



Instituto de Ingeniería Eléctrica
Universidad de la República Oriental del Uruguay
Setiembre 2009

Integrantes:

Federico Aristimuño
Luciano Ferrari
Maximiliano Flores
Ignacio Varoli

Tutor:

Ing. Javier Pereira

Agradecimientos

A nuestras familias y amigos, por su apoyo constante en todo momento a lo largo de este proyecto y del proceso de formación de todos estos años.

A nuestro tutor, el Ing. Javier Pereira.

A quienes colaboraron directamente con el proyecto, en particular a Alexander Murdoch, Rodrigo "el Buda" Toledo y Juan "Paco" Saavedra.

Contenido

1. Introducción	1
1.1. Contexto	1
1.2. Motivación	1
1.3. Descripción del Proyecto	2
1.4. Objetivos	3
2. Descripción Teórica del Proyecto	5
2.1. Introducción	5
2.2. Alcance del Proyecto	5
2.3. Arquitectura de la Red	6
2.3.1. Punto de Acceso (uAP)	7
2.3.2. Terminal de Usuario (uCel)	10
2.3.3. Servidor Central (uServer)	13
2.3.4. Base de Datos (uDataBase)	15
2.4. Desarrollo del Software	16
2.4.1. Lenguaje utilizado	16
2.4.2. Clases y Estructuras	17
2.4.2.1. uAP	18
2.4.2.2. uCel	21
2.4.2.3. uServer	25
2.4.2.4. uDataBase	30
2.4.3. Librerías Utilizadas	32
2.4.3.1. uAP	33
2.4.3.2. uCel	33
2.4.3.3. uServer	34
2.4.3.4. uDataBase	34
2.5. Requerimientos de la Aplicación	35
2.6. Casos de Uso	36
3. Protocolo de Comunicación Umaguma	39
3.1. Introducción	39
3.2. Asignación de Direcciones	40
3.3. Estructura y Entramado	42
3.3.1. Introducción	42
3.3.2. Mensajes de Datos	43
3.3.2.1. Mensajes de Texto	43

3.3.2.2.	Mensajes de Texto Store and Forward	45
3.3.2.3.	Mensajes Multimedia	48
3.3.2.3.1.	Mensajes Multimedia de Imágenes	49
3.3.2.3.2.	Mensajes Multimedia de Audio	50
3.3.3.	Mensajes de Control	52
3.3.3.1.	Mensaje de Control 255	52
3.3.3.2.	Mensaje de Control 254	53
3.3.3.3.	Mensaje de Control 253	56
3.3.3.4.	Mensaje de Control 252	57
3.3.3.5.	Mensaje de Control 251	59
3.3.3.6.	Mensaje de Control 250	61
3.3.3.7.	Mensaje de Control 248	61
4.	Desarrollo de la Red	63
4.1	uAP	63
4.1.1	Introducción	63
4.1.2	Modalidades de trabajo e interfaces	64
4.1.3	Modalidad de Trabajo con uServer	65
4.1.3.1	Inicio de uAP	65
4.1.3.2	Ingreso de usuarios a la Red	67
4.1.3.3	Envío y Recepción de Mensajes de Texto	69
4.1.3.4	Mensajes de Datos Especiales	71
4.1.3.5	Mensajes de Control	72
4.1.3.6	Envío de Mensajes desde el uAP	72
4.1.3.7	Envío de estadísticas hacia la Base de Datos	72
4.1.4	Modalidad de Trabajo Stand-Along	73
4.2	uCel	73
4.2.1	Introducción	73
4.2.2	Establecimiento de Conexión	74
4.2.3	Manejo de Usuarios	76
4.2.4	Mensajes de Texto Especiales	77
4.2.5	Mensajes Multimedia	78
4.2.6	Fin de Sesión	79
4.2.7	Interfaz Grafica de Usuario	80
4.3	uServer	80
4.3.1	Introducción	80
4.3.2	Funcionamiento General	81
4.3.3	Manejo de conexión con un nuevo uAP	83

4.3.4	Procesamiento de nuevas tramas	84
4.3.5	Perdida de conexión con un uAP	87
4.3.6	Interacción con la Base de Datos	88
4.4	uDataBase	90
4.4.1	Introducción	90
4.4.2	Elección de la Base de Datos y adaptación al sistema	91
4.4.3	Usos de la Base de Datos	94
4.4.3.1	Uso de la Base de Datos en Store and Forward	94
4.4.3.2	Uso de la Base de Datos en Sistema de Estadísticas	97
5.	Ejemplos de Uso	101
5.1.	Conexión de un uAP al uServer	101
5.2.	Perdida de conexión entre un uAP y el uServer	103
5.3.	Conexión de un celular a un AP	104
5.4.	Perdida de conexión con un Terminal	107
5.5.	Envío de mensajes de Texto desde un uCel	108
5.6.	Envío de Imágenes desde un Terminal	110
5.7.	Envío de archivos de voz desde un Terminal	111
5.8.	Store and Forward	111
5.8.1	Envío de un mensaje utilizando MAC de destino	112
5.8.2	Recepción de un mensaje Store and Forward.....	113
6.	Sistema de Estadísticas de la Red	115
6.1.	Motivación	115
6.2.	Implementación	116
6.2.1.	Software de Gestión de Estadísticas	116
6.2.2.	Definición de criterios de medición de Estadísticas	117
6.2.3.	Funcionamiento	118
6.2.4.	Procesamiento de Datos y Resultados Finales	120
6.2.4.1.	Tabla de Resultados Finales	120
6.2.4.2.	Gráficos de Resultados Finales	125
7.	Gestión de la Red y Políticas de Conexión	127
7.1.	Introducción	127
7.2.	Descripción	127
7.2.1.	Acceso con Contraseña	127
7.2.2.	Visibilidad	128
7.2.3.	Separación de la Red en zonas	129
7.2.4.	Desconexiones Forzadas	129
7.2.5.	Envío de contenidos desde Nodos	130

7.3.	Posibles Usos de las Políticas	131
8.	Interfaz Gráfica	133
8.1.	Introducción y Objetivos	133
8.2.	Herramientas de Desarrollo	133
8.2.1.	Interfaz Grafica de Terminales	133
8.2.2.	Interfaz Grafica de Nodos de Core	135
8.3.	Interfaces gráficas desarrolladas	139
8.3.1.	Interfaz gráfica de Terminales	139
8.3.2.	Interfaz gráfica de Puntos de Acceso	144
8.3.3.	Interfaz gráfica del Servidor	149
8.3.4.	Interfaz gráfica del Gestor de Estadísticas	158
9.	Resultados	165
9.1.	Introducción	165
9.2.	Objetivos Alcanzados	165
9.3.	Pruebas de Desempeño	167
10.	Conclusiones	171
Anexos	173
Anexo 1:	Bluetooth	173
Anexo 2:	TCP/IP	193
Anexo 3:	SQL	199
Anexo 4:	J2SE	203
Anexo 5:	J2ME	207
Referencias	209

Acrónimos

ACL	Asynchronous Connectionless Link
AM_ADDR	Active Member Address
AR_ADDR	Access Request Address
BD_ADDR	Bluetooth Device Address
CID	Caller ID
ETSI	European Telecommunications Standards Institute.
FHSS	Frequency Hopping Spread Spectrum
FP	File Transfer Profile
GAP	Generic Access Profile
HCI	Host Controller Interface
IEEE	Institute of Electronic and Electrical Engineering
IP	Internet Protocol
irDA	Infrared Data Association
ISM	Industrial, Scientific and Medical
L2CAP	Logical Link Control and Adaptation Protocol
LAP	Lower Address Portion
LC	Link Controller
LM	Link Manager
LMP	Link Manager Protocol
LSB	Least Significant Bit
MAC	Media Access Control
MSB	Most Significant Bit
NAP	Non-significant Address Portion

OBEX	Object EXchange Protocol
OPP	Object Push Profile
OSI	Open System Interconnection
PDA	Personal Digital Assistant
PDU	Protocol Data Unit
PM_ADDR	Parked Member Address
PTT	Push To Talk
QoS	Quality of Service
RFCOMM	Radio Frequency Communication
S&F	Store & Forward
SCO	Synchronous Connection Oriented link
SDAP	Service Discovery Application Profile
SDP	Service Discovery Protocol
SIG	Special Interest Group
SPP	Serial Port Profile
SP	Synchronization Profile
TCP	Transmission Control Protocol
TCS	Telephone Control protocol Specification
TDD	Time Division Duplex
TDM	Time Division Multiplexing
UAP	Upper Address Portion
uAP	Umaguma Access Point
uServer	Umaguma Server
uCel	Umaguma Terminal
WPAN	Wireless Personal Area Network

Capítulo 1: Introducción

1.1. Contexto

Este documento pretende describir de la manera más clara y precisa posible el proceso de desarrollo realizado para el proyecto de fin de carrera Umaguma. Este proyecto fue desarrollado durante los años 2008 y 2009 por estudiantes de la Facultad de Ingeniería perteneciente a la Universidad de la República Oriental del Uruguay. Corresponde a la carrera de Ingeniería Eléctrica, más concretamente al área de Telecomunicaciones. Su objetivo principal es llevar a cabo el diseño e implementación de un sistema completo que permita el intercambio de información, así como también brindar distintos servicios de telecomunicaciones a sus usuarios mediante el uso de dispositivos móvil, en particular teléfonos celulares. El sistema utilizaría para esto las prestaciones de tecnologías como Bluetooth y TCP/IP. Pretende ser una solución dinámica y flexible a problemas cotidianos de comunicación entre personas, capaz de adaptarse a una gran variedad de situaciones.

A lo largo de este documento se describirán las diferentes características técnicas del proyecto, así como también el proceso de desarrollo que llevó a este sistema a la práctica. Se pretende brindar al lector una descripción técnica y precisa, pero de la manera más clara y concisa posible. Para esto, en primer lugar se realizará una pequeña introducción a las características del proyecto, para luego pasar a una descripción teórica más profunda del comportamiento del sistema. Luego se describirá cómo fueron desarrollados cada uno de los componentes de la red, explicando los distintos problemas que se presentaron en cada etapa y las soluciones empleadas para cada caso. También se dedicará un capítulo a explicar el protocolo de comunicación elaborado para que todos los niveles de la red se comuniquen correctamente.

Después de haber establecido en el lector un marco teórico lo más claro y técnico posible, se proseguirá a describir diferentes situaciones prácticas que se dan en el uso típico de la red Umaguma, pretendiendo mostrar su utilidad en ámbitos reales. Finalmente, se dejarán planteados cuales fueron los resultados obtenidos y las conclusiones a las que se llegó tras el esfuerzo y dedicación de cada uno de los integrantes de este proyecto.

1.2. Motivación

La motivación principal del proyecto fue desarrollar un sistema de telecomunicaciones que se pudiera adaptar a diferentes situaciones de la vida real,

partiendo desde una estructura genérica. Para encarar el diseño del sistema, se hizo hincapié principalmente en dos ideas:

- **Distribución de contenidos:** Crear una red de comunicaciones que permitiera la distribución dinámica de contenidos diversos hacia los usuarios (como propagandas, promociones o demás información de interés). Un ejemplo de esto sería aplicar el sistema en los locales de un centro comercial, permitiendo a una empresa promocionar sus productos de manera eficiente, mejorando el alcance a sus clientes.
- **Comunicación entre usuarios:** Llevar a cabo la implementación de una red corporativa de mensajería que, por ejemplo, permita a los empleados de una empresa el intercambio de mensajes de texto o multimedia sin la necesidad de utilizar la red celular, obteniéndose de esta manera una comunicación fluida y económica, que se ejecuta sobre una red privada.

1.3. Descripción del Proyecto

La idea del proyecto consiste en desarrollar una red de telecomunicaciones capaz de brindar servicios de mensajería multimedia haciendo uso de las tecnologías de comunicaciones Bluetooth y TCP/IP. El sistema está constituido físicamente por tres partes relevantes:

- **Un Servidor central, denominado nodo uServer, el cual es único dentro de la red, y tiene como funciones principales interconectar nodos de menor jerarquía y enrutar la información entre ellos.**
- **Puntos de Acceso, denominados uAP. Estos nodos se encuentran conectados constantemente al uServer utilizando para ello TCP/IP y permiten el acceso de los usuarios a la red.**
- **Terminales, denominados nodo uCel, mediante los cuales el usuario hace uso del sistema. Estos nodos se implementan sobre dispositivos móviles que se conectan a los Puntos de Acceso mediante tecnología inalámbrica Bluetooth.**

Se diseñó y desarrolló un protocolo para el intercambio de mensajes de datos y control de la red que permite a los usuarios hacer uso de los servicios de manera eficiente, así como también mantener un constante monitoreo del estado de la red. Por otra parte, se buscó construir el sistema de manera escalable, permitiendo incrementar las zonas de cobertura mediante la expansión hacia de nuevos nodos de acceso de manera fácil y dinámica.

A continuación se muestra un diagrama esquemático de la arquitectura básica de la red.

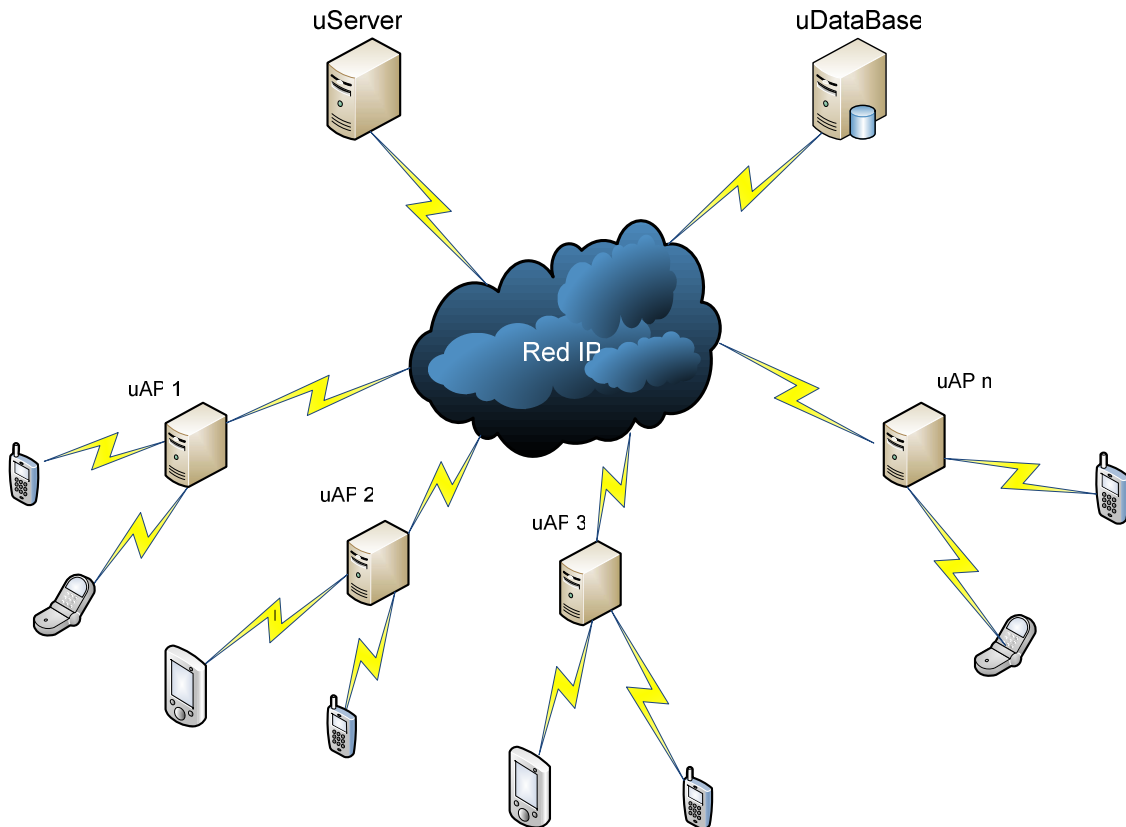


Figura 1.1 – Diagrama de arquitectura básica de la red

1.4. Objetivos

Los objetivos iniciales para este proyecto, definidos originalmente en una primera etapa de planificación (realizada previamente a comenzar el desarrollo) se describirán a continuación. Hay que destacar que durante el transcurso del proyecto estos objetivos se fueron ajustando, refinando y expandiéndose, intentando ser adaptados a las nuevas situaciones que sus desarrolladores iban enfrentando.

Dichos objetivos son:

- Intercambio de información de manera bilateral entre una PC y un dispositivo móvil, utilizando para esto el protocolo Bluetooth a través de una aplicación desarrollada en Java.
- Desarrollo de una aplicación Java, capaz de adaptarse a las características de hardware de un dispositivo móvil.
- Desarrollo Puntos de Acceso que se comuniquen con dispositivos móviles a través del protocolo Bluetooth, y al mismo tiempo lo hagan con un nodo de mayor jerarquía a través del protocolo TCP/IP.
- Desarrollo de un Servidor central que interactúe simultáneamente con todos los Puntos de Acceso intercambiando datos e información de control entre ellos.
- Diseño e implementación de un protocolo de comunicaciones que permita el correcto intercambio de información de control entre todos los niveles de la red.
- Implementación de un sistema de mensajería Store & Forward que permita el almacenamiento de mensajes de texto en una Base de Datos. Estos mensajes deberán ser entregados a los destinos correspondientes una vez que estos ingresen a la red.
- Intercambio de mensajes multimedia entre los distintos clientes de la red.
- Implementación de un sistema de gestión para establecer políticas entre grupos de usuarios y controlar el acceso al sistema
- Implementación de un sistema de contadores y estadísticas que informe dinámicamente el estado de los nodos.

