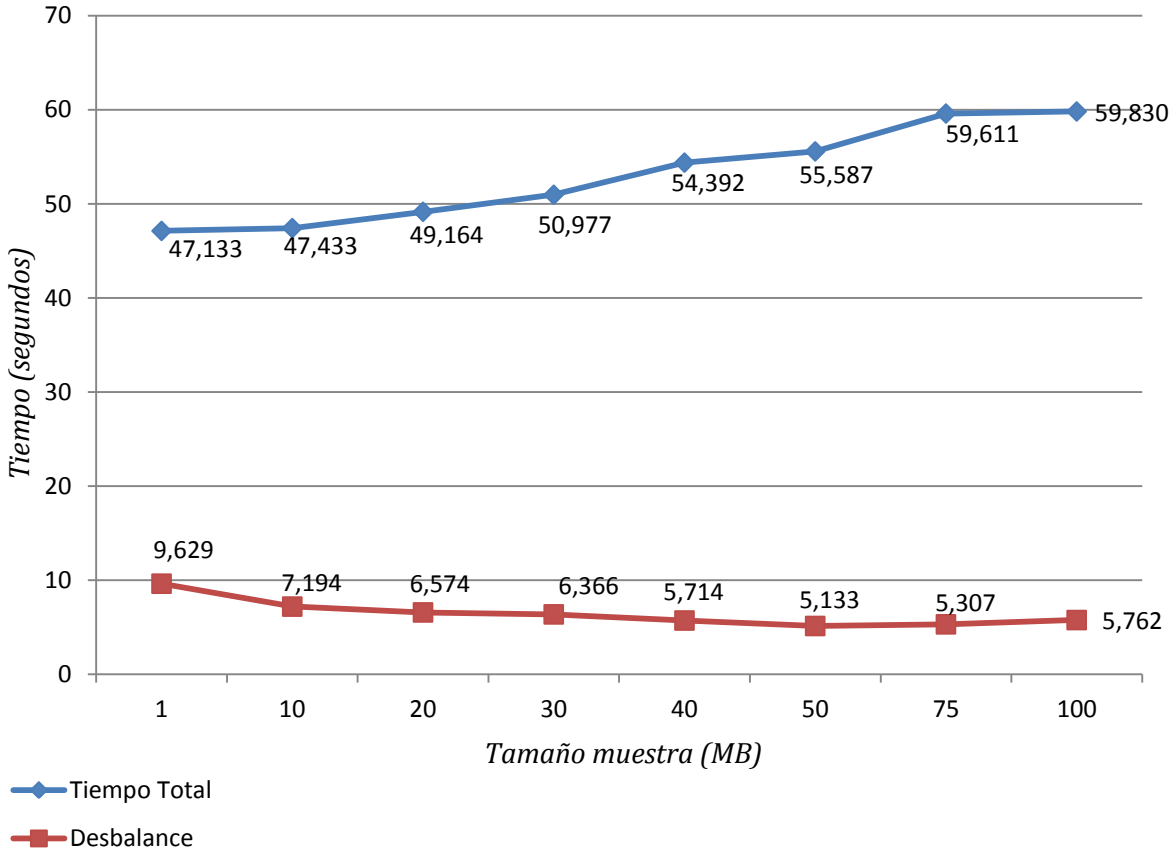

Figura 29: Gráfico "Tiempos" vs. "Tamaño muestra" con archivo de 500 MB

En el gráfico de la figura 29 se puede observar que en la muestra de 10 MB es donde el desbalance tiene su disminución más significativa, mejorando 2.435 segundos de 1 MB a 10 MB, para mantenerse ligeramente más abajo luego de esto. También, en muestras de mayor tamaño, el aumento de tiempo no compensa la disminución del desbalance en estas.

8.4.2 Muestra de 1 GB

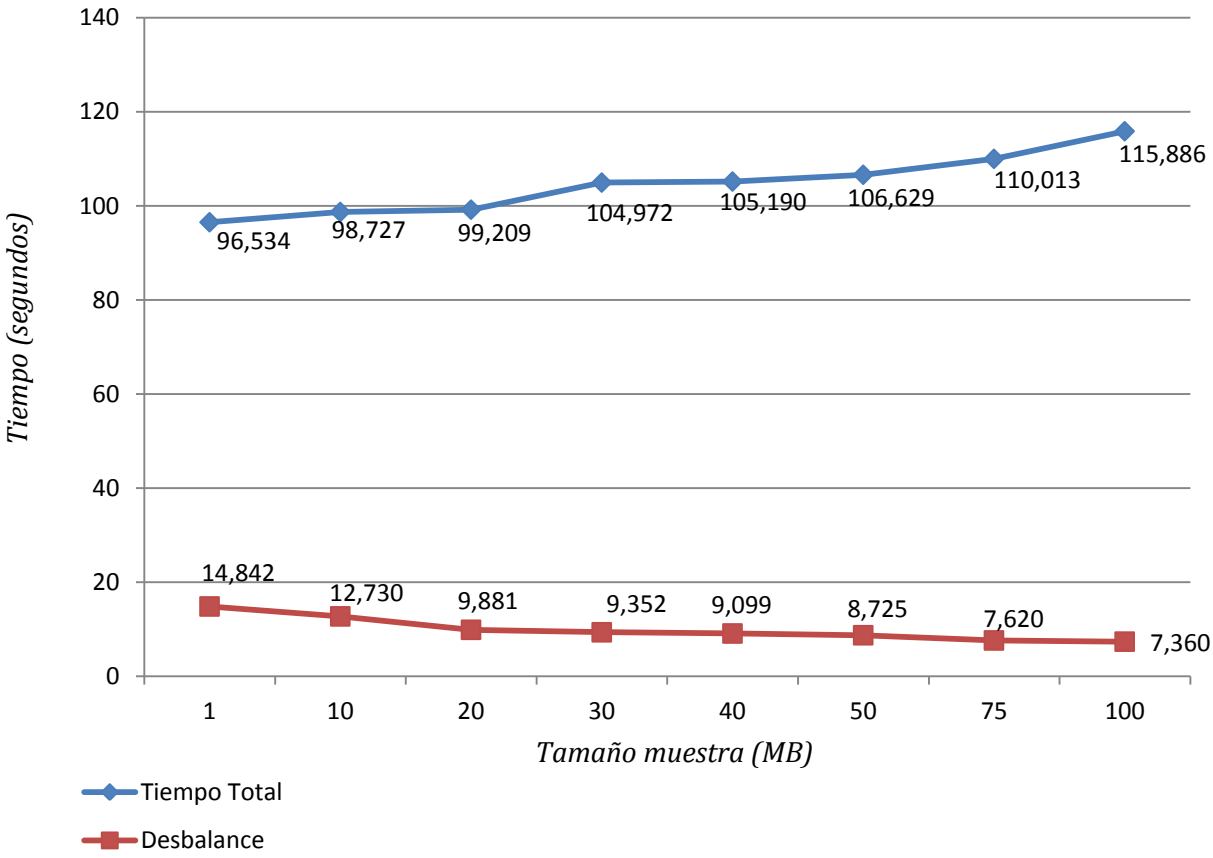

Figura 30: Gráfico "Tiempos" vs. "Tamaño muestra" con archivo de 1 GB

En conclusión para el gráfico de la figura 30, la mejor opción es la muestra de 20 MB, ya que se puede observar que comparada con la muestra de 10 MB, el desbalance disminuye en 2.849 segundos, siendo la mejor disminución de desbalance en el gráfico. Además, con respecto a la muestra de 30 MB, el aumento en el tiempo de transferencia es de 5.763 segundos, aunque el desbalance haya disminuido 0.529 segundos, esto no compensa el aumento en el tiempo de la descarga.

8.4.3 Muestra de 2 GB

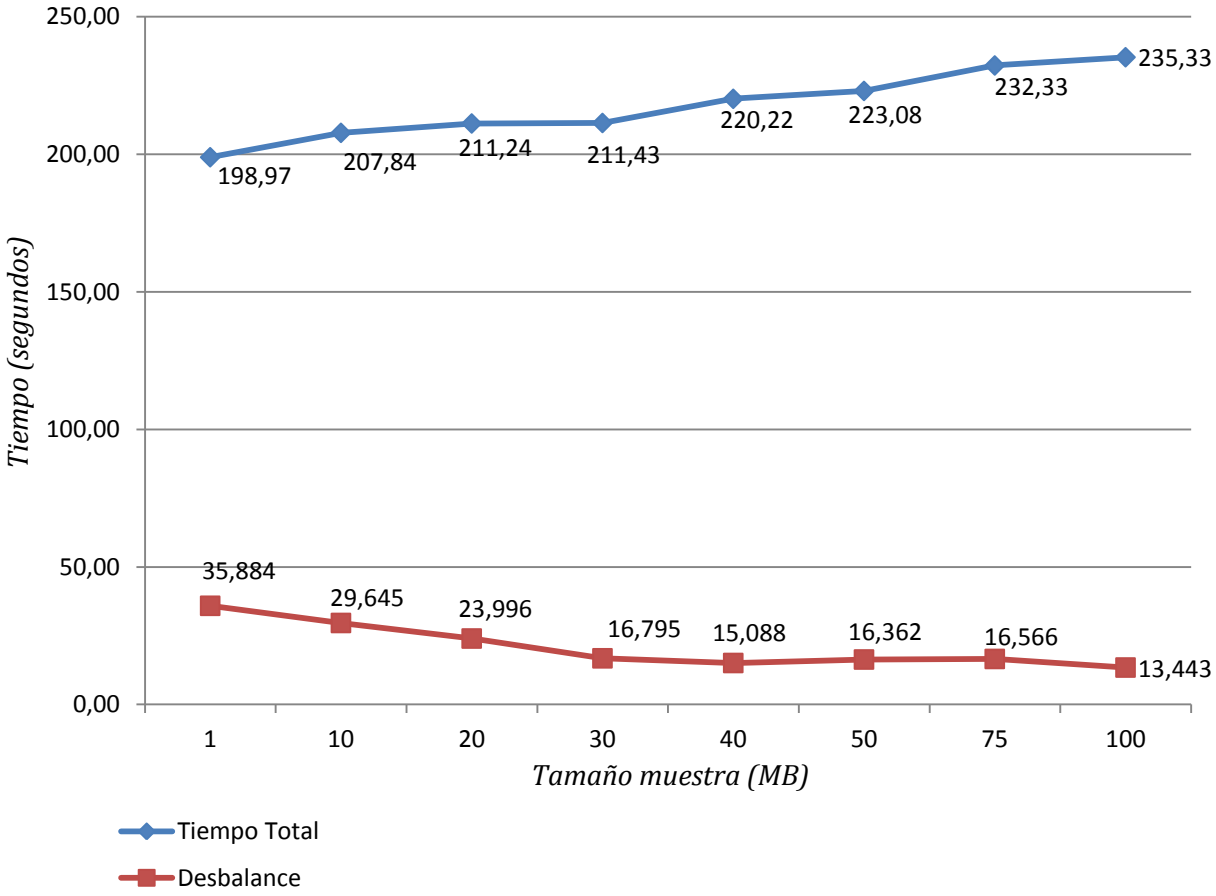

Figura 31: Gráfico "Tiempos" vs. "Tamaño muestra" con archivo de 2 GB

En el gráfico de la figura 31, se observa una disminución significativa del desbalance en la muestra de 30 MB, mientras que el tiempo total de descarga tiene un aumento mínimo con esta muestra, por lo que 30 MB sería la muestra óptima en este caso.