Capítulo 2: Descripción teórica del Proyecto

2.1. Introducción

En este capítulo se explicarán las diferentes características técnicas del proyecto desde el punto de vista de su planificación, tanto en la arquitectura de red como en la descripción de cada nodo y su diseño. En capítulos posteriores se detallará la implementación del sistema así como las distintas dificultades que se debió afrontar durante el proceso de desarrollo y de qué manera fueron sorteadas.

2.2. Alcance del Proyecto

Como se mencionó anteriormente, Umaguma surge ante la necesidad de los autores de realizar un desarrollo de ingeniería como proyecto final para la carrera de Ingeniería Eléctrica de la Universidad de la República Oriental del Uruguay, y como tal éste debe cubrir ciertos objetivos, tanto en lo técnico como en lo logístico. Es por esto que el proyecto fue precedido por la creación de un plan de trabajo que derivó en un proceso de gestión, en el cual se llevó a cabo lo planificado. Fue, justamente en la etapa de planificación, que se definió el alcance del proyecto, acordando (al menos en un nivel tentativo) los objetivos que -una vez alcanzados- dieran por exitoso el proyecto.

Si bien el alcance definitivo y detallado del proyecto lo terminó definiendo el avance de éste en el tiempo y las dificultades que surgían durante el proceso, podemos afirmar que los objetivos planteados inicialmente se mantuvieron hasta el final, llegando a cumplirse satisfactoriamente e incluso realizándose agregados que inicialmente no estaban planeados.

La idea original para nuestro proyecto fue desarrollar e implementar una red de datos que nos permita establecer comunicaciones entre dispositivos móviles (típicamente teléfonos celulares, pudiéndose utilizar también otros dispositivos como Palms, SmartPhones, etc). La red de acceso utilizaría tecnología Bluetooth y la red de núcleo sería TCP/IP. El sistema desarrollado permitiría al usuario utilizar una amplia gama de servicios multimedia como por ejemplo el manejo de sesiones de usuario, el envío de mensajería de texto, la captura y transmisión de imágenes y la mensajería con envío diferido (Store and Forward) entre otras cosas. La red debía ser capaz de ser gestionada y administrada, manejando políticas de conexión y seguimiento del tráfico. Para todo esto se debían desarrollar todos los nodos que comprenderían su arquitectura (los cuales inicialmente no

estaban definidos en su totalidad) y se debían crear protocolos que manejaran la comunicación entre éstos.

A continuación se pasará a discutir más en profundidad la arquitectura elegida para la red Umaguma y la función de cada uno de los nodos que la compone. Este esquema se terminó de definir luego de una etapa inicial de investigación en la cual se estudiaron las herramientas y tecnologías que -tentativamente- se utilizarían en el proyecto (Bluetooth, TCP/IP, Java 2 Standard Edition, Java 2 Micro Edition, SQL, etc.), llegando a la conclusión de que una topología como la planteada era la más acertada para los objetivos propuestos.

2.3. Arquitectura de la red

En la figura 2.1 se muestra esquemáticamente la arquitectura de la red Umaguma. Se observa que presenta un esquema distribuido en el cual los Terminales de red (nodo llamado uCel) se conectan a diferentes Puntos de Acceso (nodo llamado uAP), los cuales pueden estar separados físicamente representando diferentes zonas en las cuales la red puede tener diferentes comportamientos (este punto será analizado en detalle más adelante). Todos los uAP de la red están interconectados entre si mediante una LAN IP a través de la cual circula todo el trafico interno. Las conexiones se llevan a cabo mediante flujos TCP que interconectan cada uno de los Puntos de Acceso con el nodo uServer (Servidor central de la red) y con el nodo uDataBase (Base de Datos).

A continuación introduciremos cada uno de los nodos que presenta la arquitectura propuesta, explicando más detalladamente sus funciones y características desde el punto de vista teórico, los detalles de la implementación final se describirán en el capítulo 4 (Desarrollo de la red).

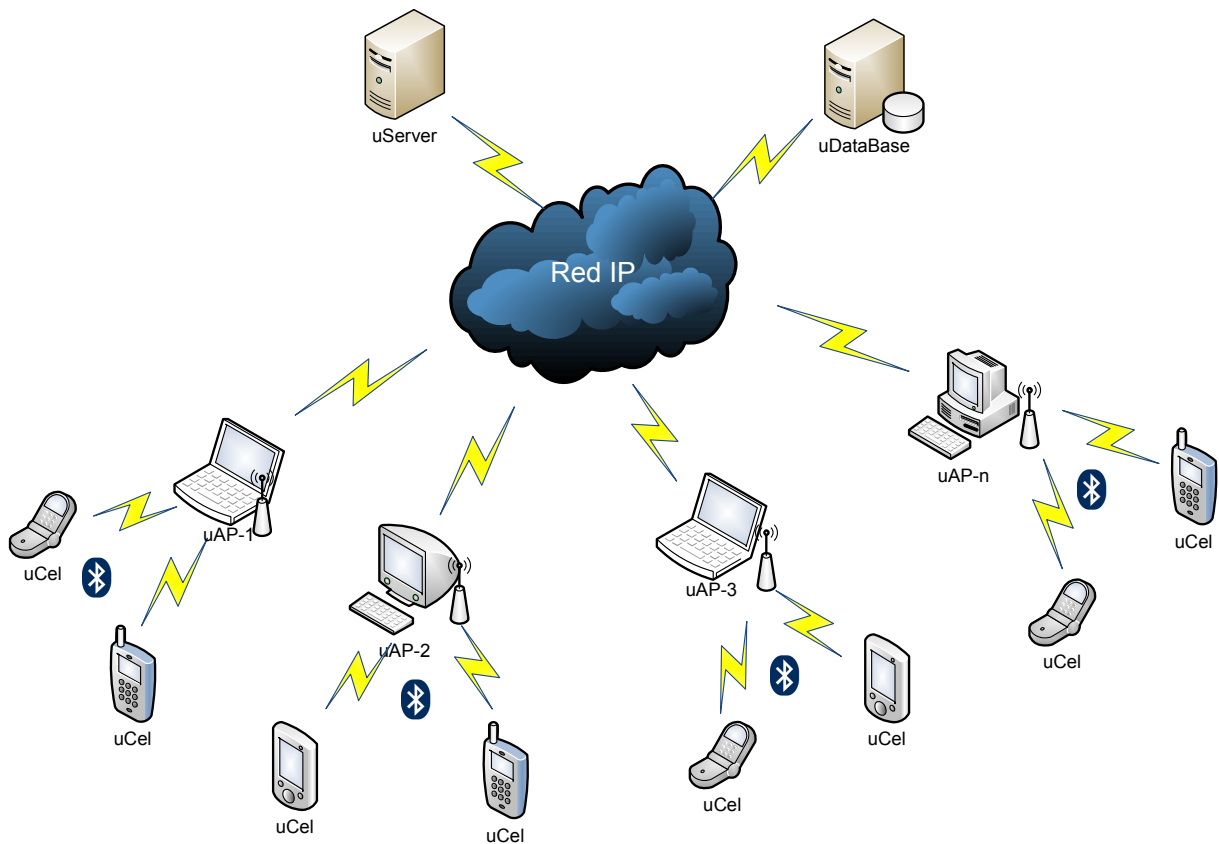


Figura 2.1 - Diagrama general de la red

2.3.1 Punto de Acceso (uAP)

El primer elemento del que se hablará es el uAP, este nodo tiene la función de ser la puerta de enlace a través de la cual el usuario ingresa a la red y es también el encargado de realizar la gestión con éste durante el transcurso de la comunicación. Básicamente está compuesto por el software que maneja todas las funciones de red y el hardware que da soporte a las diferentes tecnologías necesarias para esto, como son:

- La maquina virtual Java sobre la cual se ejecutará el programa.
- Los drivers, tarjetas y puertos necesarias para la comunicación TCP/IP.
- El stack de protocolos que maneja la comunicación Bluetooth.

La comunicación con el Terminal de usuario se realiza mediante tecnología Bluetooth, para esto utilizamos dispositivos USB que son manejados directamente por el software a

través de la Interfaz de Programación de Aplicaciones (API de Java) JSR 82, en particular se utiliza el paquete de librerías BlueCove.

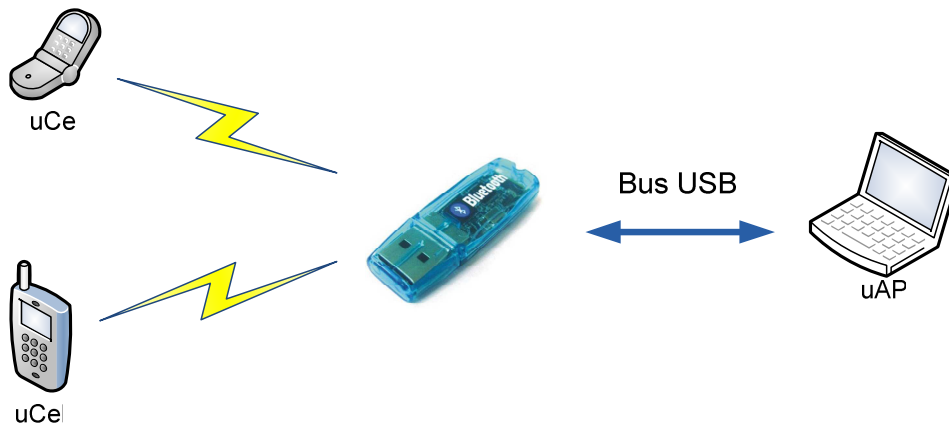


Figura 2.2 - Diagrama general de la red

En cada uAP, la interfaz de radio Bluetooth está identificada unívocamente a nivel de enlace físico (capa 2) por su dirección MAC, la cual se encuentra pre-definida “de fábrica” en cada dispositivo USB de forma única, en tanto que para la conexión hacia la red interna nos es relevante únicamente el valor de la dirección IP del nodo, siempre y cuando asumamos que la LAN que interconecta los nodos se encuentra correctamente implementada. Por otro lado, haciendo una analogía con la capa 3 del modelo de referencia OSI, podemos decir que a nivel de ruteo de red cada uAP está unívocamente identificado mediante su dirección Umaguma global, la cual es asignada por el uServer durante el establecimiento de la conexión, en el momento que el uAP ingresa a la red. Este enfoque implica asumir que si bien todas las conexiones punto a punto presentes (TCP/IP y Bluetooth) son construidas sobre capas altas dentro de cada tecnología, para nosotros se comportan como enlaces transparentes, o sea como una suerte de capa 2.

El uAP es el encargado de aceptar nuevas conexiones de Terminales que deseen ingresar a la red, manejando el establecimiento de la conexión y reportando al sistema el ingreso de este nuevo usuario, esto implica que todos los elementos de la red deben actualizar su lista de destinos para integrar al nuevo elemento. Cuando se da la situación anterior el uAP es capaz, según la situación lo requiera, de solicitar al usuario el ingreso de contraseñas y no permitir su ingreso en caso de no ser estas correctas. Una vez que el usuario logró ingresar a la red el uAP debe reportarle a éste el estado actual del sistema, reenviándole tablas generadas por el uServer con los usuarios presentes (y visibles) y sus respectivas direcciones de red. También debe ser capaz de manejar el cierre de la conexión, reportando al resto de la red la baja del usuario para que éstos actualicen sus tablas.

Probablemente la función que durante más tiempo efectúa el uAP a lo largo de una conexión es la de realizar la conmutación de los mensajes que transitan por la red, tanto a nivel local como global según lo requiera la comunicación. Por ejemplo si un mensaje es enviado desde un Terminal hacia otro cuando ambos se encuentran conectados al mismo uAP la conmutación se realiza localmente y solo dentro del uAP. Si en cambio los Terminales se conectan a distintos uAP, entonces la conmutación se hará globalmente enrutando el mensaje hacia el Servidor central, el cual decidirá hacia donde se lo deberá encaminar, esta decisión se toma en base a las direcciones globales, las cuales serán explicadas en un capítulo posterior.

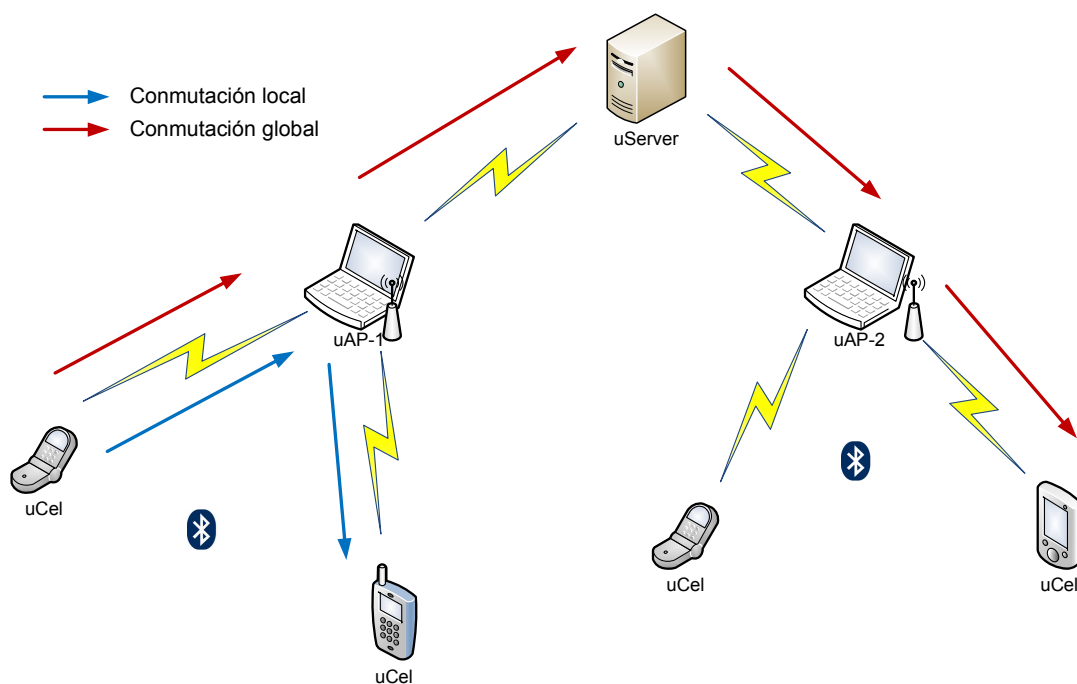


Figura 2.3 – Conmutación local y global

Otra función que debe cumplir el nodo uAP es la de manejar los protocolos utilizados por el uServer para la gestión de la red, básicamente esto implica manejar políticas de visibilidad entre los distintos sectores de la red, así como también la distribución de información, el control de conexión y el manejo de estadísticas del sistema (este tema por su importancia será explicado en un capítulo dedicado), entre otras cosas.

Por último, el uAP debe disponer de una interfaz gráfica que permita al técnico que se encuentre a cargo de la de gestión de red manejar todas sus funcionalidades de manera intuitiva, funcional y dentro de lo posible amigable.

Persiguiendo todos los objetivos que aquí se describieron fue que abordamos el desarrollo de este nodo, buscando sobre la marcha la mejor manera de resolver cada uno de los requerimientos y refinando cada vez más sus funcionalidades.

2.3.2 Terminal de usuario (uCel)

Continuando con la descripción de los requerimientos funcionales para los distintos nodos de la red Umaguma, esta vez le toca el turno al elemento con el cual el usuario interactúa y sobre el que realiza todas las acciones, el Terminal de usuario (nodo uCel). Este nodo es probablemente el más versátil en su implementación, ya que -físicamente- consiste en un teléfono celular común y corriente (pudiendo también utilizarse otro tipo de dispositivo móvil) al que se le carga un software dedicado. Obviamente el dispositivo debe cumplir una serie de requerimientos mínimos que le permitan correr la aplicación. En función de que el dispositivo disponga de más o menos prestaciones tendrá acceso a más o menos funcionalidades o servicios de la red.

En primer lugar, el dispositivo debe ser capaz de ejecutar la aplicación uCel, en caso contrario el usuario no podrá siquiera ingresar en la red. Para esto el terminal debe disponer de una máquina virtual Java y en particular debe manejar el perfil de Java conocido como MIDP, el cual forma parte de la configuración CLDC. Estos esquemas de programación forman parte de la plataforma más típicamente utilizada en la creación de software para teléfonos celulares, conocida como J2ME (Java 2 Micro Edition). Concretamente para el desarrollo de nuestro proyecto utilizamos CLDC versión 1.1 y MIDP versión 2.0. Esto se debe básicamente a que estas versiones resultaron ser la combinación más compatible con la gran mayoría de los dispositivos móviles celulares sobre los que se probó la aplicación.

Como fue descrito anteriormente, este proyecto utilizará tecnología Bluetooth para implementar la red de acceso, es por esto que el siguiente requerimiento básico, sin el cual el usuario no podría utilizar la red, es la interfaz Bluetooth (y el manejo de todo su stack), a través de la cual la aplicación se comunicará con el uAP. Durante el transcurso de la comunicación la aplicación interactuará con esta interfaz, enviando y recibiendo mensajes de datos y de control, así como también manejando el inicio y fin de las conexiones. Para esto se utilizará, -al igual que en caso del uAP- el paquete de librerías BlueCove, el cual implementa la API de Java JSR 82. Esto permite controlar las distintas funciones de Bluetooth de manera muy conveniente mediante clases y métodos escritos en lenguaje Java.

Adicionalmente a los requerimientos básicos del dispositivo (plataforma Java y manejo Bluetooth), la aplicación uCel tendrá funcionalidades de tipo multimedia (imágenes y audio) que -para poder ser utilizadas correctamente- deberán contar con el

soporte necesario en el Terminal. En particular para hacer uso de estos servicios el dispositivo deberá manejar el paquete de Java JSR 135 (MMAPI), a través del cual se controlan las capturas y reproducciones de datos multimedia, además de contar con cámara fotográfica, grabador de audio y altavoz para la reproducción.