8.4.4 Conclusiones gráficos "Tiempos" vs. "Tamaño muestra"

Como el tamaño de las muestras varía dependiendo del tamaño del archivo, o sea, a mayor tamaño del archivo, un mayor tamaño de muestra es el óptimo para una mejor transferencia. Debido a lo anterior, se calculó qué porcentaje del archivo representa cada muestra. Para un archivo de 500 MB, la muestra óptima es de 10 MB, siendo esta muestra un 2 % del archivo. Para un archivo de 1 GB, la muestra óptima es de 20 MB, siendo esta muestra también un 2 % del archivo. Finalmente, para un archivo de 2 GB, la muestra óptima es de 30 MB, que corresponde a un 1.5 % del archivo.

Dado que los porcentajes óptimos rondan el 2 %, se utiliza este porcentaje del archivo como muestra para cualquier archivos de cualquier tamaño.

8.5 "Tiempos" vs. "Número de clientes" con muestra del 2 %

En esta sección se calcularon nuevamente el tiempo total de trasferencia y el tiempo de desbalance para distintos números de cliente (1 a 4 clientes), debido a que en la sección anterior se concluyó que la muestra óptima corresponde a un 2 % del tamaño del archivo. Se realiza la transferencia con archivos de 500 MB y 1 GB, lo que permite calcular cuál es el número de clientes óptimo para el tamaño de muestra óptimo.

En teoría, se espera que mientras más clientes realicen la trasferencia en paralelo, el tiempo total de transferencia debiera ir disminuyendo.

8.5.1 Número de clientes (500 MB)

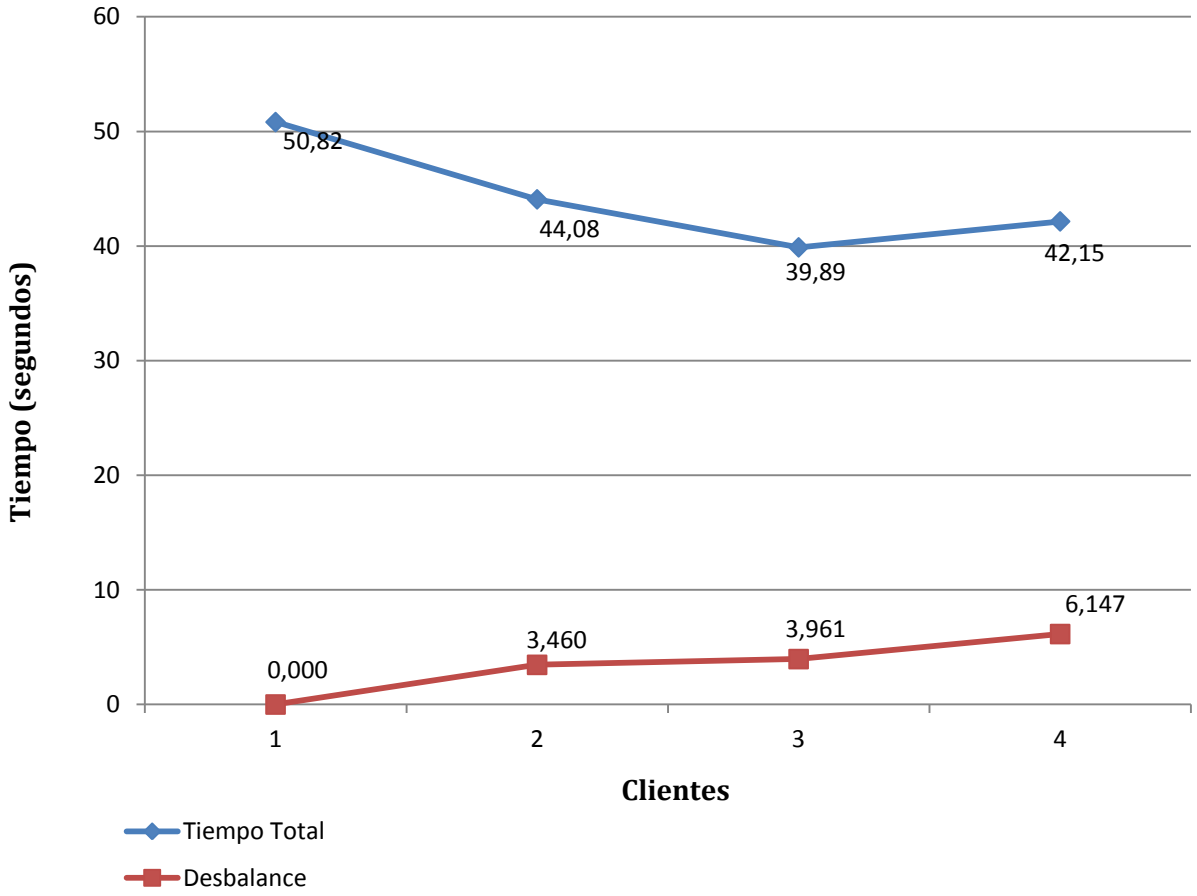

Figura 32: Gráfico "Tiempos" vs. "Número de clientes" con muestra del 2 %, archivo de 500 MB

En el gráfico de la figura 32 se observa que el tiempo total disminuye de manera progresiva mientras va aumentando el número de clientes. Sin embargo, esto sucede hasta solo la cantidad de 3 clientes, ya que con 4 clientes, el tiempo aumenta 2,265 segundos con respecto al tiempo de 3 clientes.

Por otro lado, de 2 a 3 clientes el desbalance tiene su aumento más bajo con 0.501 segundos. Además, este aumento de tiempo es compensado con la disminución de tiempo total.

8.5.2 Número de clientes (1 GB)

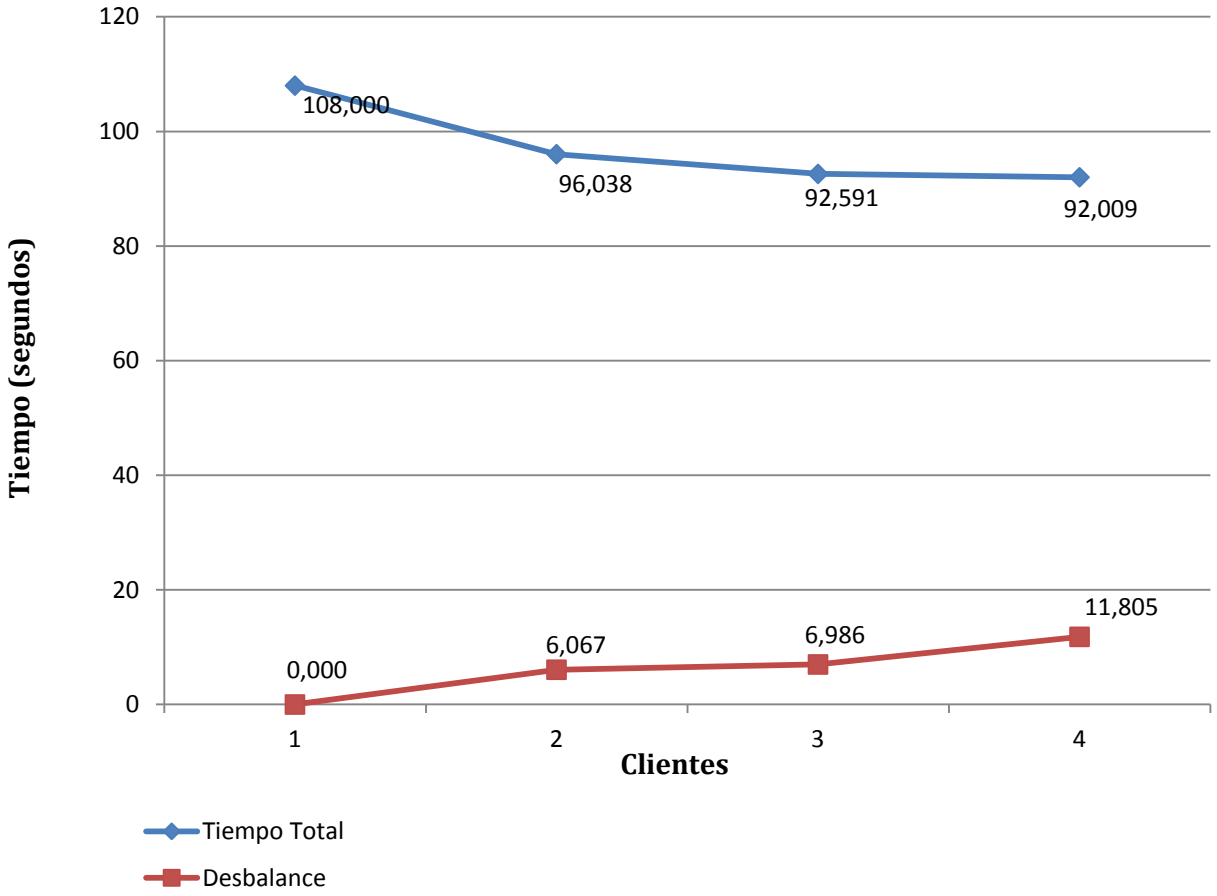

Figura 33: Gráfico "Tiempos" vs. "Número de clientes" con muestra del 2 %, archivo de 1 GB

En el gráfico de la figura 33 se observa que desde 1 a 2 clientes el tiempo total disminuye notablemente, pero el desbalance aumenta en 6.067, debido que con 1 cliente no hay un verdadero desbalance que considerar. En cambio, desde 2 a 3 cliente, el desbalance aumenta solo 0.92 segundos, aumento que es compensado con una disminución del tiempo total de 3.45 segundos.

Por otra parte, con 4 cliente, disminuye el tiempo total en 0.58 segundos, pero el aumento de desbalance es de 4.82 segundos, lo que no compensa la disminución del tiempo de total de transferencia.

8.5.3 Conclusiones gráficos "Tiempos" vs. "Número de clientes" con muestra del 2%

De los gráficos de las figuras 32 y 33 se puede concluir que la cantidad óptima de clientes para una mejor transferencia es de 3 clientes.

8.6 "Tiempos" v/s "Distancia"

Utilizando la cantidad óptima de 3 clientes, obtenida en la sección anterior, se realizan mediciones a distinta distancia para comprobar hasta qué distancia la aplicación funciona eficazmente. Las mediciones se realizaron a distancias de 20 cm, 100 cm, 200 cm, 300 m y 400 cm, transfiriendo un archivo de 500 MB.

8.6.1 Distancia (Archivo de 500 MB)

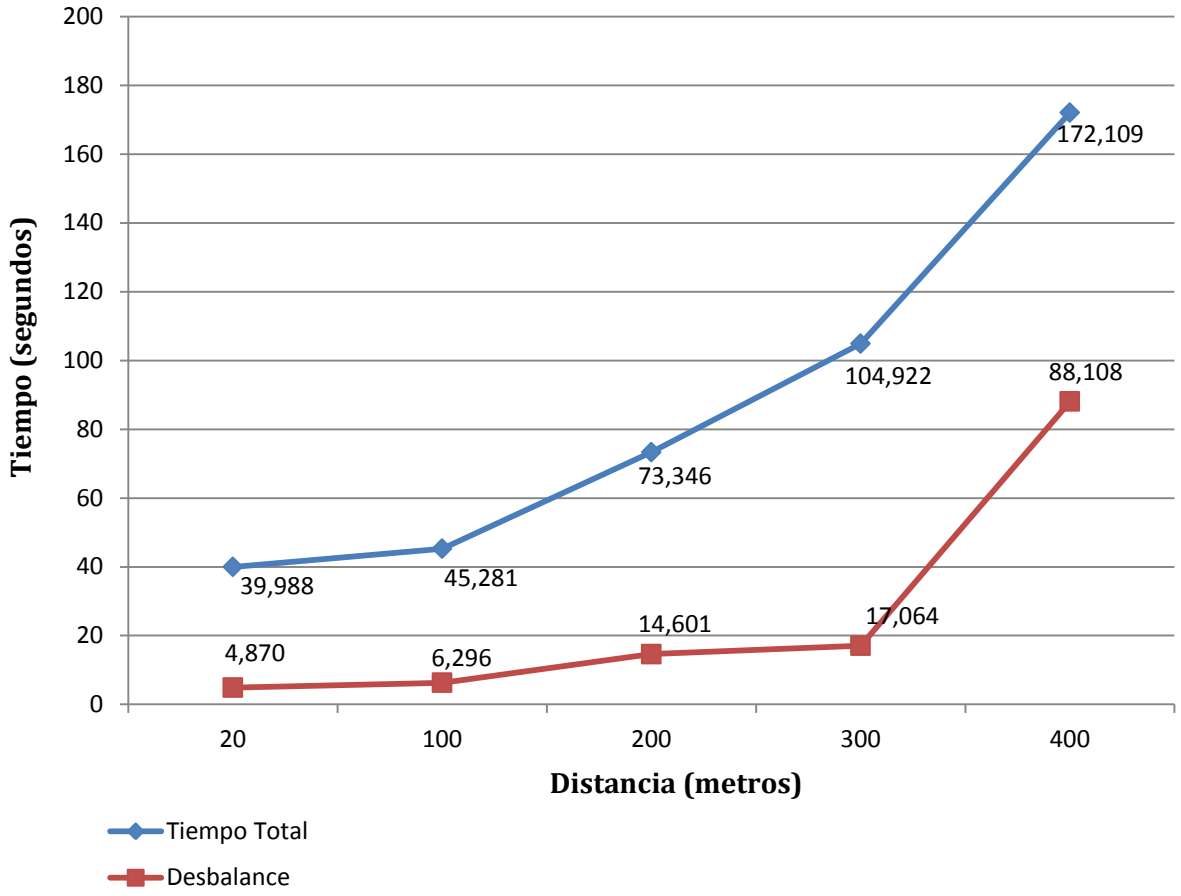

Figura 34: Gráfico "Tiempos" vs. "Distancia"

En el gráfico de la figura se puede observar que a mayor distancia, mayor será el tiempo total de trasferencia y desbalance. Sin embargo, se puede apreciar que luego de los 300 metros los tiempos aumentan en mayor proporción que en distancias menores, ya que como el GO (Group Owner) actúa como un punto de acceso, a más distancia más débil será la señal , y por ende, más lenta será la transferencia de datos.

9 CONCLUSIONES

Tantos los objetivos del proyecto como los objetivos del software fueron alcanzados con éxito al momento de finalizar el presente proyecto. La aplicación logró superar ampliamente los tiempos de transferencia de las otras tecnologías en cuanto a minimizar estos valores. La investigación y posterior utilización de la tecnología Wi-Fi Direct fue la base de la aplicación, lo que nos permitió poder conectar los dispositivos entre sí para luego incluir la descarga en paralelo. Se diseñó e implementó un algoritmo que permite medir la porción del archivo a transferir desde los dispositivos clientes, que encuentra un equilibrio entre el nivel de batería del dispositivo y la tasa de transferencia con el Group Owner. Se logró el objetivo general del software que consiste la correcta transferencia de archivos aplicando la técnica de descarga en paralelo Adaptativa P2P.

El uso de Android presenta un gran potencial, ya que posee mucha documentación con la cual pudimos crear una aplicación que sea compatible en muchos dispositivos, al igual que el uso del lenguaje de programación Java, el que nos ayudó a la implementación de la técnica de Descarga en Paralelo Adaptativa P2P, lo que se logró aplicando correctamente el uso de hilos para la descarga en paralelo. Debido a que no se contaba con los dispositivos necesarios desde el comienzo para el desarrollo correcto y eficiente de la aplicación, la planificación inicial del proyecto se extendió al doble de lo estipulado en la propuesta al inicio del proyecto. Luego de que la Universidad del Bío-Bío facilitara 5 smartphones Samsung S5 Neo, el desarrollo de la aplicación fue mucho más rápido, dado que las características de los celulares eran óptimas para la uso de la tecnología Wi-Fi Direct.

Finalmente, desde el punto de vista académico, podemos recalcar que realizar este proyecto de título nos ayudó a ahondar en lo que es programación móvil, tema que nos interesa trabajar a futuro, además de aprender sobre transferencia de datos entre celulares, el cual es un tema actual, ya que estos dispositivos se utilizan masivamente hoy en día. Lo más importante fue realizar una aplicación que disminuya los tiempos de transferencia de archivos, lo que beneficiará a muchos usuarios ya que las técnicas que se usan actualmente realizan las transferencias en un tiempo muy alto.

Desde lo personal, el proyecto nos entregó nuevas habilidades para enfrentar grandes retos en lo que concierne al trabajo laboral y esto nos motiva a crear proyectos en este mismo ámbito.

Cabe destacar que la aplicación podría desarrollarse aún más en varios aspectos, lo que podría utilizarse para proyectos o trabajos futuros.

Por ejemplo, al realizar la medición de la tasa de transferencia, actualmente esta se calcula mediante el envío de una muestra del archivo al iniciar la transferencia. Esto se realiza una sola vez durante todo el proceso. La técnica podría mejorarse realizando más mediciones durante el proceso, lo que proporcionaría un mejor cálculo de tasa de transferencia, pues estos cálculos se complementarían entre sí.

Por otra parte, para obtener el archivo final la aplicación necesita leer nuevamente cada parte enviada por los clientes para luego crear este archivo final. Para mejorar esto, las partes transferidas deberían ir generando el archivo final a medida que se transfieren, lo que lograría mejores tiempos de transferencia al no tener que leer cada parte.

Finalmente, para aprovechar la muestra calculada para la tasa de transferencia, esta podría añadirse a la generación del archivo final.

10 BIBLIOGRAFÍA

[1] "*Wi-Fi Direct*". **Fuente**: https://en.wikipedia.org/wiki/Wi-Fi_Direct

[2] "*Wi-Fi Alliance R. Discover Wi-Fi Direct Services"*. **Fuente**: http://www.wi-fi.org

[3] "*Herramientas para empezar a desarrollar con Android"*.
Fuente:http://soloelectronicos.com/2012/02/02/herramientas-para-empezar-a-desarrollar-con-android

[4] "*Archivo (informática)".*
Fuente*:* https://es.wikipedia.org/wiki/Archivo_%28inform%C3%A1tica%29

[5] "*Checksum"*. **Fuente**: http://es.gdict.org/definicion.php?palabra=checksum

[6] "*Tableta (computadora).*
Fuente: https://es.wikipedia.org/wiki/Tableta_%28computadora%29

[7] "*Wifi"*. **Fuente**: https://es.wikipedia.org/wiki/Wifi

[8] "*Punto de acceso inalámbrico"*.
Fuente: https://es.wikipedia.org/wiki/Punto_de_acceso_inal%C3%A1mbrico

[9] "*Dynamic Host Configuration Protocol"*.
Fuente: https://es.wikipedia.org/wiki/Dynamic_Host_Configuration_Protocol

[10] "*File Transfer Protocol"*.
Fuente: https://es.wikipedia.org/wiki/File_Transfer_Protocol

[11] "*El protocolo HTTP"*.
Fuente: http://neo.lcc.uma.es/evirtual/cdd/tutorial/aplicacion/http.html

[12] "*Rapid Application Development (RAD), Entorno integrado de desarrollo (IDE) Ingeniería de Software Asistida por computador (CASE)"*.
Fuente: https://ingsoftwarei2014.wordpress.com/category/rapid-applicationdevelopment-rad-entorno-integrado-de-desarrollo-ide-ingenieria-de-software-asistida-por-computador-case/

[13] "*Institute of Electrical and Electronics Engineers".*
Fuente: https://es.wikipedia.org/wiki/Institute_of_Electrical_and_Electronics_Engineers

[14] "*Near Field Communication"*.
Fuente: https://es.wikipedia.org/wiki/Near_field_communication

[15] "*Sistema".* **Fuente:** http://es.thefreedictionary.com/sistema

[16] "*Modelo OSI"*. **Fuente**: https://es.wikipedia.org/wiki/Modelo_OSI

[17] "*Peer-to-peer*". **Fuente**: https://es.wikipedia.org/wiki/Peer-to-peer

[18] *"RFID".* **Fuente:** https://es.wikipedia.org/wiki/RFID

[19] "*Special Interest Group*".
Fuente: https://en.wikipedia.org/wiki/Special_Interest_Group

[20] "*Transmission Control Protocol"*.
Fuente: https://es.wikipedia.org/wiki/Transmission_Control_Protocol

[21] "*Definición de WLAN".* **Fuente:** http://definicion.de/wlan/

[22] "*Wi-Fi Protected Setup"*.
Fuente: https://es.wikipedia.org/wiki/Wi-Fi_Protected_Setup

[23] R. Karrer, E. Knightly, "*TCP-PARIS: a Parallel Download Protocol for Replicas".*
Fuente: http://networks.rice.edu/files/2014/08/tcp-paris.pdf

[24] C. Barrientos," *Descarga en paralelo de archivos sensible a variaciones del ancho de banda"*. 2014.

[25] J. Byers, M. Luby, and M. Mitenmacher, "*Accessing Multiple Mirror Sites in Parallel: Using Tornado Codes to Speed Up Downloads*". Abril 1999.

[26] Rodriguez P. y Biersack W, "*Dynamic Parallel Access to Replicated Content in the Internet*". Agosto 2002.

[27] S. Sohail, S.K. Jha, S. Kanhere, C-T. Chou, "*QoS Driven Parallelization of Resources to Reduce File Download Delay, "IEEE Transactions on Parallel and Distributed Systems, vol.17.* Octubre 2006.

[28] Y. Nebat, M. Sidi, "*Parallel downloads for streaming applications—a resequencing analysis*". 2004.
Fuente: http://sidi.eew.technion.ac.il/files/pubs/m_sidi_resequencing_06.pdf

[29] "*What is an smartphone*".
Fuente: https://www.techopedia.com/definition/2977/smartphone

[30] "*Android (operating system)*".
Fuente: https://en.wikipedia.org/wiki/Android_%28operating_system%29

[31] *"Arquitectura de Android".*
Fuente: http://androideity.com/2011/07/04/arquitectura-de-android/

[32] Xamarin, "*Understanding Android API Levels*".
Fuente:https://developer.xamarin.com/guides/android/application_fundamentals/understanding_android_api_levels

[33] Wi-Fi Alliance, "*Who are We*". **Fuente**: http://www.wi-fi.org/who-are-we

[34] *"Bluetooth vs. Wi-Fi".* **Fuente:** http://www.diffen.com/difference/Bluetooth_vs_Wifi

[35] P. Thornycroft, "*Designed for Speed: Network Infrastructure in a 802.11n World"*. 2013.