La primera funcionalidad que el uCel efectúa en una comunicación es el establecimiento de sesión con el uAP. Esto, en principio, se puede lograr de varias maneras, dependiendo de como el usuario elige a que nodo se la red se quiere conectar. Las opciones se resumen groseramente en dos planteos.

Como primer esquema, el usuario podría seleccionar con que Punto de Acceso se quiere conectar a través de una lista no modificable, fijada previamente y por lo tanto exclusiva para cada implementación del sistema. La conexión se realizaría directamente utilizando la dirección física (MAC) del uAP, la cual se sería ingresada en el código del programa previo a su compilado y distribución. Una desventaja notoria de esta metodología es que no resulta sencillo adaptar la red al ingreso de nuevos Puntos de Acceso sin la necesidad de modificar el programa de cada Terminal, esto le quita cierta flexibilidad al escalado del sistema. En contrapartida tiene como ventaja principal la facilidad y rapidez con que se efectúa el establecimiento de una conexión, dado que de esta manera estamos ahorrando cualquier tipo de demora provocada por la búsqueda de Puntos de Acceso. Una ventaja adicional que tiene utilizar listas fijas es que el sistema cobra carácter de “cerrado” en el sentido de que se podrían diseñar redes en las que sabemos que solo los usuarios que dispongan del programa correcto podrán acceder, resultando estas invisibles para otros dispositivos que dispongan de la aplicación pero seteada para otra implementación de la red.

Como segunda opción el uCel podría realizar una búsqueda de los dispositivos Bluetooth visibles y encuestar a cada uno, listando luego únicamente aquellos que estén ofreciéndose como puertas de enlace a la red Umaguma. Esto tiene como ventaja la versatilidad de adaptarse a cualquier red desplegada, incluso de manera improvisada, sin la necesidad de fijar de antemano las direcciones físicas de cada Punto de Acceso. Como desventaja principal aparece la inevitable demora que requiere el censado de los dispositivos, y a esto debe sumarse una problemática adicional hallada durante el desarrollo del sistema que tiene que ver con el rol de cada dispositivo en una conexión Bluetooth (maestro y esclavo) y con la cantidad de conexiones máximas que puede soportar un dispositivo en cada modalidad. Este punto será descrito mas profundamente en un capítulo posterior y constituyó el elemento principal para decidir quedarnos con el primer esquema de conexión (lista fija).

Una vez que el Terminal estableció la comunicación con el Punto de Acceso éste ingresa a la red. Para permitir el acceso el sistema puede solicitar que se introduzca una contraseña de seguridad, para lo cual el usuario dispone de 3 intentos, en caso de cometer 3 errores la conexión será cerrada. Si se quisiera volver a intentar el ingreso se

debe iniciar nuevamente el programa y establecer nuevamente la conexión con el uAP. Una vez superada esta etapa el usuario ingresa al sistema con lo cual el Terminal se encuentra finalmente conectado a la red. A continuación el uCel recibe la lista de los usuarios que se encuentran conectados, con lo cual ya se está en condiciones de comenzar a comunicarse con ellos.

La modalidad de comunicación más frecuente es el mensaje de texto, este tipo de comunicaron se puede enviar a uno o varios destinatarios simultáneamente, seleccionando los usuarios deseados de la lista de destinos. A su vez, los mensajes recibidos son desplegados en la pantalla principal de la aplicación, mostrándose el último mensaje recibido en el extremo superior de la lista y el resto ordenado según su llegada hacia abajo. Junto con cada mensaje recibido se muestra el remitente correspondiente y el número de mensaje.

Como modalidades adicionales de mensajería de texto tenemos dos casos más. En primer lugar los mensajes enviados por el sistema al uCel (pudiendo provenir tanto del uAP como del uServer), que pueden ser útiles para realizar distribución de contenido o avisos sobre el funcionamiento de la red. En segundo lugar tenemos los mensajes con envío diferido (Store and Forward), esto es, el usuario envía un mensaje de texto a otro usuario que si -en ese momento- no se encuentra conectado a la red, el mensaje es almacenado en la Base de Datos del sistema, realizándose el envío ni bien se conecta el destinatario a la red. El mensaje es recibido con la fecha, hora y remitente correspondiente, apareciendo en la pantalla del Terminal como una ventana emergente.

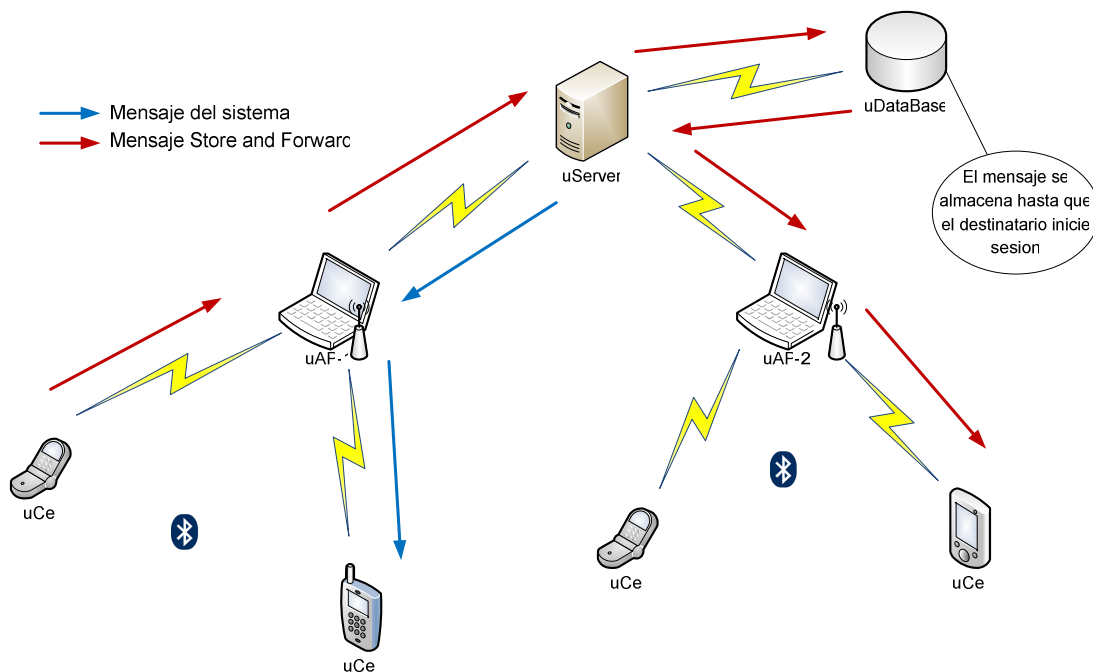


Figura 2.4 – Mensajes del sistema y mensajes Store and Forward

Como último tipo de servicio ofrecido al usuario tenemos los mensajes multimedia. La red Umaguma permite el envío de contenido multimedia, tanto en formato audio como imágenes. Para el caso de los mensajes de audio la modalidad es similar a los servicios PTT brindados por la telefonía celular clásica, esto en el sentido de que el usuario graba un mensaje de voz que luego es enviado al destinatario, el cual lo recibe como mensaje emergente, es decir, sin la necesidad de iniciar una sesión de conversación previamente. El caso de las imágenes es muy similar, en primer lugar se realiza la captura mediante la cámara fotográfica integrada al celular, enviándose luego la foto a los destinos elegidos de la lista de usuarios posibles. Obviamente, para ser capaces de utilizar estos servicios debemos disponer de Terminales que soporten el manejo de estos dispositivos (cámara, micrófono y altavoz). Los detalles de la codificación y transmisión de los mensajes multimedia serán discutidos en el capítulo correspondiente al desarrollo del sistema (capítulo 4).

Con esto queda completada la descripción funcional del nodo uCel desde el punto de vista teórico, en el capítulo 4 (Desarrollo de la red) se describirá la forma en la que se implementaron en la práctica cada una de estas características. Mientras tanto, a continuación se pasará a realizar la descripción del siguiente nodo que compone la red, el uServer.

2.3.3 Servidor (uServer)

El Servidor central de la red Umaguma (nodo uServer) se puede considerar como el corazón mismo del sistema, ya que cumple las funciones de unificar todos los Puntos de Acceso (y por ende también unifica los nodos uCel), recibiendo y enrutando correctamente hacia su destino todos los mensajes que circulan y realizando la gestión completa de la red, tanto en lo que refiere a seguimiento del estado, así como el reporte de estadísticas y manejo de políticas de visibilidad y conexión. Debe también disponer de una interfaz gráfica que le permita al administrador de la red manejar rápida e intuitivamente todos sus recursos.

La primera función que debe realizar este nodo es la de esperar conexiones de los posibles uAP y asignarles direcciones ordenadamente. Es en esta etapa de inicio de sesión que se le debe informar a cada nodo si debe (o no) solicitar contraseña a los usuarios que intenten ingresar al sistema, y -en caso afirmativo- debe informarle cuál debe ser esta clave. Este mecanismo forma parte del sistema de manejo de políticas de conexión y para llevarlo a cabo los nodos intercambian una serie de mensajes de control que pertenecen al protocolo de comunicación desarrollado.

Desde la interfaz gráfica, el administrador debe poder visualizar todos los elementos que integran la red, tanto los uAP como los uCel y sus respectivas direcciones,

desplegándolos jerárquicamente en una estructura de árbol desde donde se podrán realizar acciones sobre los usuarios, tales como enviar mensajes, archivos o incluso dar de baja una sesión en caso de ser necesario.

Es también desde la interfaz gráfica que el administrador será capaz de manejar las políticas de visibilidad, permitiendo que algunos sectores de la red permanezcan ocultos para ciertos usuarios. Cada vez que se realiza un cambio en las políticas el uServer debe enviar los mensajes correspondientes a cada uAP, informando los cambios y realizando las acciones que correspondan, para esto se utiliza el protocolo de comunicación que será explicado en el capítulo 3.

Al igual que el nodo uAP, el servidor será un importante centro de tránsito y conmutación para los mensajes de la red, por lo cual también en este caso es muy conveniente llevar un registro de los eventos que ocurren en él a lo largo del tiempo. Es por esto que el servidor debe disponer de un log que registre detalladamente todo el tráfico que circula por él, a nivel de datos y a nivel de control, también debe desplegar dinámicamente gráficas que reflejen la circulación de la información a través de él.

El intercambio de mensajes de datos y control que se realiza entre los distintos nodos que comprenden el núcleo de la red (servidor, base de datos y puntos de acceso) se realiza mediante conexiones TCP establecidas sobre una red IP. Por lo tanto el equipo donde se ejecuta el software del uServer debe ser capaz de manejar estos protocolos, así como también disponer de interfaces de conexión –a nivel de capa de enlace- hacia la red, típicamente estas interfaces utilizarán las tecnologías Ethernet (cableada) y/o WiFi (inalámbrica).

Debido al esquema de programación utilizado para el desarrollo del servidor, el equipo donde éste se ejecutará deberá contar –al igual que en los restantes nodos de la red- con una máquina virtual Java que sea capaz de interpretar la plataforma J2SE y que disponga de memoria suficiente para ejecutar las tareas del nodo. Este requerimiento tiene como contrapartida la ventaja de que nuestra aplicación será capaz de ejecutarse en múltiples plataformas a nivel de sistema operativo (Windows, Linux, MacOS, etc), siempre que dispongan de la plataforma Java necesaria.

Como ya se comentó anteriormente, este nodo desempeña funciones de conmutación y manejo de sesión entre nodos, esto sin duda lo convierte en un centro de tránsito para los mensajes de la red, por lo cual es sumamente conveniente que, a medida que se realizan las comunicaciones, el nodo sea capaz de mantener un archivo (log) de los eventos que ocurren a lo largo del tiempo, de manera de poder analizar el tráfico circulante en caso de ser necesario. También debe desplegar continuamente un reporte estadístico gráfico indicando el caudal de tráfico que circula por el nodo, que permita realizar estudios de tráfico, carga o desempeño.

Aquí finaliza esta descripción de las características globales del nodo uServer, dejando los pormenores de la implementación para la etapa de desarrollo, la cual implica sin duda nuevos detalles, problemáticas y funcionalidades que en una etapa de planificación no pueden ser contempladas en su totalidad. A continuación se pasará a explicar el funcionamiento del último nodo restante de esta red, la base de datos central.

2.3.4 Base de Datos (uDataBase)

Si bien utilizando únicamente los nodos explicados anteriormente (servidor, puntos de acceso y terminales de usuario) se podría perfectamente llevar a cabo la red propuesta, resultó conveniente desarrollar, como nodo separado, una base de datos central que interactuará tanto con el servidor como con los puntos de acceso y que concentrara las funciones de almacenado y procesamiento masivo de la información de la red. Esta decisión responde principalmente a cuestiones de eficiencia, puesto que la opción alternativa consistía en integrar todas las funciones que realiza este nodo dentro del servidor central, al cual de esta manera probablemente estaríamos asignando demasiada carga de procesamiento.

Desde un punto de vista funcional, la base de datos (nodo uDabaBase) es la encargada de almacenar toda la información que los restantes nodos del sistema necesitarán para hacer uso de sus funciones. Esto lo realiza mediante el manejo de tablas en las cuales se va ingresando y quitando dinámicamente la información según las instrucciones que recibe. Para esto se hará uso del lenguaje SQL, en el cual las diferentes acciones se efectúan mediante sentencias estructuradas que provienen de los usuarios de la base de datos (en este caso el uServer y los uAP).

Una función importante que cumplirá la base de datos en este sistema es la de almacenar –por tiempo indefinido- los mensajes Store and forward que son enviados cuando su destino se encuentran ausente. Para esto se dispone de una tabla en la cual se almacenan todos los mensajes que se encuentran a la espera de su destinatario, junto con la información necesaria para su envío como ser: dirección MAC del destino, información del remitente, fecha y hora del envío. Esta tabla es consultada por el uServer cada vez que un nuevo usuario se conecta a la red, verificando si su dirección física coincide con la de alguno de los posibles mensajes.

Por último tenemos aquellas tablas en las que se registran las estadísticas del sistema. Tanto el servidor como cada uno de los puntos de acceso le envían periódicamente a la base de datos información sobre el tráfico circulante en ellos de manera que se puedan confeccionar reportes estadísticos (gráficos o numéricos) sobre el uso y desempeño de la red en cada momento, distinguiendo por zonas y por tipo de tráfico.

Hasta aquí hemos descrito desde un punto de vista teórico el funcionamiento de la red, así como su arquitectura y las características de cada uno de los nodos que la componen, sin embargo aún no hemos dado mucha información sobre como estará compuesta físicamente y cual será su implementación real. A continuación se explicarán los detalles del desarrollo del prototipo, concretamente se hablará sobre el software que compondrá la red y de como se pensó llevar a cabo su desarrollo. Se discutirá el lenguaje utilizado y se hará un estudio sobre a estructura de clases de cada uno de los nodos.

2.4. Desarrollo del Software

Una vez fijados todos los lineamientos de diseño para este proyecto, estamos en condiciones de encarar las cuestiones relativas a su implementación, en particular se debe resolver todo lo relativo al desarrollo del software de los nodos, esto incluye:

- Resolver sobre que lenguaje/es de programación trabajaremos.
- Realizar una planificación de la estructura del software que desarrollaremos.
- Disponer de un entorno de desarrollo (IDE) adecuado para las necesidades.
- Investigar y obtener todas las librerías necesarias para nuestras aplicaciones.

En las siguientes secciones se discutirá cada uno de estos puntos previo a comenzar con el desarrollo.

2.4.1 Lenguaje utilizado

Llegado el momento de decidirse por el lenguaje de programación sobre el que se desarrollaría el proyecto surgieron inmediatamente una serie de requerimientos básicos que la plataforma elegida debía cubrir:

- El desarrollo se debe realizar utilizando únicamente software libre.
- Debe ser un lenguaje que permita realizar aplicaciones en alto nivel y orientado a objetos.
- Capaz de ejecutarse en múltiples plataformas.
- Manejar diversos protocolos de comunicación.

Teniendo en cuenta estas características fue que Java se posicionó como una opción conveniente para enfrentar el proyecto. En primer lugar porque parecía adaptarse muy bien a todos los requerimientos propuestos para el desarrollo, en segundo lugar -y no

menos importante- por ser un lenguaje de programación conocido por todos los integrantes del grupo y en el cual ya se contaba con cierta experiencia, y en tercer lugar se puede agregar que Java (en su versión para celulares: J2ME) iba a ser obligadamente el lenguaje para la implementación del nodo uCel, por lo cual resultaba coherente utilizar Java para el desarrollo del resto de los nodos.