[36] B.G. Lee, S. Cho, "*Broadband Wireless Access and Local Networks: Mobile WiMax and WiFi"*. 2008.

[37] B. Sikdar, "*Environmental Impact of IEEE 802.11 Access Points: A Case Study"*. 2010.

[38] M. Frodigh, P. Johansson, P. Larsson, "*Wireless Ad hoc networking - The art of networking without a network"*. 2000.

[39] L.M. Feeny, M. Nilsson, "*Investigating the Energy Consumption of a Wireless Network Interface in an Ad hoc Networking Environment"*. 2001.

[40] "*A Look at the Basics of Bluetooth Technology"*.
Fuente: https://www.bluetooth.org/pages/basics.aspx

[41] S.A. Ahson, M. Ilyas, "*Near Field Communication Handbook*". 2011.

[42] NFC Forum, "*Our Mission & Goals*".
Fuente: http://nfc-forum.org/about-us/the-nfc-forum

[43] Wi-Fi Alliance, "*FAQ*".
Fuente: http://www.wi-fi.org/knowledge-center/faq/does-wi-fi-direct-work-on-80211-abgn

[44] Hughes Systique Corporation, "*Wi-Fi Direct™*".
Fuente:hsc.com/Portals/0/Uploads/Articles/WFD_Technology_Whitepaper_v_1.7635035318321315728.pdf

[45] Android Developers, "*Wi-Fi Peer-to-Peer*".

Fuente: http://developer.android.com/intl/es/guide/topics/connectivity/wifip2p.html

[46] B.A. Forouzan, "*Data Communications and Networking"*. 2003.

[47] Camps-Mur, D. (2013). NEC Network Laboratories. Device-To-Device Communications.

[48] Wi-Fi Alliance, "*Technical Committee. P2P Task Group. Wi-Fi Peer-to-Peer (P2P) Technical Specification Version 1.2"*. 2011.

[49] *"Handshaking"*. **Fuente**: https://es.wikipedia.org/wiki/Handshaking

[50] "BPMN, Business Process Modeling Notation".
Fuente: http://www.bizagi.com/esp/descargas/BPMNbyExample.pdf

11 ANEXO: DISEÑO

11.1 Árbol de descomposición funcional

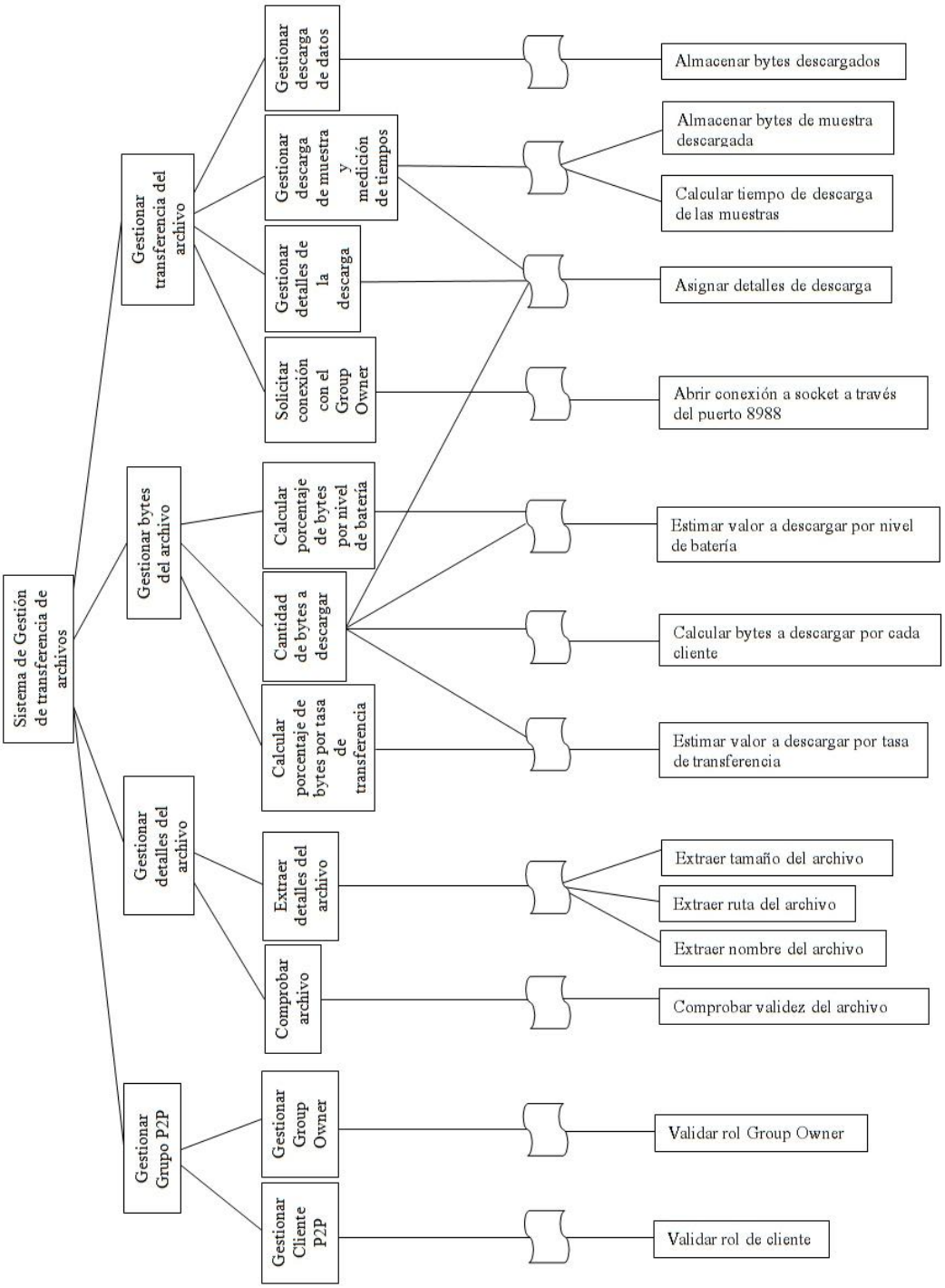

Figura 35: Árbol de descomposición funcional

11.2 Diseño de interfaz y navegación

11.2.1 Diseño de Interfaz Gráfica de Usuario

11.2.1.1 Diseño de Interfaz Gráfica de Usuario General

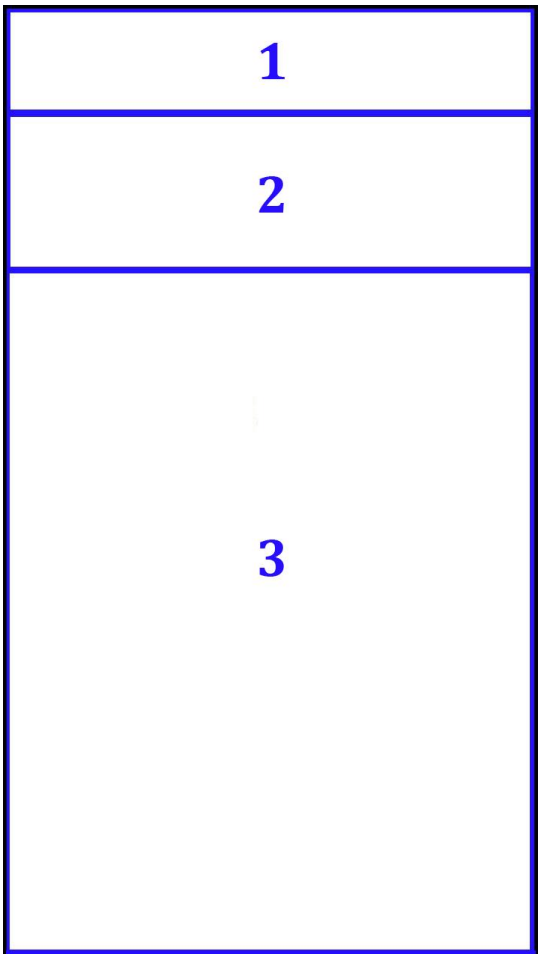

Figura 36: Diseño interfaz gráfica de usuario general

Área 1: Encabezado de la aplicación. Incluye: logo y nombre de la aplicación; ícono de búsqueda de dispositivos e ícono para cerrar la aplicación.

Área 2: Incluye el nombre del dispositivo que ejecuta la aplicación y su estado.

Área 3: Despliegue de los dispositivos pares encontrados y su correspondiente estado.

11.2.1.2 Diseño de interfaz gráfica de usuario cliente

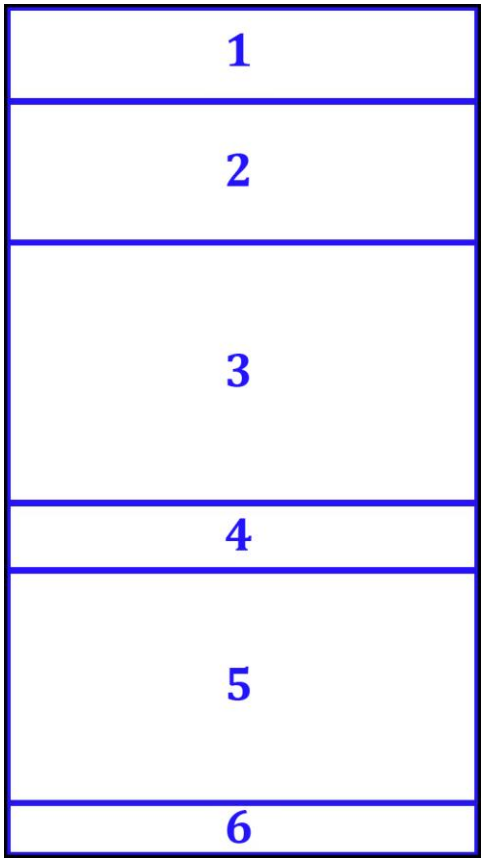

Figura 37: Diseño interfaz gráfica de usuario cliente

Área 1: Encabezado de la aplicación. Incluye: logo y nombre de la aplicación; ícono de búsqueda de dispositivos e ícono para cerrar la aplicación.

Área 2: Incluye el nombre del dispositivo que ejecuta la aplicación y su estado.

Área 3: Despliegue de los dispositivos pares encontrados y su correspondiente estado.

Área 4: Botones de opción ('Desconectar' y 'Enviar Archivo').

Área 5: Despliegue de datos del dispositivo que ejecuta la aplicación.

Área 6: Sección de mensajes de notificación.

11.2.1.3 Diseño de Interfaz Gráfico de Group Owner

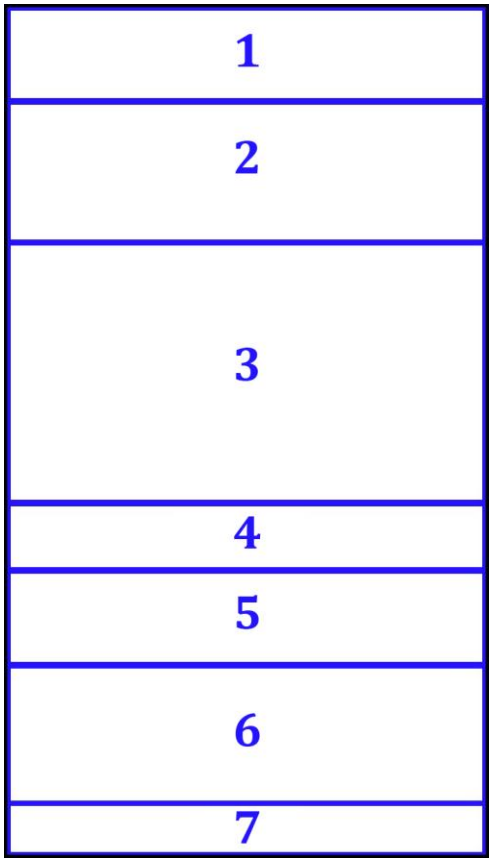

Figura 38: Diseño interfaz gráfica de usuario Group Owner

Área 1: Encabezado de la aplicación. Incluye: logo y nombre de la aplicación; ícono de búsqueda de dispositivos e ícono para cerrar la aplicación.

Área 2: Incluye el nombre del dispositivo que ejecuta la aplicación y su estado.

Área 3: Despliegue de los dispositivos pares encontrados y su correspondiente estado.

Área 4: Botones de opción ('Desconectar').

Área 5: Despliegue de datos del dispositivo que ejecuta la aplicación.

Área 6: Sección de selección de cantidad de clientes.

Área 7: Sección de mensajes de notificación.

11.3 Jerarquía del Menú

Estos diagramas muestran los niveles de profundidad que tienen las opciones del menú de cada tipo de usuario.