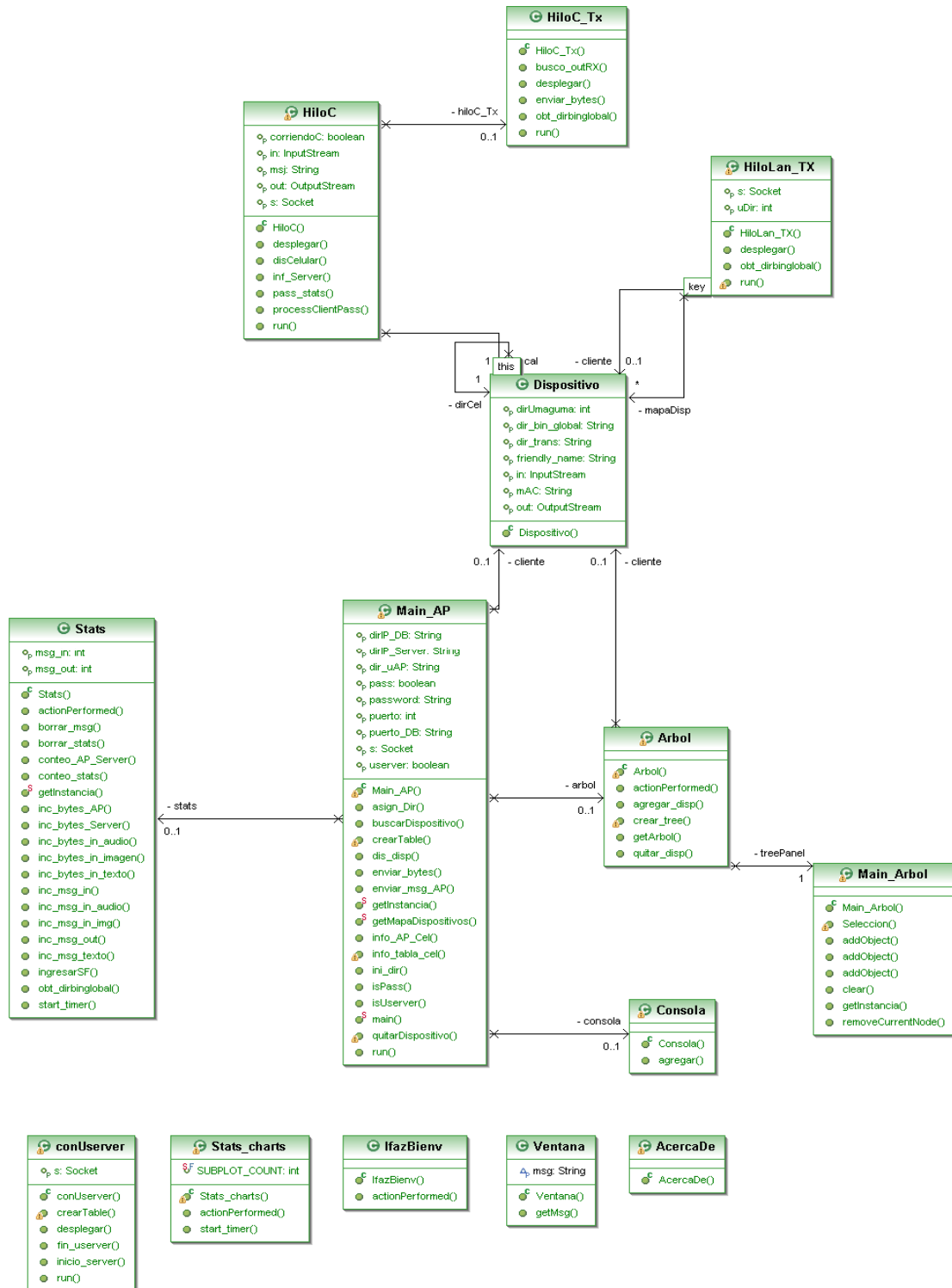
En base a todo lo anterior la decisión final fue la de utilizar J2SE (Java 2 Standard Edition) para implementar servidor, base de datos y puntos de acceso y J2ME (Java 2 Micro Edition) en el terminal de usuario. Para el proceso de desarrollo, simulación y debugging se utilizó la interfaz de desarrollo (IDE) Eclipse, interfaz ya utilizada anteriormente por todos los integrantes del grupo. Se usó también JDK versión 6 como kit de desarrollo en para las aplicaciones en J2SE y Wireless Toolkit 2.5.2 for CLDC en el caso de J2ME.

Todo el proceso de desarrollo del sistema se realizó utilizando el sistema operativo Linux (concretamente la distribución Ubuntu), sobre todo por su conveniencia operativa en lo que refiere a la supervisión de los eventos de red (tanto en TCP/IP como en Bluetooth) a medida que se iban realizando pruebas sobre el sistema. No obstante lo anterior, las aplicaciones fueron desarrolladas pensando en un ámbito de funcionamiento multiplataforma, dado que los programas corren sobre una máquina virtual Java que interactúa con protocolos independientemente de cómo los implemente cada sistema operativo. Por ejemplo las aplicaciones son capaces de interactuar tanto con el stack de protocolos Bluetooth utilizado en Linux (Bluez), como el utilizado por Windows.

2.4.2 Clases y Estructura

A continuación se presentan los diagramas de clases correspondientes a cada uno de los programas que implementan los nodos de la red y se realiza una breve explicación de las funciones de cada clase. En ellos se observa la estructura de las clases y las relaciones entre cada una de ellas.

2.4.2.1 uAP



- **Clase Main_AP:**

Esta clase corre como hilo principal del software uAP, es la encargada de iniciar los hilos secundarios del sistema. Realiza la negociación de conexión de nuevos usuarios a la red, delegándole el control de los mismos a instancias de otra clase y almacenando cada usuario en una colección.
- **Clase Dispositivo:**

Es la encargada de representar a los usuarios en el sistema. Cada vez que un nuevo usuario ingresa a la red se crea una instancia de la clase, la cual almacena todos los parámetros necesarios para la correcta su identificación en la red. Sus instancias son almacenadas en una colección de la clase Main_AP.
- **Clase HiloC:**

Esta clase es la encargada de manejar la recepción de datos que provienen desde los terminales de usuario. Se crea una instancia por usuario que ingresa a la red, la misma es ejecutada como un hilo secundario del sistema. La instancia no se destruye hasta que el usuario no abandona la red.
- **Clase HiloC_Tx:**

Es la encargada de manipular la correcta entrega de mensajes que provienen desde los usuarios. La clase HiloC crea una instancia de esta clase por cada mensaje que se recibe desde un usuario, también es la encargada de hallar el destino y entregar el mensaje correctamente.
- **Clase conUserver:**

Esta clase es una de las primeras en ser instanciada por la clase principal (Main_AP), es la encargada de -primero- manipular la conexión TCP/IP con el uServer, y luego quedarse a la espera de mensajes que provengan desde ese nodo. La instancia entra en loop esperando mensajes y solamente es desechada cuando se cierra la conexión con el uServer. En caso de que el uAP trabaje en modalidad Standalone, esta clase nunca es utilizada.
- **Clase HiloLan_TX:**

Es la encargada de gestionar la correcta entrega de mensajes que provienen desde el uServer. La clase conUserver instancia esta clase cada vez que se recibe un mensaje desde el servidor. Esta instancia es la encargada de enrutar y entregar el mensaje de manera correcta, luego es desechada.

- **Clase IfazBienv:**

Representa la ventana inicial de la aplicación. Su función es que el administrador del nodo configure la modalidad de trabajo del uAP, y los parámetros TCP/IP del servidor y de la Base de Datos en la red.
- **Clase Árbol:**

Es la encargada de representar uno de los paneles de la ventana principal del software. En este panel se despliegan -en formato de árbol- todos los usuarios conectados a la red, sobre los que se puede realizar acciones mediante botones, así como obtener información relacionada al proyecto y cerrar el software.
- **Clase Main_Arbol:**

Esta clase es la encargada de realizar toda la lógica relacionada con el panel descrito en la clase anterior. La misma ejecuta acciones cuando el controlador de la red presiona los botones o realiza clicks sobre los usuarios.
- **Clase Consola:**

Representa otro de los paneles de la ventana principal del software, en este caso se tiene un cuadro de texto en el que se van registrando eventos del nodo, de gran importancia para el administrador, a modo de log.
- **Clase Stats:**

Maneja las estadísticas del nodo al mismo tiempo que va almacenando los datos correspondientes. Cuenta con métodos que se ejecutan a medida que circula tráfico por el nodo, actualizando los valores de las estadísticas. También envía periódicamente información a la base de datos central.
- **Clase Stats_Charts:**

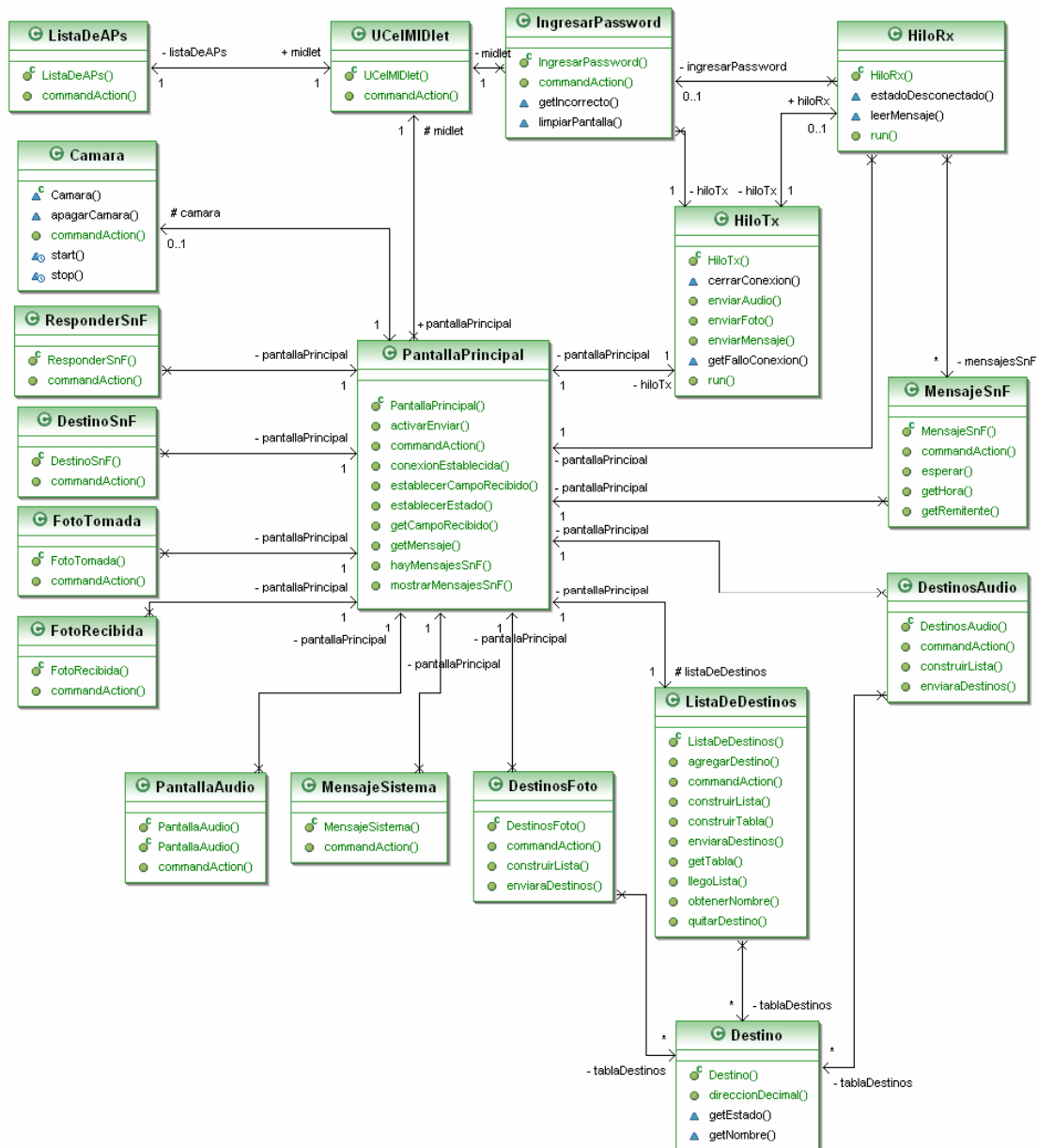
Es la encargada de representar el tercer y último panel de la ventana principal, en el cual se tienen gráficas que muestran a cada instante los datos estadísticos del nodo, permitiendo al administrador obtener una perspectiva general del tráfico con un simple vistazo.
- **Clase Ventana:**

Es la encargada de desplegar la ventana utilizada por el controlador de la red para enviar mensajes de texto a los celulares conectados al uAP.

- **Clase Acerca De:**

Esta clase representa la interfaz grafica utilizada para desplegar información relativa al proyecto Umaguma.

2.4.2.2 uCel



- **Clase UCelMIDlet:**

Esta es la clase principal de la aplicación, por lo que hereda todas las características de la clase MIDlet. En ella se definen los métodos de inicio, pausa y finalización del programa. La primera acción que realiza es mostrar la pantalla inicial durante 2 segundos y luego desplegar la lista de posibles uAPs.
- **Clase ListaDeAPs:**

Representa la pantalla en la cual el usuario selecciona con cual punto de acceso se desea conectar. Es una lista de tipo implícita, hereda sus características de la clase List.
- **Clase IngresarPassword:**

En esta pantalla se ingresa la clave de acceso en caso de ser necesaria, es un Displayable de tipo Form. El campo (TextField) donde se despliega la contraseña ingresada permanece oculto utilizando asteriscos.
- **Clase PantallaPrincipal:**

Esta es probablemente la clase más grande e importante de la aplicación. Representa la pantalla principal donde se escriben y reciben los mensajes comunes y desde donde el usuario accede a todas las opciones, por lo que en ella se crean gran parte de las restantes clases.
- **Clase HiloTx:**

Al crear esta clase se inicializa la conexión con el punto de acceso y es también la encargada de enviar mensajes hacia la red. Desde el punto de vista de Java es un hilo que permanece siempre corriendo en paralelo con el resto del programa.
- **Clase HiloRx:**

Esta clase es también un hilo de Java pero su función es la de escuchar permanentemente el canal RFCOMM, agrupar y procesar los mensajes recibidos desde la red.
- **Clase Destino:**

Contiene la información relevante sobre un destino visible para el usuario. Cada destino posible es una instancia de esta clase y el total se encuentra agrupado en un objeto de la clase ListaDeDestinos.

- **Clase ListaDeDestinos:**
Hereda sus características de la clase List, es una lista múltiple y almacena los destinos visibles como objetos de la clase Destino. Su contenido es modificado por mensajes de la red que utilizan el protocolo de comunicación Umaguma. Es accedida por el usuario cuando desea seleccionar destinos para los mensajes.
- **Clase DestinosFoto:**
Implementa la pantalla en la que se seleccionan los destinos para una foto tomada por el usuario. Extrae su información de la instancia de de ListaDeDestinos perteneciente a la pantalla principal.
- **Clase DestinosAudio:**
Análoga a la clase anterior pero para el caso de envió de grabaciones de audio realizadas por el usuario.
- **Clase DestinoSnF:**
En esta pantalla se introduce la dirección MAC del destino al que se desea enviar un mensaje de tipo Store and Forward.
- **Clase ResponderSnF:**
En esta pantalla se ingresa un nuevo mensaje a enviar para responder un mensaje Store and Forward recibido sin la necesidad de ingresar su dirección MAC.
- **Clase MensajeSnF:**
La instancias de esta clase representan mensajes Store and Forward recibidos, los cuales son desplegados cuando el usuario inicia sesión en la red. Estos mensajes son mostrados como pantallas emergentes (subclase de Alert).
- **Clase MensajeSistema:**
Cuando se recibe un mensaje del sistema se crea una instancia de esta clase y se despliega en pantalla, es una pantalla emergente por lo que hereda de Alert.
- **Clase FotoRecibida:**
Pantalla de tipo Form en la que se despliega una imagen recibida y se muestra al usuario junto con la información del remitente.

- **Clase FotoTomada:**

En esta pantalla -de tipo Form- se muestra la foto tomada por el usuario y se ofrece la opción de enviarla a otro usuario o de regresar a la pantalla principal.

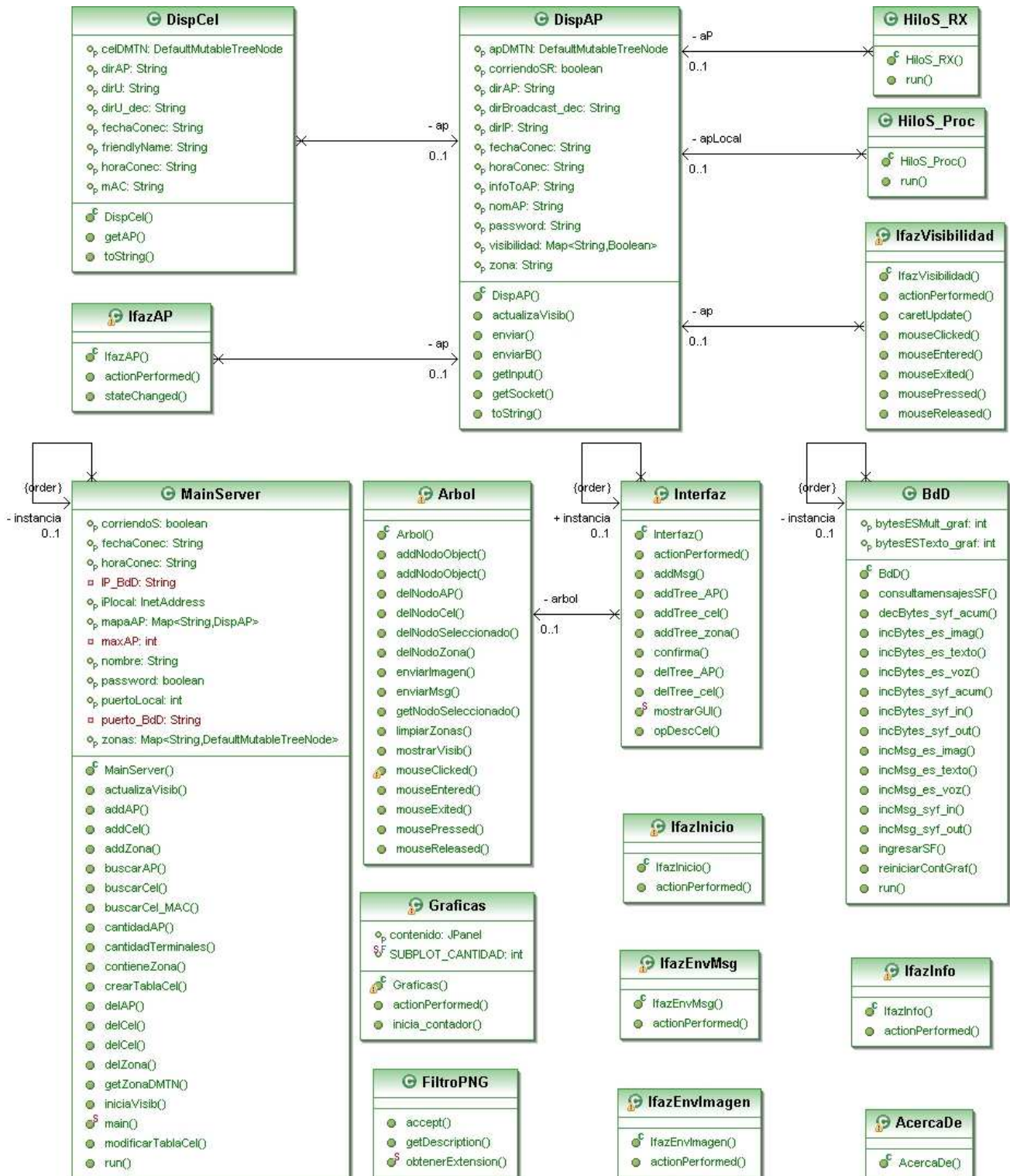
- **Clase Camara:**

Esta clase es la encargada de desplegar al usuario la cámara fotográfica para permitirle enviar imágenes. También toma la foto y a continuación crea la lista de destinos (clase DestinosFoto).