11.3.1 Jerarquía del Menú del Group Owner

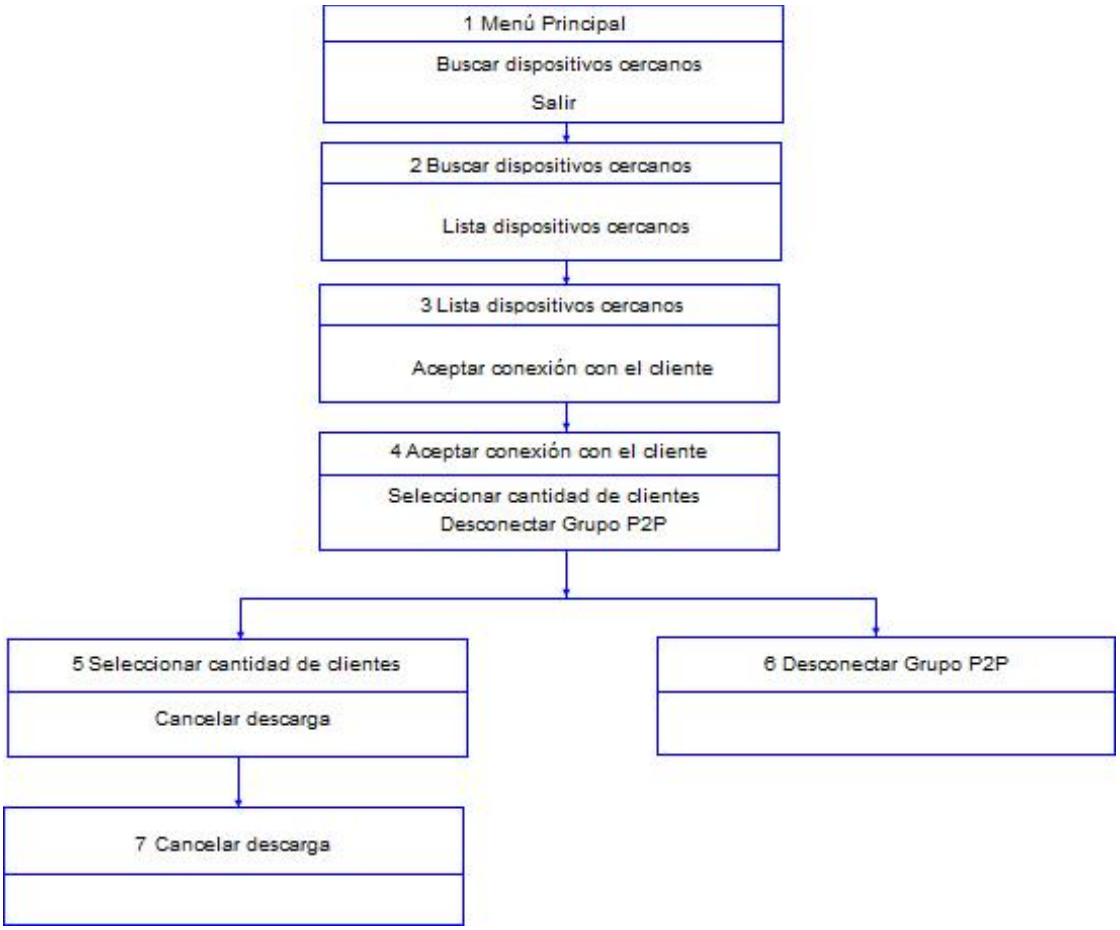

Figura 39: Jerarquía del menú del Group Owner

11.3.2 Jerarquía del menú del cliente

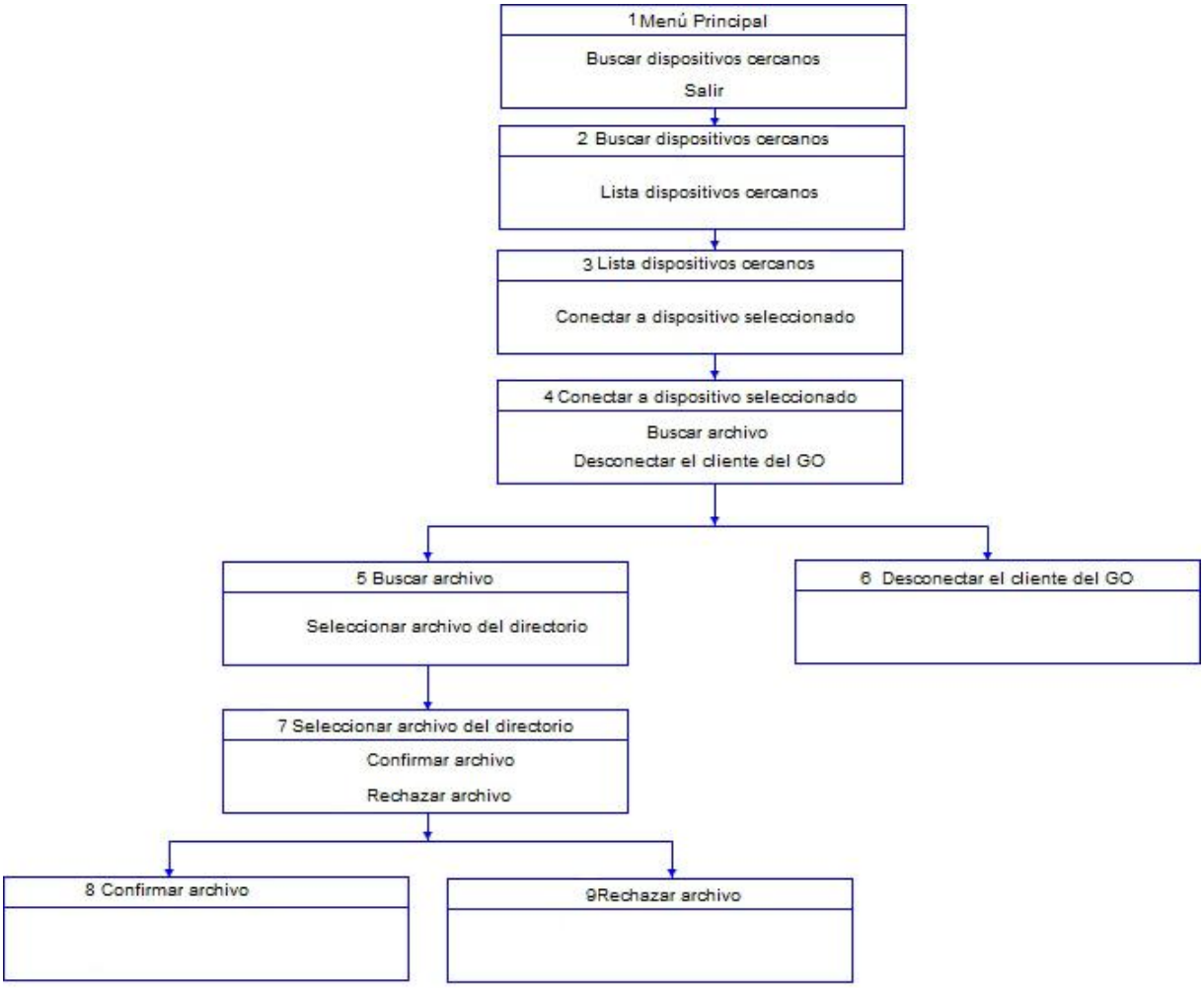

Figura 40: Jerarquía del menú del cliente

11.4 Navegación del Menú

11.4.1 Navegación del menú del cliente

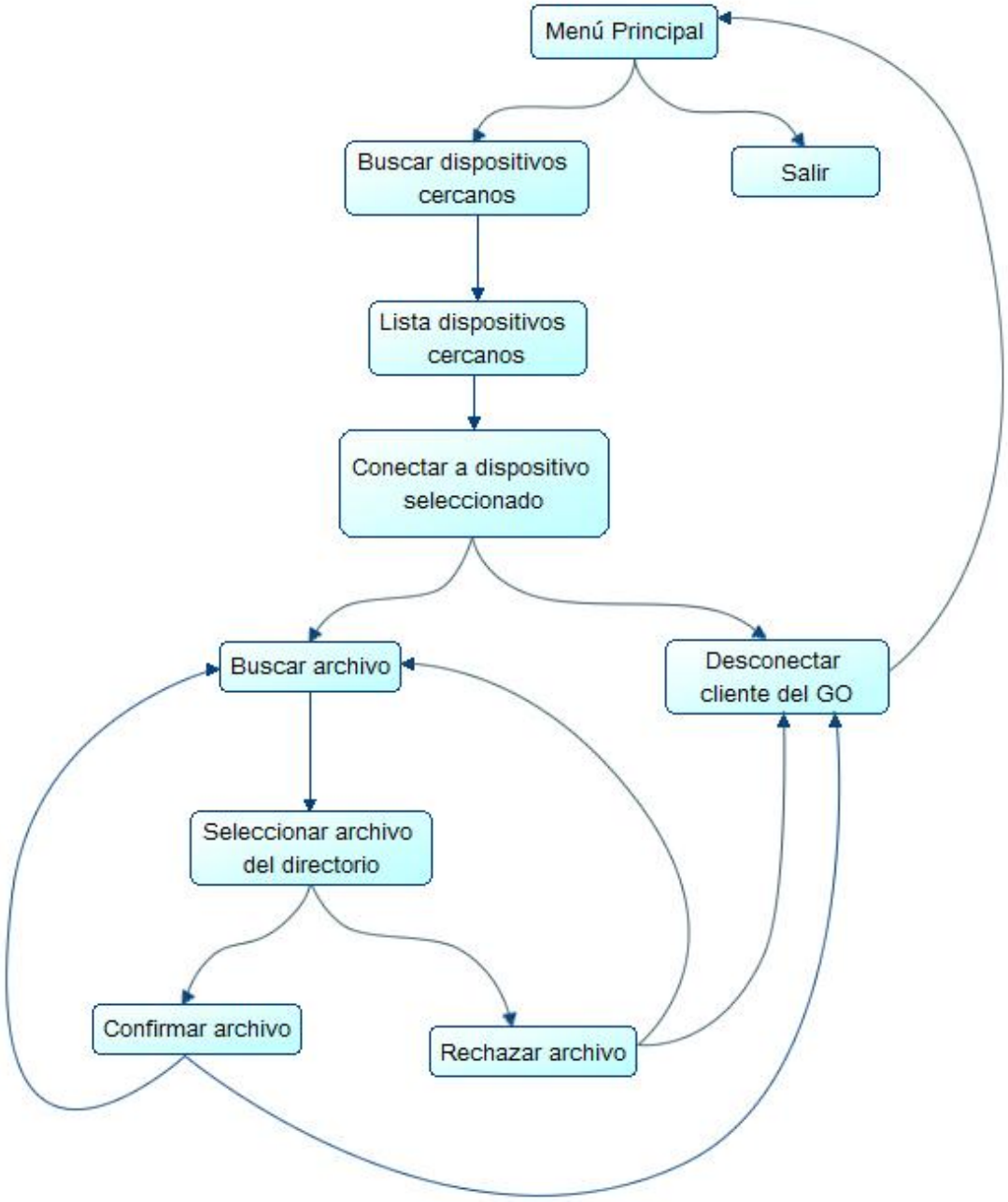

Figura 41: Navegación del menú del cliente

11.4.2 Navegación del menú del Group Owner

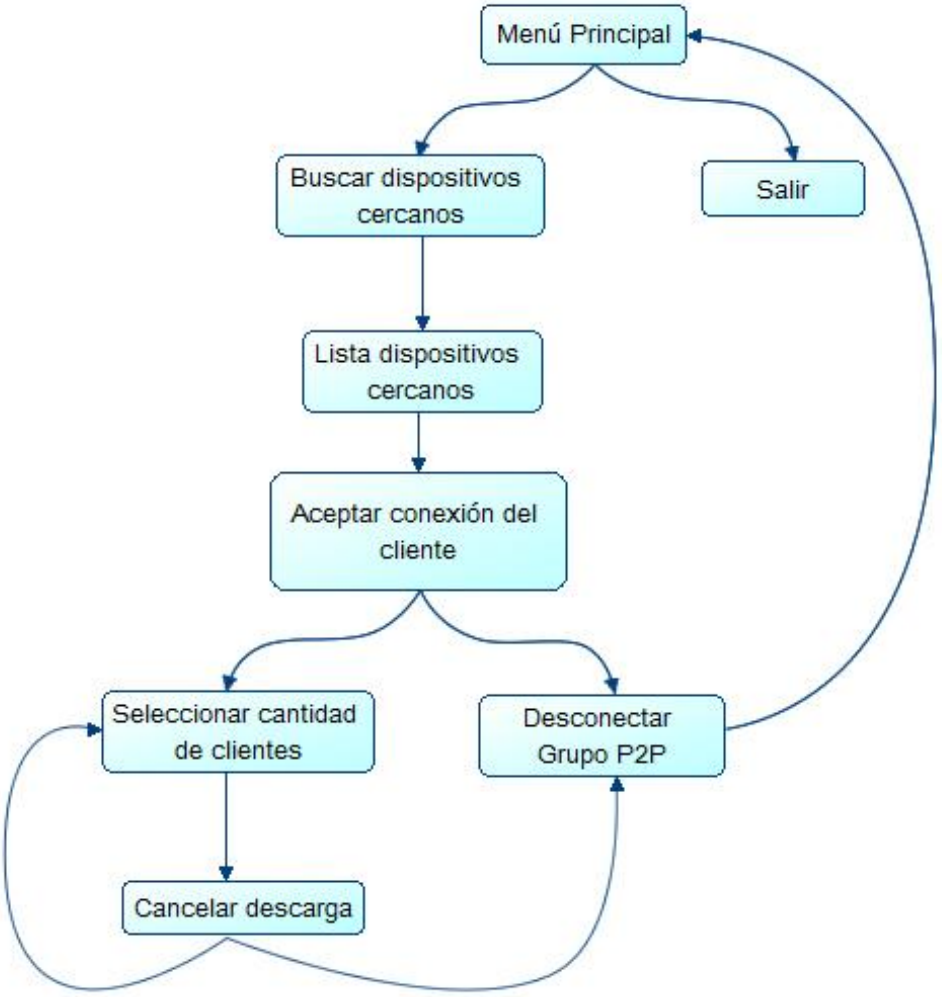

Figura 42: Navegación del menú del Group Owner

11.5 Especificación de módulos

N° módulo: 1		Nombre módulo: Validar rol de cliente	
Parámetros de entrada		Parámetros de salida	
Nombre: Conexión Cliente	Tipo de dato: String	Nombre: Resultado de validación	Tipo de dato: int

Tabla 10: Módulo "Validar rol de cliente"

N° módulo: 2		Nombre módulo: Validar rol del Group Owner	
Parámetros de entrada		Parámetros de salida	
Nombre: Conexión Group Owner	Tipo de dato: String	Nombre: Resultado de validación	Tipo de dato: int

Tabla 11: Módulo "Validar rol del Group Owner"

N° módulo: 3		Nombre módulo: Extraer nombre del archivo	
Parámetros de entrada		Parámetros de salida	
Nombre: Archivo seleccionado del directorio	Tipo de dato: File	Nombre: Nombre del archivo	Tipo de dato: String

Tabla 12: Módulo "Extraer nombre del archivo"

N° módulo: 4		Nombre módulo: Extraer tamaño del archivo	
Parámetros de entrada		Parámetros de salida	
Nombre: Archivo seleccionado del directorio	Tipo de dato: File	Nombre: Tamaño del archivo	Tipo de dato: Long

Tabla 13: Módulo "Extraer tamaño del archivo"

N° módulo: 5		Nombre módulo: Extraer ruta del archivo	
Parámetros de entrada		Parámetros de salida	
Nombre: Archivo seleccionado del directorio	Tipo de dato: File	Nombre: Ruta del archivo	Tipo de dato: String

Tabla 14: Módulo "Extraer ruta del archivo"

N° módulo: 6		Nombre módulo: Estimar valor a descargar por tasa de transferencia	
Parámetros de entrada		Parámetros de salida	
Nombre: Tiempos de transferencia de la muestra.	Tipo de dato: Long	Nombre: Porcentaje de bytes a descargar por transferencia de descarga	Tipo de dato: float

Tabla 15: Módulo "Estimar valor a descargar por tasa de transferencia"

N° módulo: 7		Nombre módulo: Estimar valor a descargar por nivel de batería	
Parámetros de entrada		Parámetros de salida	
Nombre: Nivel de Batería del dispositivo cliente	Tipo de dato: int	Nombre: Porcentaje de bytes a descargar por nivel de batería	Tipo de dato: float

Tabla 16: Módulo "Estimar valor a descargar por nivel de batería"

N° módulo: 8		Nombre módulo: Calcular bytes a descargar por cada cliente	
Parámetros de entrada		Parámetros de salida	
Nombre: Porcentaje de bytes a descargar por nivel de batería Porcentaje de bytes a descargar por nivel de batería	Tipo de dato: float float	Nombre: Bytes a transferir por cada cliente	Tipo de dato: long

Tabla 17: Módulo " Calcular bytes a descargar por cada cliente "

N° módulo: 9		Nombre módulo: Abrir conexión a socket a través del puerto 8988	
Parámetros de entrada		Parámetros de salida	
Nombre: puerto al cual conectarse	Tipo de dato: int	Nombre: socket para transferencia de datos	Tipo de dato: socket

Tabla 18: Módulo "Abrir conexión a socket a través del puerto 8988"

N° modulo: 10		Nombre modulo: Asignar detalles de la descarga	
Parámetros de entrada		Parámetros de salida	
Nombre: Batería del dispositivo cliente, nombre del archivo, tamaño del archivo, ruta del archivo.	Tipo de dato: int, string, long, string.	Nombre: Detalle requerido en distintos procesos (ruta archivo, nivel batería)	Tipo de dato: String, int

Tabla 19: Módulo "Asignar detalles de la descarga"

N° módulo: 11		Nombre módulo: Calcular tiempos de descarga de las muestras	
Parámetros de entrada		Parámetros de salida	
Nombre: Tiempo del SO del dispositivo móvil Group Owner.	Tipo de dato: long	Nombre: Tiempo total de descarga desde el cliente	Tipo de dato: long

Tabla 20: Módulo "Calcular tiempos de descarga de las muestras"

N° módulo: 12		Nombre módulo: Almacenar bytes de muestras descargadas	
Parámetros de entrada		Parámetros de salida	
Nombre: Buffer de 8192 bytes	Tipo de dato: array de bytes	Nombre: archivo de muestra de 1mb	Tipo de dato: bytes

Tabla 21: Módulo "Almacenar bytes de muestras descargadas"

N° módulo: 13		Nombre módulo: Almacenar bytes descargados	
Parámetros de entrada		Parámetros de salida	
Nombre: Buffer de 8192 bytes	Tipo de dato: array de bytes	Nombre: archivo final transferido	Tipo de dato: bytes

Tabla 22: Módulo "Almacenar bytes de muestras descargadas"

12 ANEXO: CÓDIGO FUENTE

En esta sección se presenta la codificación de un segmento significativo de la aplicación. La clase "DeviceDetailFragment", llamada así ya que es un fragmento Android (porción de la interfaz de usuario que puede añadirse o eliminarse de la interfaz de forma independiente al resto de elementos de la aplicación), muestra los componentes que puede ejecutar cada dispositivo, permite la configuración entre el cliente y GO (Group Owner), permite la selección del archivo a ejecutar y lo más importante, permite la gestión de la transferencia de archivos en paralelo. Esta clase es ejecutada mayormente por la aplicación del lado del GO, donde se encuentran los hilos que se ejecutan por cada cliente que transfiera su archivo, las mediciones de los tiempos del envío de la muestra, la asignación de los bytes a descargar por cada cliente y, finalmente, la transferencia de las partes del archivo.

La otra clase importante es la clase "FileTransferService", que es la que permite crear sockets del lado del cliente que piden conexión al GO, y también se encarga de transferir los detalles del dispositivo, desde el lado cliente al GO.

Otras clases presentes en la codificación están relacionadas con la conexión entre dispositivos basados en la tecnología Wi-Fi Direct, como lo son las clases "WifiDirectActivity", "WifiDirectBroadcastReceiver" y "DeviceListFragment", que contienen la búsqueda de dispositivos cercanos, la petición de conexión al dispositivo seleccionado y detalles propios de Wi-Fi Direct.

A continuación se muestran las partes más importantes de la clase "DeviceDetailFragment", ya que la codificación es extensa.

- ✓ Método "onClick": Permite la configuración y establecimiento del dispositivo como usuario cliente y que pide conexión al dispositivo GO.

```
public void onClick(View v) {

      WifiP2pConfig config = new WifiP2pConfig();
      config.deviceAddress = device.deviceAddress;
      config.wps.setup = WpsInfo.PBC;
      config.groupOwnerIntent = 0;
```

```
      if (progressDialog != null && progressDialog.isShowing())
         progressDialog.dismiss();

progressDialog = ProgressDialog.show(getActivity(), "Presione ATRÁS
para cancelar","Uniéndose a '" + device.deviceName + "'", true,  true,new
DialogInterface.OnCancelListener() {

          @Override
          public void onCancel(DialogInterface dialog) {
          ((DeviceActionListener) getActivity()).cancelDisconnect();
          }
      } );
      ((DeviceActionListener) getActivity()).connect(config);
}
```

- ✓ Método "onActivityResult": Permite traspasar los datos del archivo seleccionado a la clase "FileTransferService" para el envío de estos al Group Owner.

```
@Override
public void onActivityResult(int requestCode, int resultCode, Intent data)
{
  try {
      String result = data.getStringExtra("fileB");
      Uri uri = Uri.parse(result);
      File f = new File(uri.getPath());
      String filePath = f.getPath();
      String fileName = f.getName();
      Long fileLength = f.length();

TextView statusText = (TextView)
mContentView.findViewById(R.id.texto_estado);
      statusText.setText("Enviando: " + uri);
Intent serviceIntent = new Intent(getActivity(),
FileTransferService.class);
serviceIntent.setAction(FileTransferService.ACTION_SEND_FILE);
serviceIntent.putExtra(FileTransferService.EXTRAS_FILE_PATH,uri.toString())
;
serviceIntent.putExtra(FileTransferService.EXTRAS_GROUP_OWNER_ADDRESS,info.
groupOwnerAddress.getHostAddress());
serviceIntent.putExtra(FileTransferService.EXTRAS_GROUP_OWNER_PORT, 8988);
      serviceIntent.putExtra(FileTransferService.NOMBRE_FILE, fileName);
serviceIntent.putExtra(FileTransferService.TAMAÑO_FILE, fileLength + "");
serviceIntent.putExtra(FileTransferService.NIVEL_BATERIA,
WiFiDirectBroadcastReceiver.nivelbateria);

      mProgressDialog[0]=null;
      showprogress("Enviando "+fileName);
      getActivity().startService(serviceIntent);
      } catch (Exception e) {}
}
```

- ✓ Creación de una subclase "FileServerAsyncTask" para la administración de los hilos y el manejo de los sockets para la gestión de las muestras y transferencia de archivos. Esta subclase es creada solo por el Group Owner.

```
if (info.groupFormed && info.isGroupOwner)
      new FileServerAsyncTask(getActivity(),mContentView.findViewById
      (R.id.texto_estado)).execute();
else
      if (info.groupFormed) {
      mContentView.findViewById(R.id.boton_ini_arch).setVisibility
      (View.VISIBLE);
((TextView)mContentView.findViewById(R.id.texto_estado)).setText(getResourc
es().getString(R.string.texto_client));
        }
public static class FileServerAsyncTask extends AsyncTask<Void, String,
Void>
{
 public Context context;
 public TextView statusText;

 public FileServerAsyncTask(Context context, View statusText) {
   this.context = context;
   this.statusText = (TextView) statusText;
}

@Override
 protected Void doInBackground(Void... params) {

  runningThread = true;

  class ClientThread implements Runnable {
   private Socket clientSocket;
   private int thisDevice;
   private String clientName;

   ClientThread(Socket client, int thisDevice) {
    this.clientSocket = client;
    this.thisDevice = thisDevice;
    this.clientName = "UNKNOWN";
   }

   public void run(){
    try{

     long startTime,endTime;
     spinner.setClickable(false);
     OutputStream stream = clientSocket.getOutputStream();
     ObjectOutputStream oos = new ObjectOutputStream(stream);
     InputStream inputstream = clientSocket.getInputStream();
     ObjectInputStream ois= newObjectInputStream(inputstream);
     ServerData serverData = null;

     if(serverData == null)
      serverData = new ServerData();
      serverData = new ServerData(thisDevice, totalDevices);
```

```
      oos.writeObject(serverData);
      ClientData clientData = null;

      try {
       clientData = (ClientData) ois.readObject();
      } catch (Exception e) {}
      clientName = clientData.getClientName();
      clientsNamesArray.add(thisDevice, clientName);

      //-----------------CREACIÓN ARCHIVO DE PRUEBA--------
final File preF = new     File(Environment.getExternalStorageDirectory()+
"/UBBifiDirect/"
          + clientData.getFileName()+"_0"+(thisDevice));
      File preDirs = new File(preF.getParent());
      if (!preDirs.exists()){
           preDirs.mkdirs();
      }
      if(preF.exists())
       preF.delete();
      preF.createNewFile();
      startTime = System.currentTimeMillis();
      preCopyFileServer(inputstream, new FileOutputStream(preF));
      endTime = System.currentTimeMillis();

      //-------------------FIN ARCHIVO DE PRUEBA---------
      while(preCount<totalDevices){}

      if(thisDevice==1){
       finalFileName = clientData.getFileName();
       finalFileLength = clientData.getFileLength();
      }
      batteriesArray.add(thisDevice-1, clientData.getBatteryLevel());
      long testTime = endTime-startTime;

      testTimesArray.add(thisDevice,testTime);
algorithmPart1(thisDevice,batteriesArray.get(thisDevice-1),    testTime);
      boolean createAlgorithm = false;

      do{

        if(algorithmPartFinalReady){
          AlgorithmData algorithmData = null;
          if(algorithmData == null)
            algorithmData = new AlgorithmData();
algorithmData = new AlgorithmData(algorithmPartFinalArray,
totalDevices);
  try {
           oos.writeObject(algorithmData);
          } catch (Exception e) {}
          createAlgorithm = true;
        }
      }while(!createAlgorithm);

final File f = new File(Environment.getExternalStorageDirectory()+
"/UBBifiDirect/"+"Part0"+(thisDevice)+"_"+clientData.getFileName();

      File dirs = new File(f.getParent());
      if (!dirs.exists())
```

```
        dirs.mkdirs();


       if(f.exists()) f.delete();//SI EL ARCHIVO EXISTE SE ELIMINA
       f.createNewFile();
       copyFileServer(inputstream, new FileOutputStream(f),thisDevice);
      }
      catch (Exception e){}
     }
    }
    try {

     ServerSocket serverSocket = new ServerSocket(8988);
     ClientThread c;
     boolean algP2Ready = false;

      try {

       while(runningThread){

        Socket s = serverSocket.accept();
        c = new ClientThread(s, currentDevice);
        currentDevice++;
        new Thread(c).start();
        if(currentDevice == totalDevices+1)
        {
         do{
           if(algorithmPart1Ready == totalDevices){
algorithmPart2Batteries = algorithmPart2(batteriesArray,
"algorithmPart2Batteries");
algorithmPart2Times = algorithmPart2(algorithmPart1Times,
"algorithmPart2Times");
             algorithmPartFinal();
             while(join != totalDevices){}
             if(algorithmPartFinalReady) algP2Ready = true;
           }
        }while(!algP2Ready);
        algP2Ready = false;
       }
       if(join == totalDevices){
        joinFile();
        resetVariables();
       }
      }
     } catch (Exception e) {
     serverSocket.close();
    } catch (IOException e) {}
   return null;
}
```

- ✓ Método "algorithmPart2": Permite calcular el porcentaje de bytes a descargar de acuerdo al nivel de batería de cada dispositivo cliente y a la tasa de transferencia, calculada con los tiempos de descarga de la muestra.

```
public static ArrayList<Float> algorithmPart2(ArrayList<Float> array,
String string)
{

  float xCount = 0;
  for (int i=0 ; i<totalDevices ; i++){
    xCount += array.get(i);
  }
  float y = finalFileLength/xCount;
  int percCount = 0;
  int [] perc = new int[totalDevices];

  for (int i=0 ; i<totalDevices ; i++){
    perc[i] = (int) ((y*array.get(i)*100)/finalFileLength);
    percCount += perc[i];
  }
  int rest = 100-percCount;
  perc[totalDevices-1] += rest;
  ArrayList<Float> array2 = new ArrayList<Float>();
  for (int i=0 ; i<totalDevices ; i++){
    array2.add(i,(float) (perc[i]*0.01));
  }
  return array2;
}
```

- ✓ Método "algorithmPartFinal": Permite calcular el porcentaje final de bytes a descargar por cada cliente, realizando una combinación de los porcentajes de bytes de acuerdo a la tasa de transferencia y el nivel de batería.

```
public static void algorithmPartFinal(){

   float batteries = 0.05f, times = 0.95f;

   for(int i=0;i<totalDevices;i++){
     if(batteriesArray.get(i)<=15){
       batteries = 0.1f;
       times = 0.9f;
     }
   }
   float finalValue;
   for(int i=0;i<totalDevices;i++){
     finalValue = (float)((batteries*algorithmPart2Batteries.get(i))+
                          (times*algorithmPart2Times.get(i)));
     algorithmPartFinalArray.add(i, finalValue);
   }
   algorithmPartFinalReady = true;
}
```

- ✓ Método "copyFileClient": Permite al cliente escribir el archivo en el socket para la próxima lectura del Group Owner.

```
public static boolean copyFileClient(InputStream in, OutputStream out, Long
fileLength, int deviceNumber,int totaldevices, AlgorithmData
algorithmData){

   byte buf[] = new byte[1024*8];
   int totalLaps = (int) (fileLength/(1024*8)); // 8.0036
   float remainingBytes = fileLength%(1024*8),lapsAmount = 0;
   int len, lenAmount = 0,lapsIntAmount =0,lenLaps = 0;
   long cycles = 1000, start = 0, end, totalTime,skippedAmount = 0;
   Percentage = 0;
   float[] laps = new float [totaldevices];
   int []lapsInt = new int[totaldevices];

   if (remainingBytes > 0)
      totalLaps++;
   for (int i=0; i<totaldevices ; i++){
      laps[i] = (algorithmData.getAlgorithmFinal().get(i) * totalLaps);
      lapsInt[i] = (int) laps[i];
      lapsAmount += laps[i];
      lapsIntAmount += lapsInt[i];
   }
   int rest = (int) lapsAmount - lapsIntAmount;
   lapsInt[totaldevices-1] += rest;

   try{
      for (int j = 1; j <= deviceNumber ; j++){
         if (j == deviceNumber){
         in.skip(skippedAmount);
         start = System.currentTimeMillis();

         for (int k = 0; k < lapsInt[j-1] ; k++){
           len = in.read(buf);
           out.write(buf, 0, len);
           lenLaps ++;
           try{
              lenAmount += len;
              if ((lapsInt[j-1]*(1024*8)) > 0)
                 Percentage = (int) ((lenLaps * 100) / (lapsInt[j-1]) );
              mProgressDialog[0].setProgress(Percentage);
           } catch (Exception e) {}
           if (WiFiDirectActivity.apagar || cancelProgressDialog){
              out.close();
              in.close();
              cancelProgressDialog = false;
              end = System.currentTimeMillis();
              totalTime = (end-start)/cycles;
              return true;
           }
```

```
            }
         }
         skippedAmount = skippedAmount + (lapsInt[j-1]*8192);
      }
      end = System.currentTimeMillis();
      totalTime = (end-start)/cycles;
      out.close();
      in.close();
      if (mProgressDialog != null) {
         if (mProgressDialog[0].isShowing()) {
            mProgressDialog[0].cancel();
            mProgressDialog[0] = null;
            Percentage = 0;
         }
      }
    } catch (IOException e) {
    return false;
  }
  return true;
}
```