- **Clase PantallaAudio:**

En esta clase se manejan los dispositivos responsables de los mensajes de audio, tanto en la grabación como en la reproducción. Contiene dos posibles pantallas mediante las cuales el usuario hace uso de esta funcionalidad.

2.4.2.3 uServer



- **Clase MainServer:**

Es la clase principal del software uServer y hereda de la clase Thread. El constructor se encarga de crear la interfaz que se muestra al inicio del programa (IfazInicio), utilizada para configurar los parámetros del Servidor y de la RED, crear una instancia de BdD y ejecutar el thread correspondiente, crear y mostrar la interfaz principal del programa (Interfaz) y crear las colecciones utilizadas para almacenar a los objetos que representaran a los diferentes elementos de la red (Terminales, AP y Zonas).

El método run() deja al servidor listo para recibir conexiones y entra en un bucle en el que se bloquea a la espera conexiones de nuevos uAPs. El resto de los métodos tienen como finalidad obtener y modificar datos de las colecciones que almacenan a los objetos representativos de los distintos elementos de la red y propiedades de dichos objetos.

- **Clase DispAP:**

Esta clase es instanciada desde MainServer cuando se conecta un nuevo uAP, las instancias se utilizan para representar a los puntos de acceso que forman parte de la red.

El constructor se encarga de definir los distintos parámetros de un objeto DispAP a partir del socket de la conexión y de la dirección asignada a dicho AP automáticamente en la clase MainServer. El resto de los métodos se utilizan para obtener o setear alguno de los parámetros y para enviar un String o un array de bytes desde el uServer al uAP representado por la instancia.

- **Clase HiloS_RX:**

Es la clase instanciada por MainServer luego de crear un objeto DispAP. También extiende de la clase Thread, ya que la función principal de los objetos de esta clase es recibir datos desde los uAPs, y por lo tanto deben ejecutarse en un hilo en paralelo al resto del programa.

El método run() se encarga de definir algunos parámetros que caracterizaran a dicho punto de acceso (mediante la interfaz IfazAP), como son el nombre y la Zona en la que se encuentra. También se encarga de terminar de establecer la conexión con el uAP, para luego ingresar en un bucle el cual se reciben los datos transmitidos desde el AP correspondiente. Además, cuenta con un método (desconectar()) para llevar a cabo la desconexión del AP cuando sea necesario.

- **Clase HiloS_Proc:**

Esta clase es instanciada desde HiloS_RX cada vez que se recibe una trama. La función principal es la de procesar dicha trama, y dado que nos interesa que el uServer pueda seguir recibiendo datos mientras se procesan los ya recibidos, la clase HiloS_Proc también hereda de la clase Thread.

El método run() es el que se encarga de procesar las tramas a partir del formato de las mismas y de la utilización del resto de los métodos presentes en la clase.

- **Clase DispCel:**

Similar a la clase DispAP, esta clase se utiliza para representar a los Terminales conectados a la red. El constructor se encarga de definir los parámetros de dicho elemento a partir de una trama especial que le notifica de la nueva conexión. Es además, durante el procesamiento de ésta trama en la clase HiloS_Proc, que se crea una instancia de DispCel. Además del constructor, la clase posee una variedad de métodos para obtener y setear algunos de los parámetros del Terminal.

- **Clase BdD:**

Esta clase se encarga de negociar con la Base de Datos para ingresar, obtener y eliminar registros de las tablas de interés para el Servidor. Dado que una de las funciones será reportar estadísticas periódicamente, y se desea que esta funcionalidad no afecte al resto de las funciones del uServer, se decidió ejecutar la instancia de la clase en cuestión en paralelo con el resto del programa, por lo que la clase BdD también extenderá de la clase Thread.

La principal función del constructor es iniciar los contadores utilizados para llevar las estadísticas por el uServer. El método run() se encarga de enviar estos datos hacia la Base de Datos cada 15 minutos. Además del método utilizado para ingresar los registros de estadísticas y los métodos utilizados para incrementar y decrementar los contadores, la clase BdD tendrá también tres métodos para ingresar, obtener y eliminar registros de la tabla Store and Forward y dos métodos para obtener los datos a graficar en la Interfaz principal del programa.

- **Clase Interfaz:**

Es la interfaz principal del programa, contiene un árbol en el cual se muestran todos los elementos conectados a la red, un menú donde se muestran las opciones disponibles para cada uno de los elementos, dos gráficas donde se ilustra el tráfico de bytes correspondiente a mensajes de texto y mensajes multimedia (imágenes y

archivos de voz) y una consola para desplegar eventos como conexión de nuevos usuarios, envíos de mensajes, etc.

- **Clase Arbol:**

Esta clase representa al árbol utilizado para desplegar los elementos conectados a la red. Además del constructor, contiene métodos para agregar y quitar nodos del árbol y para realizar acciones sobre ellos, como son el envío de texto o imágenes desde el servidor.

- **Clase Graficas:**

Al igual que la clase Arbol, esta clase se instancia una sola vez en el constructor de Interfaz y se utiliza para crear las gráficas que aparecen en dicha pantalla. Obtiene los datos necesarios para realizar las gráficas de contadores de la clase BdD.

- **Clase IfazInicio:**

Como ya se mencionó, esta clase implementa la interfaz con que se inicia el uServer, y permite la configuración de parámetros del uServer y de la red.

- **Clase IfazAP:**

Representa la pantalla que se muestra cada vez que se conecta un uAP. Permite configurar el nombre del nuevo nodo, la zona a la que pertenecerá y la contraseña (en caso de que se decida utilizar).

- **Clase IfazEnvMsg:**

Esta clase se instancia cada vez que se selecciona la opción de enviar un mensaje de texto en el menú de la interfaz principal. Su función principal es capturar el texto introducido por el usuario y enviarlo al destino seleccionado, el cual -como se verá más adelante- puede ser un Terminal, todos los Terminales de un uAP, todos los Terminales dentro de una Zona o todos los Terminales de la red.

- **Clase IfazEnvImagen:**

Es similar a la clase descrita en el punto anterior, solo que en este caso la instancia se utiliza para enviar una imagen previamente archivada en el uServer. Para esto, implementa un explorador de archivos que permite buscar la imagen a enviar. El explorador (FileChooser) utiliza la clase FiltroPNG para asegurarse que solo se muestren y puedan elegir imágenes con formato “png”, dado que este formato es el soportado por el software Umaguma que corre en los Terminales.

- **Clase FiltroPNG:**

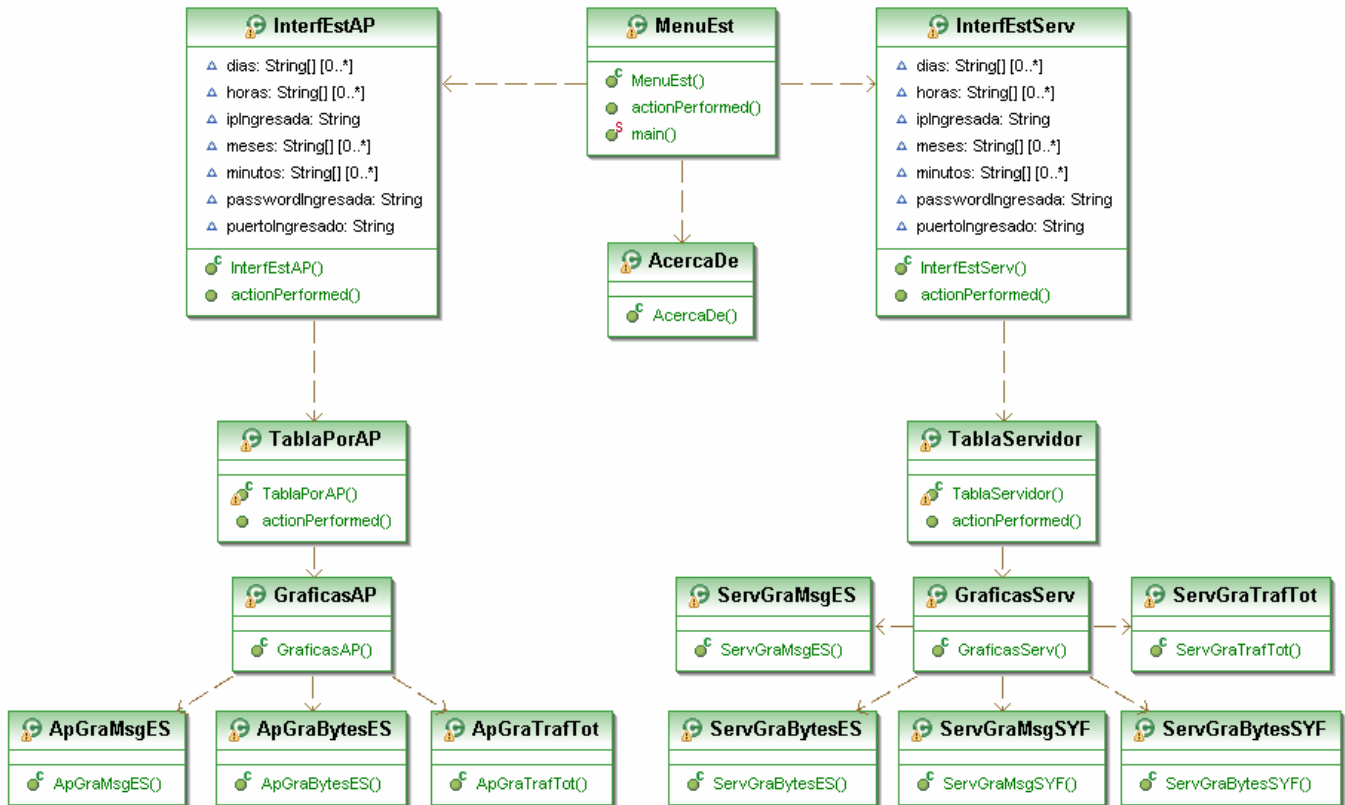
Como se mencionó en la descripción de IfazEnvImagen, esta clase se utiliza para restringir los archivos que se pueden visualizar en el explorador utilizado para buscar la imagen que se desea enviar. Filtra de tal manera que solo sean visibles los directorios y las imágenes con extensión “png”.
- **Clase IfazInfo:**

Esta clase es instanciada cuando se selecciona la opción INFO en el menú de la clase principal. Es una interfaz de solo lectura cuyo fin es informar al usuario de las propiedades más importantes de cada uno de los elementos de la red. Dado que la información a mostrar dependerá del tipo de elemento seleccionado (Terminal, Punto de Acceso, Zona, red), el comportamiento del constructor de esta clase también lo hará.
- **Clase IfazVisibilidad:**

La visibilidad es una propiedad de los uAP, que, como se verá en capítulos posteriores, permite restringir la comunicación entre dos o más puntos de acceso. Para esto, cuando se selecciona un uAP en la interfaz principal aparecerá la opción de configurar la visibilidad en la barra de menú, lo que lleva a la creación de una instancia de IfazVisibilidad. Esta Interfaz permite al usuario configurar la visibilidad del uAP en cuestión, y se encarga de realizar las modificaciones necesarias para aplicar dicha configuración.
- **Clase AcercaDe:**

Esta clase representa la interfaz grafica utilizada para desplegar información relativa al proyecto Umaguma. Se instancia cada vez que se selecciona la opción “Acerca De...” del menú de la interfaz principal.

2.4.2.4 uDataBase



- Clase MenuEst:**
 Es la clase principal de la aplicación. En ella se construye la pantalla inicial y se definen las variables que contendrán la información para realizar la conexión con la base de datos. Además, a partir de ella se crearán las posibles instancias que controlaran el flujo del programa.
- Clase AcercaDe:**
 Esta clase representa la interfaz grafica utilizada para desplegar información relativa al proyecto Umaguma.
- Clase InterfEstAP:**
 En esta clase se realiza la interfaz gráfica que solicitará al usuario que ingrese el período para el cual desea analizar estadísticas, así como también la dirección del AP a analizar o la opción de analizarlos a todos. Los datos ingresados los guardará

en variables que luego utilizará para crear las instancias a las clases que entregarán los resultados finales al usuario.

- **Clase TablaPorAP:**

Esta clase se encarga de todo lo que respecta a la obtención de información de la base de datos. Establece conexiones temporales con la base, realiza las consultas necesarias y recopila la información obtenida. A partir de esta información, crea una interfaz gráfica donde construye una tabla en la cual le presenta al usuario los resultados obtenidos. Si el usuario lo requiere, esta clase además creará las instancias a las clases encargadas de representar gráficamente los resultados.

- **Clase GraficasAP:**

En esta clase se crea la interfaz que reúne todas las gráficas presentadas al usuario de los resultados obtenidos en el análisis estadístico.

- **Clase ApGraMsgES:**

Esta clase realiza, a partir de los datos recopilados, la gráfica donde se muestra la cantidad de mensajes E/S en función del tiempo para el AP seleccionado, distinguiendo en tres trazos diferentes los tres tipos de mensajes (texto, imágenes y voz).

- **Clase ApGraBytesES:**

Análoga a la clase anterior, con la diferencia de que grafica los bytes de E/S en función del tiempo para el AP seleccionado.

- **Clase ApGraTrafTot:**

Esta clase crea el gráfico que representa el tráfico total (en bytes) cursado por el uAP en función del tiempo, distinguiendo aquel tráfico que se cursó entre Terminales bajo el mismo uAP y el que fue enrutado hacia el Servidor.

- **Clase InterfEstServ:**

La función de esta clase es análoga a la de la clase InterfEstAP pero aplicada al Servidor.

- **Clase TablaServidor:**
Análoga a la clase TablaPorAP pero aplicada al Servidor. Los parámetros que se analizan estadísticamente en APs y Servidor son diferentes por lo tanto aunque las funciones de estas clases son similares, la estructura y los resultados no lo son.
- **Clase GraficasServ:**
Análogo a lo expuesto en la clase GraficasAP pero aplicado al Servidor.
- **Clase ServGraMsgES:**
Análogo a lo expuesto en la clase ApGraMsgES pero aplicado al Servidor.
- **Clase ServGraBytesES:**
Análogo a lo expuesto en la clase ServGraBytesES pero aplicado al Servidor.
- **Clase ServGraMsgSYF:**
Esta clase se encarga de la creación del gráfico que reporta la cantidad de mensajes de Store and Forward en función del tiempo, distinguiendo en diferentes trazos los mensajes que se enviaron a almacenar a la base de datos, los que se entregaron a los terminales desde la base de datos, y lo que quedaron acumulados en la base de datos.
- **Clase ServGraBytesSYF:**
Análogo a lo explicado en el punto anterior, con la diferencia de que es aplicado a la cantidad de bytes en vez de la cantidad de mensajes.
- **ServGraTrafTot:**
Es la clase que construye la gráfica en donde se representa en dos trazos diferentes el tráfico total entrante y saliente del servidor en función del tiempo.

2.4.3 Librerías Utilizadas

Como ya se mencionó anteriormente, para llevar a la práctica las aplicaciones de los nodos se utilizó el lenguaje de programación Java en sus distintas variantes, sin embargo en muchos casos se necesita acceder a prestaciones que no están incluidas en el conjunto básico de clases contenidas en cada versión. Es por esto que se debe utilizar distintos paquetes de librerías, las cuales incluyen clases que son capaces de expandir las

funcionalidades del programa, sobretodo en lo que refiere a interacción con dispositivos externos al programa. A continuación se mencionarán las librerías utilizadas en cada uno de los nodos, describiendo brevemente su función dentro del programa. Cabe destacar que todas las librerías que se utilizaron son del tipo “Open Source”, por lo que se dispone de licencia libre para su uso.

2.4.3.1 uAP

- **Bluecove:**
Maneja todo lo relativo a las comunicaciones mediante Bluetooth, desde la conexión con los terminales, la transmisión de información hasta la desconexión. Para esto maneja los protocolos SDAP, L2CAP, RFCOMM y OBEX. También se encuentra presente en el nodo uCel.
- **JFreeChart:**
Es utilizado para construir los gráficos estadísticos que reportan continuamente el tráfico circulante a través de él. También esta presente en los nodos uServer y uDataBase.
- **JCommon:**
Esta librería es utilizada auxiliariamente por JFreeChart para realizar los gráficos estadísticos. También esta presente en los nodos uServer y uDataBase.

2.4.3.2 uCel

- **Wireless Toolkit (JSR 82):**
Este paquete incluye lo necesario para desarrollar programas utilizando J2ME, desde las clases correspondientes hasta las herramientas para realizar la pre-verificación, simulación y compilación.
- **Bluecove:**
Maneja todo lo relativo a las comunicaciones mediante Bluetooth, desde la búsqueda de dispositivos, la conexión con los puntos de acceso, la transmisión de información hasta la desconexión. Para esto maneja los protocolos SDAP, L2CAP, RFCOMM y OBEX. También se encuentra presente en el nodo uAP.

- **MMAPI (JSR 135):**

Es la API encargada de manejar todos los dispositivos multimedia del teléfono celular, en nuestro caso la cámara de fotos, así como la grabación y reproducción de audio.

2.4.3.3 uServer

- **JFreeChart:**

Es utilizado desplegar de los gráficos estadísticos que reportan continuamente el tráfico circulante a través de él. También esta presente en los nodos uAP y uDataBase.

- **JCommon:**

Esta librería es utilizada auxiliariamente por JFreeChart para realizar los gráficos estadísticos. También esta presente en los nodos uAP y uDataBase.

2.4.3.4 uDataBase

- **PostgreSQL JDBC:**

Esta librería es utilizada para establecer conexiones con la base de datos PostgreSQL, así como también para realizar un embebido de comandos SQL dentro del código Java y de esta manera realizar todo el manejo de consultas a la base de datos, incluyendo la inserción y borrado de datos.

- **JFreeChart:**

Es utilizado para desplegar de los gráficos estadísticos de los distintos reportes que se pueden efectuar. También esta presente en los nodos uAP y uDataBase.

- **JCommon:**

Esta librería es utilizada auxiliariamente por JFreeChart para realizar los gráficos estadísticos. También esta presente en los nodos uAP y uDataBase.

2.5. Requerimientos de la aplicación

En esta sección se describirá -dentro de lo posible- los distintos requerimientos que se deben tener en cuenta a la hora de montar el sistema, concretamente en los equipos (computadoras) sobre los que se ejecutarán los distintos nodos.

Si bien las aplicaciones que implementan cada uno de los nodos de la red presentan diferencias en sus requerimientos de funcionamiento (respondiendo a las necesidades de software y hardware de cada una), en términos generales se encontraron necesidades básicas similares en cuanto al rendimiento y capacidad de procesamiento para el caso de los nodos que integran la parte interna de la red (Servidor, Base de Datos y Puntos de Acceso). Actualmente no se dispone de una medida exacta de los requerimientos mínimos de cada aplicación, no obstante se encontró que en todos los casos las aplicaciones funcionan con muy buen desempeño en los equipos que fueron utilizados por el grupo para el desarrollo del proyecto. A continuación se listan estos equipos a modo de guía:

- HP 550, procesador Core 2 Duo T5470 y 3 GB de memoria RAM
- Toshiba Satellite Pro, procesador AMD Turion X2 y 2 GB de memoria RAM
- Acer TravelMate 4220, procesador Centrino Duo y 1Gb de memoria RAM
- Dell Inspiration 1525, procesador Core 2 Duo T5800 y 2 GB de memoria RAM

En resumen se puede decir que todas las aplicaciones desarrolladas para la red interna presentan requerimientos de funcionamiento similares entre si y fácilmente alcanzables por equipos como los que usualmente encontramos y disponemos, incluso a un nivel domiciliario. Sobre estos equipos se debe correr algún sistema operativo que sea capaz de ejecutar un intérprete Java así como manejar las interfaces de red necesarias (ya sea Ethernet, Wifi o Bluetooth) según el caso. Los sistemas operativos utilizados para las pruebas fueron:

Ubuntu 8.10 LTS Desktop Edition
Ubuntu 8.04 LTS Desktop Edition
Windows XP Professional, Service Pack 2
Windows Vista Home Basic Edition

Para todos estos casos las aplicaciones mostraron muy buen desempeño y no se presentaron problemas de compatibilidad o conflictos con los dispositivos de hardware asociados.

Adicionalmente a lo anterior, las computadoras utilizadas deben poseer -al menos- una interfaz de red para la interconexión con el resto del sistema, pudiendo ser, por

ejemplo, un puerto Ethernet (cableado) o una interfaz Wifi (inalámbrica) dependiendo de cómo se lleve a cabo el diseño de la LAN IP. En el caso de los puntos de acceso el equipo debe disponer de un puerto USB en el que se conectará un dispositivo Bluetooth que manejará la conexión con los Terminales. También se deben tener instalados los drivers correspondientes a cada puerto según el sistema operativo que se utilice.

El en caso de los terminales de usuario se tienen requerimientos de funcionamiento más simples, básicamente este software se debe correr sobre un dispositivo móvil (típicamente un teléfono celular) que disponga de dos cosas: 1. Debe ser capaz de ejecutar programas escritos en J2ME (utilizando CLDC v1.1 y MIDP v2.0). 2. Debe manejar comunicaciones mediante Bluetooth. Actualmente estos dos puntos son -en general- cubiertos por la mayoría de los teléfonos celulares disponibles en el mercado, particularmente se listan los terminales utilizados por el grupo para las pruebas:

- Sony Ericsson W810
- Sony Ericsson W300
- Motorola L6i
- Nokia 5200
- Nokia 6131
- Nokia E51

En general la aplicación (uCel) funcionó sin inconvenientes en todos los casos de prueba, destacando únicamente un problema de compatibilidad en la reproducción de audio para los casos del Sony Ericsson W810 y el Nokia E51. Estos dispositivos no permitieron reproducir los mensajes de voz recibidos, admitiendo sí la grabación. Como último requerimiento a destacar, los teléfonos celulares utilizados deben disponer de cámara fotográfica en tanto se quiera transmitir imágenes a través de la red, de no poseerla se puede utilizar todo el resto de los servicios del sistema.

2.6. Casos de Uso

Resta definir -por ahora sin entrar en muchos detalles técnicos- los casos de uso típicos que tiene el proyecto Umaguma, con esto nos referimos a ejemplos de situaciones en las cuales se podría utilizar el sistema como solución a un nivel empresarial. Desde un punto de vista más formal (UML), se estará definiendo un caso de uso en formato breve e informal y en estilo de escritura esencial, en el cual se describe un escenario principal, enfatizando más que nada en los posibles usos del sistema y no tanto en los procesos, de los cuales se hablará en un capítulo posterior (Capítulo 5), donde se explicará más

detalladamente como ocurren estos procesos durante el uso del sistema para distintos ejemplos.

Como primer caso de uso se puede citar el de una empresa o cadena de locales que disponga de varias sucursales y que necesite mantener comunicados a sus empleados, durante el horario de trabajo, mediante un sistema de terminales móviles. En esta situación la empresa podría instalar un punto de acceso (uAP) en cada uno de las sucursales y repartir entre sus empleados teléfonos celulares con el software correspondiente instalado. Los distintos uAP se conectarían entre si mediante Internet (o bien una red IP privada) a través de un servidor instalado en la casa central desde donde se comandarían las comunicaciones de toda la empresa. De esta manera se pueden restringir las zonas de visibilidad o los horarios de uso según las políticas de la empresa lo requieran. Durante el horario de trabajo los empleados podrán comunicarse entre si (cuando las restricciones así lo permitan) mediante mensajes de texto, o -para situaciones más complejas- se podría utilizar una grabación de voz o una captura fotográfica. Otra posibilidad es que el directorio de la empresa decida realizar difusión de mensajes a todos sus empleados (o a determinadas zonas), utilizando para esto los servicios de broadcasting que provee el servidor.

Esta propuesta, a diferencia de un sistema de telefonía convencional, utiliza solamente equipos e interconexiones privadas (salvo únicamente la interconexión de nodos mediante Internet, en caso de usarse), lo cual hace que el sistema no presente un costo proporcional a su uso, además de brindar seguridad frente a espionajes de actores externos al sistema. Además de esto la empresa podría realizar un seguimiento estadístico de las comunicaciones internas analizando como fue el tráfico entre los distintos locales.

Otro posible uso que cabe dentro de un esquema como el anterior es el de realizar, mediante el seguimiento de las conexiones de los empleados, un control de jornadas laborales al estilo del tradicional “marcado de tarjeta”. De esta manera se podría comprobar cual fue el horario laboral efectivo observando los logs de cada uno de los puntos de acceso para cada usuario. El usuario debería llegar a su puesto e iniciar sesión en la red, permaneciendo en ese estado (en el que utiliza el sistema del modo convencional) hasta el final de su jornada en el que cierra la sesión. De esta manera queda registrado en el sistema la cantidad exactas de horas trabajadas por el empleado.

Para los ejemplos mencionados anteriormente se presentan como actores principales el sistema (Umaguma) y los usuarios (empleados), por ser quienes interactúan directamente con la situación. Como actor de apoyo se puede nombrar al administrador de la red, que interviene en la escena pero de una manera secundaria durante el uso normal. Y como actor pasivo tenemos al directorio de la empresa que, si bien esta claramente interesado es el buen funcionamiento del sistema, no participa activamente en su uso.

Se tiene también, como caso de uso posible, el de la distribución selectiva de contenido dependiendo de las zonas en la que se encuentre el usuario. Se pone como ejemplo un museo en el cual cada habitación representa una temática distinta en las exposiciones. El usuario iniciaría sesión en distintos puntos de acceso cada vez que ingrese a una nueva habitación, entre tanto el sistema le estaría enviando constantemente información sobre la temática a la cual hace referencia la habitación, dando incluso la posibilidad de que el usuario solicite cierta información más detallada sobre algún tema en particular enviando mensajes especiales a una dirección preestablecida, que serán recibidos por el administrador de la red.

Una vez más se observa que los actores principales para este caso son básicamente el sistema (Umaguma) y los usuarios (en este caso visitantes del museo), como actor de apoyo también se tiene al administrador de la red, el cual ingresa en escena en situaciones particulares, pero se mantiene normalmente al margen. En este caso el actor pasivo a destacar sería la autoridad del museo, que tiene interés en el buen desenlace de la situación pero no participa de ella.

Con esto quedaría completado el planteo teórico y funcional del proyecto. Los siguientes pasos son dar comienzo al desarrollo práctico del sistema. En el capítulo 4 se detallarán las características de la implementación y las dificultades que fueron surgiendo durante el proceso.

Capítulo 3: Protocolo de Comunicaciones Umaguma

3.1. Introducción

Como fue mencionado anteriormente uno de los primeros objetivos alcanzados por los integrantes del proyecto, fue el intercambio de datos entre teléfonos móviles. Los cuales eran enrutados de un Terminal a otro por un nodo uAP.

El paso siguiente a este fue lograr el enrutamiento de mensajes entre varios celulares los cuales se encontraban conectados a una misma PC. Para lograr esto fue necesario en primer lugar crear una estructura de direcciones las cuales pudiesen identificar de manera única a cada uno de los usuarios de la red.

Como se detallará más adelante en este capítulo, fue creada una estructura de direcciones la cual supliera de la mejor manera posible las necesidades de la red. Esta estructura permitiría identificar de manera única a los usuarios de la red, ya que la misma estaría compuesta por la dirección de un cierto nodo uAP concatenada a la dirección del terminal debajo de ese nodo.

Habiendo elegido una apropiada estructura de direcciones, el proyecto se topó con otro problema. Cuando un Terminal quisiera enviar un mensaje tendría que especificar la dirección del celular de destino, así como también su dirección propia en el mensaje. Para que de esta manera el PC cuando recibiera este mensaje lo pudiese enrutar hacia el destino correcto.

Fue en este momento en el cual se resolvió crear una trama común de mensajes para todos los agentes de la red, permitiendo así que estos pudiesen identificar parámetros como la dirección de destino, dirección de origen, entre otros y de esta manera poder tomar la acción correcta al procesar el mensaje.

A medida que el proyecto continuaba creciendo, se fueron desarrollando diferentes características de la red, lo que llevo a la necesidad del intercambio de información de control entre los diferentes agentes de la red.

Debido a esto se fueron utilizando las direcciones más altas del rango. Haciendo que cada mensaje con alguna de esas direcciones de destino transportara información de control relevante para alguno de los componentes de la red Umaguma.

A continuación se pasará a detallar todo el protocolo Umaguma desarrollado para el correcto funcionamiento de red.

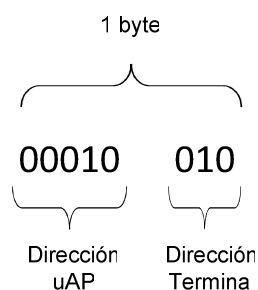
3.2. Asignación de Direcciones

Como se dijo durante el desarrollo del funcionamiento lógico del nodo uAP, el protocolo Bluetooth tiene como limitación permitir hasta siete celulares conectados hacia un mismo dispositivo.

Esta limitación en la máxima cantidad de usuarios por punto de acceso, fue un dato determinante para designar la cantidad de bits a usar, al momento de identificar un usuario debajo de un cierto nodo uAP.

Teniendo en cuenta esto, se designó un byte para determinar de manera global a cualquier usuario de toda la red. Utilizando los tres bits menos significativos para identificar a los usuarios que se encuentran dentro del área de cobertura de cierto nodo uAP, y los cinco bits más significativos para identificar el nodo uAP bajo el cual se encuentra el usuario.

Un ejemplo de esta estructura de direcciones globales sería el siguiente:



El proceso de asignación de direcciones se lleva a cabo en dos etapas bien definidas. En primer lugar cuando un nodo uAP inicia, si esta seteado para trabajar sin uServer este asume por defecto que su dirección es 00000, obteniéndose así los primeros cinco bits del byte global de direcciones.

En el caso en que este se encuentre seteado para trabajar con uServer, este último le asigna una dirección de cinco bits, luego de la conexión. La cual por supuesto es única en toda la red.

Luego de finalizada esta etapa se inicializa la interfaz Bluetooth, y cuando los celulares empiezan a iniciar las conexiones al uAP, este comienza a asignarle direcciones a los

mismos en un rango que va de 000 a 110, cubriendo de esta manera la máxima cantidad posible de conexiones por uAP. La dirección 111 es utilizada como broadcast dentro del área de cobertura del uAP.

Por lo tanto cuando un terminal que se encuentra en la red envía un mensaje, debe especificar en el encabezado la dirección global de destino y origen.

La dirección global de destino es el resultado de la concatenación de la dirección del uAP de destino (5 bits) junto con la dirección del Terminal debajo de ese uAP (3 bits). La dirección global de origen se genera de forma análoga, pero a partir de los datos del remitente del mensaje.

La dinámica de conexiones y desconexiones de terminales con el uAP, se realizan de manera que al desconectarse un celular esta dirección es liberada, y cuando un Terminal se conecta al uAP, se le asigna la dirección más baja disponible.

La figura a continuación muestra un ejemplo de la asignación de direcciones en la red Umaguma.

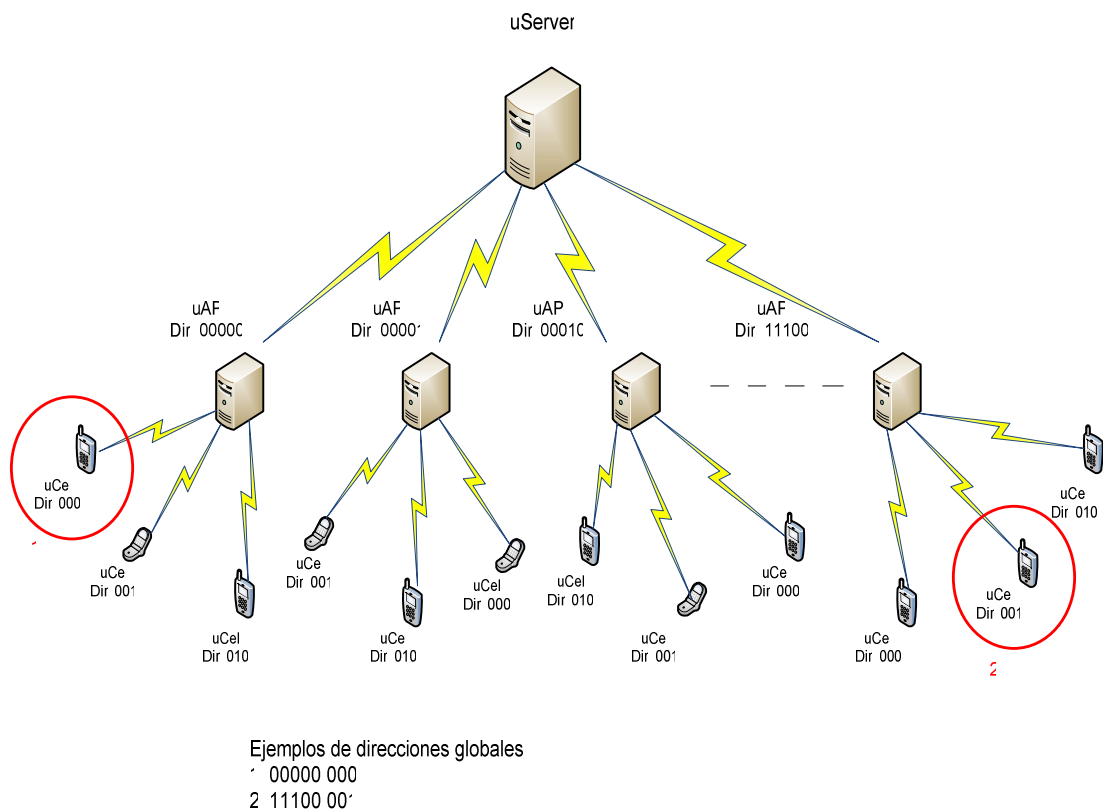


Figura 3.1 – Asignación de direcciones globales

3.3. Estructura y Entramado

3.3.1 Introducción

Luego de haber diseñado la estructura de las direcciones y el proceso de asignación de las mismas, se prosiguió por crear una estructura de trama para el intercambio de mensajes entre los diferentes componentes de la red.