- ✓ Método "copyFileServer": Permite leer los bytes del archivo, escrito por el cliente a través del socket. Luego de la lectura de los datos, el Group Owner realiza la escritura de los archivos que cada cliente transfirió para que sean unidos.

```
public static boolean copyFileServer(InputStream in, OutputStream out, int
ndevice )
{
   byte buf[] = new byte[1024*8];
   int len, laps = 1, lenAmount = 0;
   long start, end, totalTime, mil=1000;
   try {
      start = System.currentTimeMillis();
      startTimeArray.add(ndevice,start);
      while ((len = in.read(buf)) != -1) {

      if(laps%1000 == 0) Log.i(BAN, "LAPS: " +laps);
         out.write(buf, 0 , len);
         lenAmount = lenAmount+len;
         laps++;
         if(WiFiDirectActivity.apagar)
         {
            out.close();
            in.close();
            end = System.currentTimeMillis();
            totalTime = (end-start)/mil;
            return true;
         }
      }
      end = System.currentTimeMillis();
      totalTime = (end-start)/mil;
```

```
      out.close();
      in.close();
   } catch (IOException e) {
       return false;
     }
   join++;
   return true;
}
```

✓ Método "joinFile": Permite la unión de las partes transferidas por cada cliente.

```
protected void joinFile(){
  try {
    publishProgress("Uniendo archivos");
    OutputStream os = null;
    byte[] joinBuff = new byte[1024*16];
    os = new FileOutputStream(Environment.getExternalStorageDirectory()+
           "/UBBifiDirect/"+finalFileName);
    InputStream[] isList = new InputStream[totalDevices];
    int cycles = 1,len;
    for(int i=0;i<totalDevices;i++){
isList[i] = new
FileInputStream(Environment.getExternalStorageDirectory()+
      "/UBBifiDirect/"+"Part0"+(i+1)+"_"+finalFileName);
    }
    int lenAmount = 0;
    for(int j=0;j<totalDevices;j++){
Toast.makeText(context, "Recibiendo
"+finalfileName,Toast.LENGTH_SHORT).show();
      while((len=isList[j].read(joinBuff))!=-1){
      if(cycles % 6000 == 0) {
      publishProgress("Uniendo archivos");
      }
      os.write(joinBuff,0,len);
      lenAmount = lenAmount + len;
      cycles++;
    }
    endTimeJoin = System.currentTimeMillis();
    long oldestTime = searchOldestTime();
    long totalTime = endTimeJoin - oldestTime;
    publishProgress("Archivo copiado /UBBiFiDirect/"+finalFileName);
    os.close();
    for(int i=0;i<totalDevices;i++)
      isList[i].close();
   }catch (Exception e) {}
}
```

13 ANEXO: PLANIFICACIÓN INICIA DEL PROYECTO

13.1 Carta Gantt

Id	Nombre de tarea	Duración	Comienzo	Fin
1				
2	Inicio	0 días	lun 16-03-15	lun 16-03-15
3	Reunión profesor guía	1 día	lun 16-03-15	lun 16-03-15
4	Realización Propuesta	5 días	lun 23-03-15	vie 27-03-15
5	Lectura documentación Android	7 días	mié 01-04-15	jue 09-04-15
6	Aplicación ejemplos código Android	10 días	lun 13-04-15	vie 24-04-15
7	**Fase 1: Análisis**	**36 días**	**lun 27-04-15**	**lun 15-06-15**
8	Búsqueda documentos relacionados	5 días	lun 27-04-15	vie 01-05-15
9	Lectura documentos Wifi Direct	5 días	lun 04-05-15	vie 08-05-15
10	Especificación de requerimientos del Software	5 días	lun 11-05-15	vie 15-05-15
11	Análisis sistema	5 días	lun 18-05-15	vie 22-05-15
12	Analizar código fuente Wifi P2P	15 días	lun 25-05-15	vie 12-06-15
13	Reunión profesor guía	1 día	lun 15-06-15	lun 15-06-15
14	**Fase 2: Diseño**	**20 días**	**mié 17-06-15**	**mar 14-07-15**
15	Reunión profesor guía	1 día	mié 17-06-15	mié 17-06-15
16	Definir componente de la App	5 días	jue 18-06-15	mié 24-06-15
17	Revisión modelo Wifi Direct	5 días	jue 25-06-15	mié 01-07-15
18	Desarrollar modelo e íconos	5 días	jue 02-07-15	mié 08-07-15
19	Avance informe	4 días	jue 09-07-15	mar 14-07-15
20	**Fase 3: Desarrollo y Codificación**	**129 días**	**lun 20-07-15**	**jue 14-01-16**
21	Reunión profesor guía	2 días	lun 20-07-15	mar 21-07-15
22	Revisión código Wifi direct	5 días	mié 22-07-15	mar 28-07-15
23	Revisión modelo Wifi Direct	5 días	lun 03-08-15	vie 07-08-15
24	Conexión wifi P2P (Direct)	5 días	lun 10-08-15	vie 14-08-15
25	Estructura cliente-group owner	3 días	lun 17-08-15	mié 19-08-15
26	Prueba en dispositivos	5 días	jue 20-08-15	mié 26-08-15
27	Implementación técnicas de medición de batería	4 días	lun 31-08-15	jue 03-09-15
28	Búsqueda técnicas de medición de conectividad	5 días	lun 07-09-15	vie 11-09-15
29	Revisión protocolos de transmisión de datos	3 días	lun 21-09-15	mié 23-09-15
30	Mejora implementación sockets transmisión de datos	10 días	vie 25-09-15	jue 08-10-15
31	Transmisión distintos tipos de archivos	3 días	vie 09-10-15	mar 13-10-15
32	Prueba en dispositivos y corrección de errores	6 días	mié 14-10-15	mié 21-10-15
33	Reunión profesor guía	1 día	lun 26-10-15	lun 26-10-15
34	Avance informe	5 días	mié 28-10-15	mar 03-11-15
35	Avance diseño interfaz de usuario	6 días	mié 04-11-15	mié 11-11-15
36	Creación buscador de archivos	3 días	jue 12-11-15	lun 16-11-15
37	Implementación conexión de dispositivos en paralelo	10 días	lun 16-11-15	vie 27-11-15
38	Reunión profesor guía	2 días	lun 30-11-15	mar 01-12-15
39	Revisión documentación descarga en paralelo	5 días	mié 02-12-15	mar 08-12-15
40	Implementación técnicas de descarga en paralelo	10 días	mié 09-12-15	mar 22-12-15
41	Implementación técnicas medición de descarga	4 días	mié 23-12-15	lun 28-12-15
42	Implementación unión de archivos	5 días	vie 01-01-16	jue 07-01-16
43	Implementación mejoras de la app	5 días	vie 08-01-16	jue 14-01-16
44	Avance diseño interfaz de usuario	4 días	vie 15-01-16	mié 20-01-16
45	Verificación y corrección de errorres	5 días	lun 25-01-16	vie 29-01-16
46	Reunión profesor guía	1 día	mar 02-02-16	mar 02-02-16
47	Avance informe Final	16 días	mié 03-02-16	mié 24-02-16
48	Pruebas de la Aplicación	6 días	jue 25-02-16	jue 03-03-16
49	Entrega final del informe	1 día	vie 04-03-16	vie 04-03-16

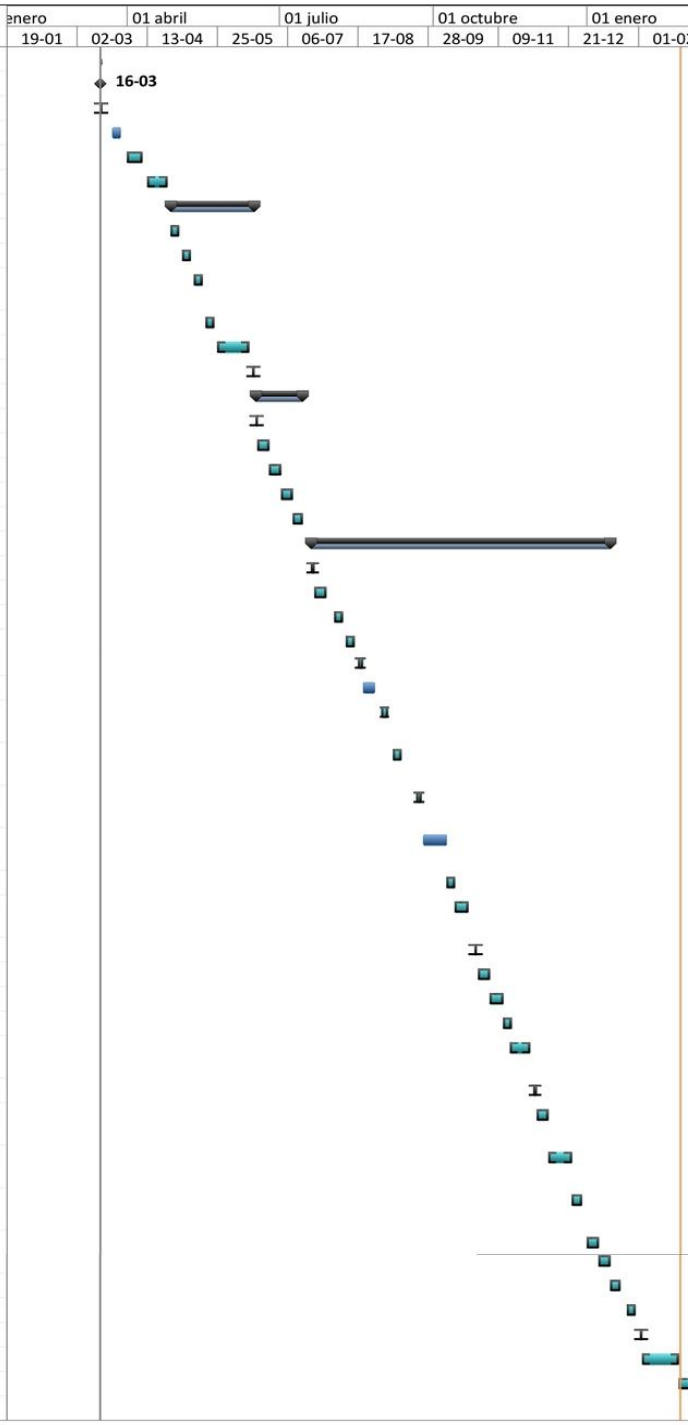

Figura 43: Carta Gantt