La primera consideración a tener en cuenta para la creación de la estructura de trama, es que todo el software de los nodos sería creado para trabajar a nivel de la capa de aplicación. Esto llevó a crear el encabezado como una cadena de caracteres la cual luego iba a ser concatenada al mensaje para finalmente ser enviada al destino.

Esto también trajo acompañado que las direcciones que se utilizarían en el encabezado de los mensajes serían representadas como números que irían desde el 000 al 255, lo cual permite representar todo el rango de números decimales posibles a escribir con un byte.

Ya teniendo armada la manera a estructurar la trama, se procedió a diseñar como tratar este paquete de información entre la punta que envía el mensaje y la adyacente que lo recibe. En primer lugar, se crea la cadena de caracteres la cual contiene información relevante para el correcto enrutamiento del mensaje (esta estructura será explicada más adelante), y luego se concatena con el mensaje introducido por el usuario.

Como ya se ha dicho en capítulos anteriores, los mensajes que maneja la red se pueden dividir básicamente en dos grandes grupos: Mensajes de Datos y Mensajes de Control. A su vez dentro de los mensajes de datos, tenemos dos subgrupos, los mensajes de texto y los mensajes multimedia, los cuales representan los servicios de la red.

En el caso de los mensajes de texto, la cadena de caracteres introducida por el usuario es concatenada detrás de la cadena de caracteres del encabezado. Luego este mensaje es convertido a bytes para ser enviado hacia la punta adyacente. En su recepción se realiza el proceso inverso para analizar el mismo y determinar el destino del mensaje.

En el caso de los mensajes multimedia, el encabezado es construido como una cadena de caracteres la cual luego es convertida a bytes, para ser concatenada delante del mensaje del usuario, previo al envío del mismo hacia el nodo adyacente.

En su recepción se convierte la cadena de bytes a una cadena de caracteres, y se analiza el encabezado. Cuando se detecta que es un mensaje multimedia el nodo

adyacente solo manipula el encabezado para encontrar el destinatario del mensaje y posteriormente enviar la cadena de bytes hacia el componente de la red correspondiente.

A continuación, se explicará la estructura de trama para los mensajes utilizados por la red Umaguma.

3.3.2 Mensajes de Datos

Este grupo de mensajes son los encargados de representar a los usuarios, los servicios provistos por la red Umaguma. Los mismos transportan el contenido de importancia de la red, y cada nodo de ésta fue diseñado para tratar de la manera más dinámica posible a los mismos.

El primer gran objetivo planteado, fue poder crear una red la cual soportara el intercambio de mensajes de texto entre todos sus usuarios. Dicho objetivo fue realizado obteniendo resultados más que satisfactorios. Luego de haber desplegado el primer prototipo de la red Umaguma, se procedió a construir el soporte necesario para el transporte de la mensajería multimedia, otro de los grandes objetivos propuestos por los integrantes del proyecto.

En una primera instancia se propuso el envío de mensajes multimedia de imágenes, lo cual trajo acompañado grandes desafíos para los integrantes del proyecto, que fueron superados de manera exitosa como se verá en el siguiente capítulo.

Luego de haber realizado pruebas de la red Umaguma con soporte para el envío de texto e imágenes, y haber obtenido muy buenos resultados, se procedió a desarrollar el soporte de red necesario para el envío de mensajes de audio (objetivo que también fue alcanzado durante el desarrollo del proyecto).

Para que la red Umaguma pudiera dar un correcto soporte a estos servicios, fue necesario tratar cada uno de estos mensajes de manera diferente ya que el contenido de los mismos era distinto. Esto llevo a tener que construir diversos tipos de tramas según el mensaje.

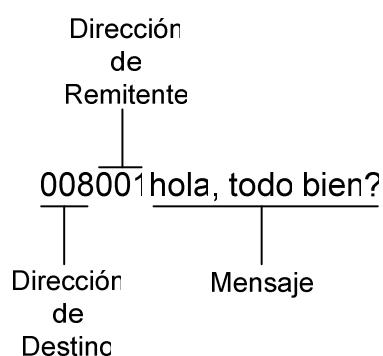
En la siguiente sección se describirá las diferentes estructuras de tramas utilizadas.

3.3.2.1 Mensajes de Texto

Estos mensajes surgen de la concatenación de dos cadenas de caracteres. La primera es utilizada como encabezado, la cual contiene datos necesarios para el correcto

procesamiento del mensaje como dirección de destino y origen. La segunda cadena representa el mensaje enviado por el usuario.

La figura que se muestra a continuación muestra un ejemplo de los mensajes de texto que son manejados por la red Umaguma.



Como se puede ver en esta figura, los mensajes de texto están compuestos por tres partes, bien determinadas.

En primer lugar se tiene el encabezado el cual está compuesto por dos partes necesarias para el correcto pasaje del mensaje por la red Umaguma.

Estas partes son las siguientes:

- **Dirección de Destino:** los primeros 3 caracteres del mensaje representan la dirección de destino del mensaje. Como se puede ver, la misma es representada por un número entero que va desde 000 al 255. Al leer estos tres caracteres, y constatar que la dirección no es ningún mensaje especial, convierte este número a un byte para realizar un correcto enrutamiento del mensaje
- **Dirección de Remitente:** los siguientes 3 caracteres a la dirección de destino, representan la dirección de origen del mensaje. Como se puede ver, la misma tiene el mismo formato que la dirección de destino. Dicha dirección de origen es utilizada por el Terminal destino de la de la comunicación, para informar el remitente del mensaje recibido.

Resumiendo, los mensajes de texto están compuestos por un encabezado de seis caracteres concatenado al mensaje del usuario el cual tiene un largo variable. Por lo tanto, se puede decir que el largo total del mensaje es variable, dependiendo este del mensaje escrito por el usuario.

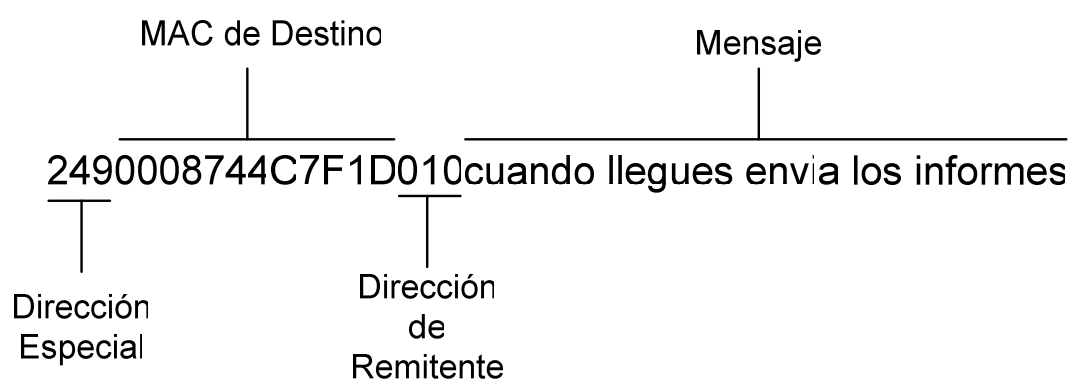
3.3.2.2 Mensajes de texto Store and Forward

Estos mensajes son una variante de los anteriores. Presentan como gran virtud la posibilidad de poder enviar mensajes de texto a usuarios que no se encuentran conectados a la red en ese momento.

Para el envío de este tipo de mensajes se reservó una dirección especial de la parte superior del rango. Esto es necesario ya que la estructura de trama utilizada para estos mensajes cambia con respecto a los mensajes descriptos anteriormente.

Otra particularidad de estos mensajes es que la estructura de trama utilizada en el proceso de envío del mensaje (desde que sale del terminal de un usuario hasta que llega al uServer), varía con respecto a la estructura de trama en la recepción del mensaje (desde que sale del uServer hacia el terminal que ingresa a la red).

El proceso de envío y recepción de mensajes Store and Forward será descripto en el Capítulo 5 (Ejemplos de uso). La figura a continuación muestra la estructura de trama utilizada para el envío de los mensajes Store and Forward.



La estructura de trama enviada por el terminal, no es modificada por el uAP y es enviada sin alteraciones al uServer. Como se puede ver en la figura, el mensaje enviado por el usuario también es el resultado de la concatenación de un encabezado de largo fijo con el mensaje escrito por el usuario.

A continuación se describen los componentes del encabezado:

- **Dirección Especial:** Los primeros tres caracteres del encabezado son la dirección de destino del mensaje. En este caso se utiliza el número 249, el cual pertenece a la parte alta del rango de direcciones. Cuando un nodo lee un 249 como dirección de

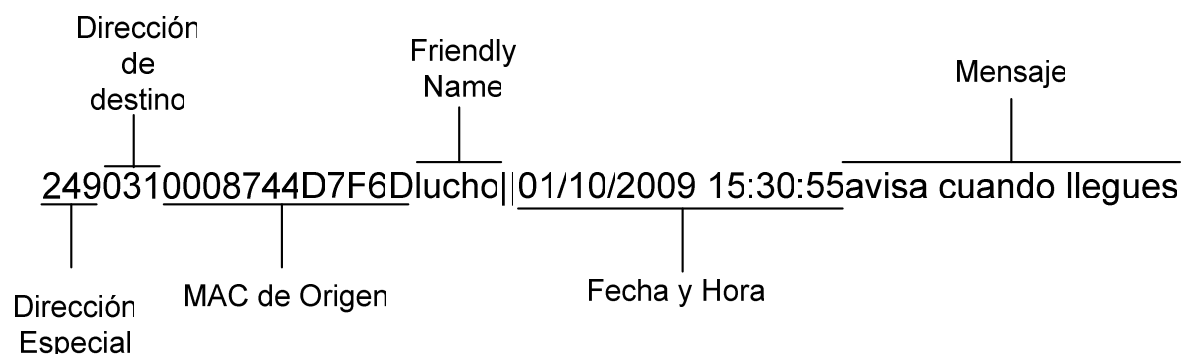
destino lo identifica como un mensaje Store and Forward, procesándolo de manera diferente a los mensajes de texto ordinarios.

- **MAC de Destino:** Los siguientes doce caracteres representan la dirección MAC del terminal de destino. Dicha dirección es única para cada dispositivo Bluetooth (más detalles sobre el protocolo Bluetooth serán dados en el Anexo correspondiente), lo que permite identificar de manera única a cualquier usuario de la red. La función de este parámetro es permitir identificar al terminal de destino del mensaje cuando este ingrese nuevamente a la red, ya que el mismo podría ingresar con otra dirección Umaguma.
- **Dirección de Remitente:** Los siguientes tres caracteres a la MAC de destino representan la dirección de remitente. Los mismos permiten identificar al usuario que envía el mensaje Store & Forward. Dicha dirección también es simbolizada como un número decimal de tres dígitos.

Concluyendo, la trama de los mensajes Store & Forward está compuesta por un encabezado de dieciocho caracteres concatenado al mensaje escrito por el usuario, el cual al igual que con los mensajes de texto ordinario, tiene un largo variable.

Luego de que un mensaje Store and Forward es enviado por un usuario, el mismo queda almacenado en la base de datos de la red a la espera de que el usuario destino ingrese nuevamente a la red. Cuando esto sucede, el uServer procede a enviar el mensaje hacia el usuario y en este proceso la estructura de trama utilizada es diferente a la anterior. La misma se pasa a describir a continuación.

La siguiente figura muestra la trama enviada por el uServer al uAP, cuando el primero detecta que el usuario destinatario del mensaje ingresa a la red. Esta identificación la realiza mediante la MAC de destino.



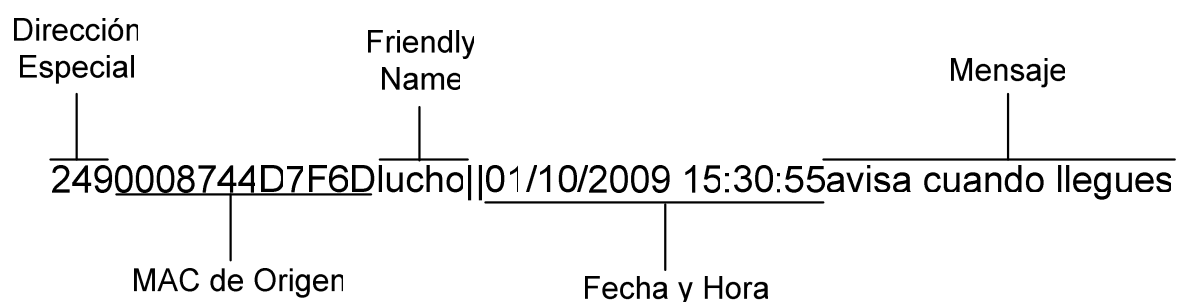
Como se puede ver, este mensaje contiene mayor información que el enviado por el usuario. El mismo cuenta con un encabezado con mas parámetros, los cuales se detallan a continuación.

- **Dirección Especial:** Los primeros tres caracteres del encabezado representan la dirección de destino 249, la cual funciona como identificador del mensaje Store and Forward.
- **Dirección de Destino:** Los siguientes tres caracteres representan la dirección del usuario destino. Esta es utilizada por el uAP, para realizar un correcto enrutamiento del mensaje.
- **MAC de Origen:** Los siguientes doce caracteres a la dirección de destino, representan la dirección MAC del terminal de origen. La misma es desplegada junto con el mensaje cuando este llega a destino.
- **Friendly Name:** Este parámetro, es uno de los obtenidos por la red Umaguma cuando un Terminal ingresa a la red, lo cual fue explicado en el capítulo 2 (Desarrollo de Red uAP). Es de gran importancia al momento de desplegar el mensaje en el Terminal de usuario, ya que el mismo es el utilizado por los usuarios para identificarse en la red de manera amigable, como lo indica su propio nombre.
- **Fecha y Hora:** Parámetro de diecinueve caracteres indicando la fecha y hora en la cual fue enviado el mensaje Store and Forward. Estos datos son desplegados al usuario junto con el mensaje.

Como se puede ver en la figura, dentro del encabezado existe una suerte de marcador indicado por dos caracteres, “||”. Este parámetro es utilizado por los componentes de la red para saber cuál es el largo del Friendly Name, ya que el mismo no tiene un largo definido

Concluyendo, se puede decir que el largo del encabezado es de treinta y nueve caracteres sumándole además el largo del Friendly Name. Concatenado a este encabezado se encuentra el mensaje enviado por el usuario.

La figura a continuación representa la trama enviada desde el uAP hacia el usuario destino. Como se puede ver, la misma sufre solamente una modificación con respecto a la trama anterior.



Esta estructura de trama no contiene la dirección de destino del mensaje, ya que la misma es utilizada únicamente por el uAP, por lo que luego de utilizada esta es quitada de la trama. En cambio, el resto de los parámetros son de vital importancia para el Terminal del usuario destino, por lo que se dejan intactos.

Esta trama tiene un largo de treinta y seis caracteres sumándole además el largo del friendly name. La misma es procesada por el software del Terminal de destino para luego desplegarle al usuario el mensaje Store and Forward recibido junto con los parámetros de origen ya descriptos.

3.3.2.3 Mensajes Multimedia

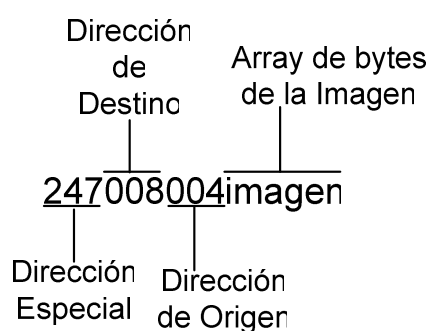
Los mensajes multimedia representan otra de las variantes en los servicios provistos por la red Umaguma. Dentro de estos se distinguen los mensajes multimedia de imágenes y los mensajes multimedia de audio.

A continuación se pasa a explicar la estructura de trama utilizada para el envío de ambos mensajes multimedia.

3.3.2.3.1 Mensajes Multimedia de Imágenes

Para el envío de estos mensajes fue reservada una dirección alta del rango, ya que los mismos tienen que ser procesados de una manera diferente por parte de los nodos de la red.

La figura que se presenta a continuación muestra la estructura de trama utilizada para el envío de imágenes.



Como se puede ver se tiene un encabezado con tres parámetros, los cuales se pasan a explicar a continuación:

- **Dirección Especial:** como se muestra en la figura los tres primeros caracteres del encabezado es la dirección 247, la cual se encuentra reservada para el envío de mensajes multimedia de imágenes. Esto hace que cuando un nodo recibe un mensaje, al leer estos tres caracteres, identifica al mensaje como un mensaje multimedia procesándolo de la manera adecuada.
- **Dirección de Destino:** los siguientes tres caracteres representan la dirección de destino del mensaje multimedia. Al transformar esta dirección decimal a un byte, se leen los tres bits menos significativos para determinar el destino del mensaje. En caso de que fuese "111", el mensaje es enviado a todos los celulares conectados a ese uAP.

- **Dirección de Origen:** los últimos tres caracteres del encabezado indican la dirección del usuario de origen, la cual es utilizada por el software del terminal de destino para indicarle al usuario el remitente del mensaje.

Por lo tanto se tiene un encabezado de largo fijo de nueve caracteres.

Al momento de enviar un mensaje multimedia, el software del terminal utiliza la cámara del mismo para obtener la imagen, para luego almacenarla como una cadena de bytes. La cual es previamente concatenada al encabezado antes del envío del mensaje. Esto lleva a que en el momento de armar el mensaje lo que se concatenen sean cadenas de bytes y no cadenas de caracteres como en los mensajes de texto.

El procedimiento de armado estos mensajes consiste en primer lugar obtener la cadena de bytes que representa a la imagen, para luego concatenarla detrás de una cadena de bytes la cual representa el encabezado. Este encabezado es creado como una cadena de caracteres que luego es transformado a un array de bytes.

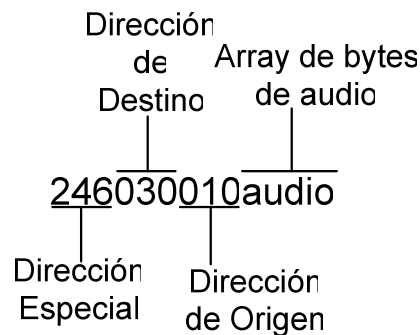
Cuando un mensaje llega a un nodo, el mismo recibe una cadena de bytes la cual es transformada a una cadena de caracteres para su procesamiento. Al leer los primeros tres caracteres de esta cadena y ver que se trata de un mensaje multimedia debido al 247, el nodo procede a leer los siguientes tres caracteres de la cadena, los cuales, como ya se vio representan la dirección de destino del mensaje. Luego de determinar el destino correspondiente, el nodo le envía a este la cadena de bytes del mensaje tal cual fue recibida.

Como se explico recién el contenido del mensaje nunca es alterado ya que esto resultaría en un deterioro del mismo, impidiéndole al usuario destino poder desplegar la imagen.

3.3.2.3.2 Mensaje Multimedia de Audio

Al igual que con los mensajes multimedia anteriores, fue reservada una dirección de la parte alta del rango. Ya que estos mensajes también tienen una estructura especial de trama y tienen que ser manipulados por los nodos de manera delicada.

La figura a continuación muestra la estructura de trama utilizada para el envío de mensajes multimedia de audio:



Como se puede ver en la figura el encabezado de estos mensajes tiene tres parámetros al igual que los mensajes anteriores. A continuación se pasan a detallar los mismos:

- **Dirección Especial:** como se puede ver en la figura los tres primeros caracteres representan la dirección reservada para el envío de mensajes multimedia de audio, en este caso se eligió la dirección 246.
- **Dirección de Destino:** los siguientes tres caracteres son la dirección de destino del mensaje multimedia, la cual es utilizada por los nodos para el enrutamiento del mensaje.
- **Dirección de Origen:** los últimos tres caracteres del encabezado indican la dirección del usuario de origen, la cual es utilizada por el terminal de destino para indicarle al usuario el remitente del mensaje.

Como se puede ver en la figura este encabezado también cuenta con tres parámetros y tiene un largo fijo de nueve caracteres, al igual que los mensajes multimedia anteriores.

El procedimiento de armado de los mensajes multimedia de audio es similar al realizado para los de imágenes. En este caso el usuario utiliza el software del terminal para grabar un mensaje de audio, el cual es almacenado como una cadena de bytes.

El encabezado del mensaje es construido como una cadena de caracteres, antes de ser transformado a una cadena de bytes. Luego esta cadena de bytes es concatenada con el array de bytes del audio, obteniéndose así el mensaje final, el cual es enviado hacia el nodo adyacente.

El procesamiento realizado por los nodos al recibir un mensaje multimedia de audio, es análogo al realizado con los mensajes multimedia antes desarrollados.

Cabe la pena aclarar que al igual que con las imágenes, en ningún momento es alterado el mensaje de audio por parte de los nodos, ya que esto haría imposible la reproducción por parte del terminal destino.

3.3.3 Mensajes de Control

Como ya fue mencionado, la red cuenta con un sistema de control, el cual permite a todos los componentes de la red estar sincronizados e informados en tiempo real del estado de la red.

Este sistema es de gran importancia para el correcto funcionamiento de la red y a pesar de encontrarse oculto para los usuarios finales, es el encargo de dar el soporte necesario para el correcto despliegue de los servicios provistos.

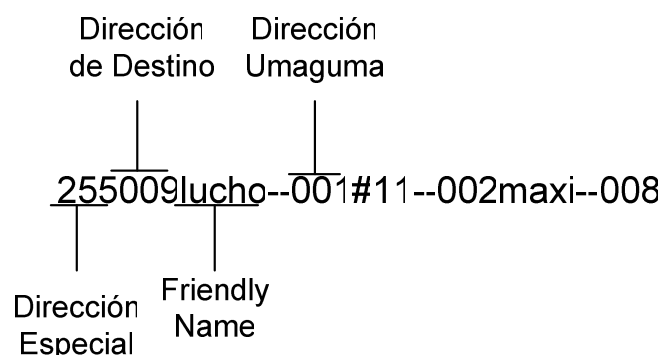
El mismo está provisto de dos partes fundamentales, en primer lugar cuenta con una lógica en cada uno de sus componentes la cual evalúa su situación, a medida que el estado del nodo va cambiando debido a ciertos eventos que se verán más adelante. El segundo gran componente de este sistema son los mensajes de control los cuales son utilizados para el intercambio de información entre los diferentes componentes de la red. Para los cuales se reservaron ciertos valores de la parte alta del rango.

A continuación se pasara a detallar cada uno de los mensajes de control utilizados por la red Umaguma. Describiendo la función de los mismos, la estructura de trama utilizada y entre que nodos de la red es intercambiado el mensaje.

3.3.3.1 Mensaje de Control 255

Este mensaje de control es utilizado por los nodos uAP y uServer para enviar tablas de usuarios conectados. En caso de que la red se encuentre funcionando en modalidad Stand-Alone, el uAP es el encargado de generar estos mensajes. En caso de contarse con un uServer, este es el encargado de generarlos. Como lo indica el título la dirección reservada para este mensaje es la 255.

A continuación se mostrara un ejemplo de la estructura de trama utilizada para estos mensajes de control:



El encabezado de este mensaje cuenta únicamente con dos parámetros, los cuales serán detallados a continuación:

- **Dirección Especial:** los primeros tres caracteres del mensaje representan la dirección especial reservada para estos mensajes de control.
- **Dirección de Destino:** los siguientes tres caracteres representan la dirección de destino global a la cual será enviado este mensaje. Al transformar esta dirección a su forma de byte, se leen los tres bits menos significativos para determinar el destino del mensaje. En caso de que fuese “111”, el mensaje es enviado a todos los celulares conectados a ese uAP.

Como se puede ver en la figura este mensaje cuenta con un encabezado de largo fijo de seis caracteres.

Luego de este encabezado se tiene el mensaje de control, el cual contiene el Friendly Name y la dirección Umaguma de usuarios que actualmente se encuentran conectados a la red Umaguma. Ambos datos se encuentran separados por un “--”, esto se debe a que el largo del Friendly Name puede ser variable, lo que hace que se necesite un símbolo identificador para poder determinar el fin de este parámetro. Esto también deriva en que el largo de estos mensajes de control sea variable, dependiendo del Friendly Name de los usuarios así como también de la cantidad de usuarios conectados.

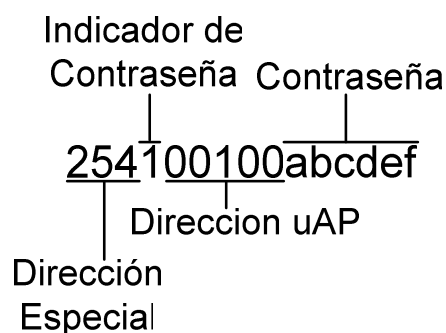
Cuando un uAP recibe un mensaje de control desde el uServer este lo procesa quitándole los últimos tres caracteres del encabezado para luego enviárselo al o a los terminales de destino, dependiendo de la dirección de destino del mismo.

3.3.3.2 Mensaje de Control 254

Como lo indica su título la dirección reservada para estos mensajes de control es el 254. Estos son utilizados para el intercambio de información entre los distintos componentes de la red, durante los distintos procesos de conexión que se dan en el sistema. Es por eso que su trama varía según su uso.

A continuación se pasará a detallar cada una de las distintas estructuras de trama utilizadas.

La figura a continuación representa la estructura de trama utilizada por el uServer para informarle a un nuevo uAP cual será su dirección Umaguma así como también si el mismo debe solicitar o no contraseña a los usuarios que ingresen a la red a través de él.

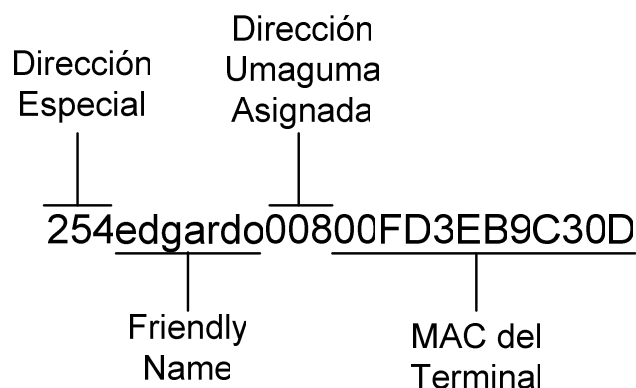


Como se puede ver esta trama está compuesta por cuatro parámetros, los cuales se pasaran a explicar a continuación:

- **Dirección Especial:** los primeros tres caracteres del mensaje representan la dirección especial reservada para estos mensajes de control.
- **Indicador de Contraseña:** este carácter, tiene dos posibles valores 0 o 1. En caso de que el uServer le indique al uAP, que debe pedir contraseña a todos los usuarios que ingresan a la red a través de él, se setea este valor en 1. En caso contrario se setea el valor en 0, lo que hace que este mensaje no contenga el último parámetro el cual representa la contraseña a solicitar.
- **Dirección uAP:** los siguientes cinco caracteres al indicador de contraseña, representan la dirección umaguma asignada por el uServer al nuevo uAP. Estos cinco caracteres son los utilizados para formar el byte de dirección.
- **Contraseña:** este parámetro es la contraseña provista por el uServer al uAP, la cual será pedida por este último a todos los usuarios que intenten ingresar a la red a través de este uAP. El largo de la contraseña es de seis caracteres.

Este mensaje tiene un largo fijo de quince caracteres, en el caso en que se tenga en uno el indicador de contraseña. En caso contrario el largo de este mensaje pasa a ser de nueve caracteres.

La figura a continuación describe la estructura de otro mensaje del tipo 254. En este caso es el utilizado por el uAP, para informarle al uServer que un nuevo usuario ha ingresado a la red.

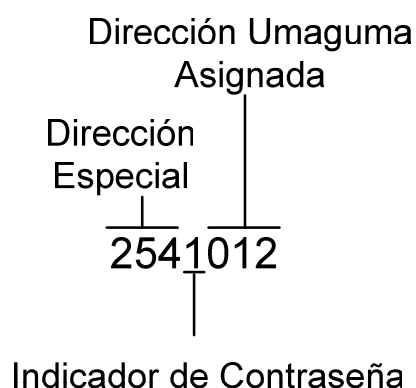


Como se puede ver en la figura esta trama está compuesta por cuatro parámetros los cuales serán explicados a continuación:

- **Dirección especial:** los primeros tres caracteres del mensaje representan la dirección especial reservada para estos mensajes de control.
- **Friendly Name:** parámetro de largo variable, utilizado para identificar al usuario en la red de manera amigable.
- **Dirección Umaguma Asignada:** los siguientes tres caracteres al Friendly Name, representan la dirección Umaguma asignada por el uAP, al nuevo usuario conectado.
- **MAC del terminal:** los siguientes doce caracteres al parámetro anterior, representan la dirección MAC del nuevo terminal conectado.

Como se puede ver en la figura, el largo de este mensaje es variable ya que el mismo depende de la longitud del Friendly Name. En este caso no fue necesaria la utilización de marcas para ubicar de manera correcta los parámetros dentro de la trama, ya que el resto de ellos tienen largo fijo.

La siguiente figura describe la última trama de los mensajes 254. Este es utilizado cuando un nuevo usuario intenta ingresar a la red Umaguma.



Esta trama está compuesta por tres parámetros los cuales serán explicados a continuación:

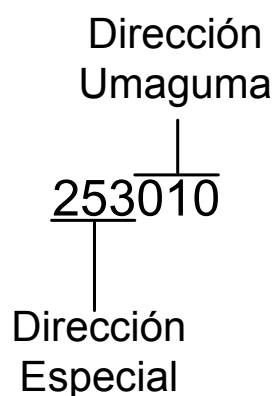
- **Dirección Especial:** los tres primeros caracteres, sirven como identificador del mismo.
- **Indicador de Contraseña:** este parámetro está compuesto por único carácter. En caso de encontrarse seteado en 1 le informa al nuevo usuario que debe validarse utilizando una contraseña. En caso de estar seteado en 0, el mismo puede ingresar sin validación.
- **Dirección Umaguma Asignada:** este parámetro está compuesto por tres caracteres y representa la dirección global asignada por el uAP al nuevo usuario conectado.

Como se pudo ver en la figura esta trama tiene un largo fijo de siete caracteres.

3.3.3.3 Mensaje de control 253

Como se puede ver en el título se reserva la dirección 253 del rango de direcciones para este mensaje. Este es utilizado por los nodos de la red Umaguma, para notificar la desconexión de un usuario. Cuando un usuario abandona la red, el uAP al cual se encontraba conectado informa al uServer mediante uno de estos mensajes. Finalmente el uServer re direcciona este mensaje a todos los uAP de la red, los cuales informan a cada uno de sus usuarios de la desconexión.

La figura a continuación es un ejemplo de la estructura de trama utilizada para estos mensajes de control:



Como se puede ver en la figura esta trama está compuesta por dos parámetros los cuales serán explicados a continuación:

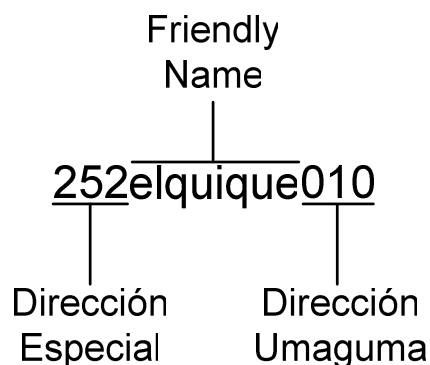
- **Dirección Especial:** los primeros tres caracteres de este mensaje representan la dirección especial reservada para este mensaje de control. Cuando un componente de la red lee un 253, asume que le llegó un mensaje informando una desconexión.
- **Dirección Umaguma:** los siguientes tres caracteres representan la dirección Umaguma global del celular que abandonó la red. Con este dato le es posible a todos los componentes de la red identificar al usuario que realizó la desconexión.

Como se puede ver en la figura la trama de este mensaje tiene un largo fijo de seis caracteres.

3.3.3.4 Mensaje de Control 252

Como se puede ver en el título la dirección Umaguma reservada para estos mensajes de control es la 252. Este es utilizado por los nodos de la red Umaguma, para informar la conexión de un nuevo usuario. Cuando un usuario ingresa la red, el nodo uAP envía un mensaje de control 254 al uServer notificando la conexión. Esto ocasiona que el uServer envíe un mensaje 252 a todos los uAP, haciendo que estos utilicen este mensaje para notificar a sus usuarios, de la conexión.

La figura a continuación muestra la estructura de trama utilizada para estos mensajes:



Como se puede ver en la figura esta trama está compuesta por tres parámetros los cuales se pasan a explicar a continuación:

- **Dirección Especial:** los primeros tres caracteres de esta trama representan la dirección Umaguma reservada para estos mensajes de control. Cuando un componente de la red lee un 252, asume que llegó un mensaje indicando el ingreso de un nuevo usuario a la red.
- **Friendly Name:** los siguientes caracteres representan el nombre amigable del nuevo usuario. Este es el desplegado por el software de los terminales a los usuarios.
- **Dirección Umaguma:** los últimos tres caracteres de la trama representan la dirección Umaguma global del nuevo usuario de la red. Este parámetro es el utilizado por el software de los componentes de la red, al momento de enviar un mensaje a este usuario.

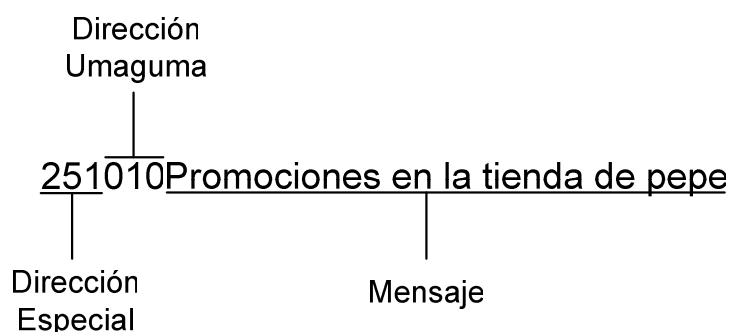
El largo de este mensaje depende del Friendly Name, ya que el mismo no tiene un largo fijo. En esta trama tampoco surge la necesidad de colocar caracteres indicadores, ya que el resto de los parámetros tiene largo fijo. Lo que hace que se pueda obtener el largo del Friendly Name, utilizando el largo del mensaje.

3.3.3.5 Mensaje de Control 251

Este mensaje de control es utilizado para el envío de mensajes de texto a usuarios por parte de los uAP, o el uServer. Como se puede ver en el título la dirección Umaguma reservada para estos mensajes de control es la 251.

Para este tipo de mensajes se manejan dos estructuras de trama, y la misma depende del nodo que genere el mensaje.

La figura a continuación es un ejemplo de trama utilizada en el envío de mensajes desde el uServer hacia un uCel.



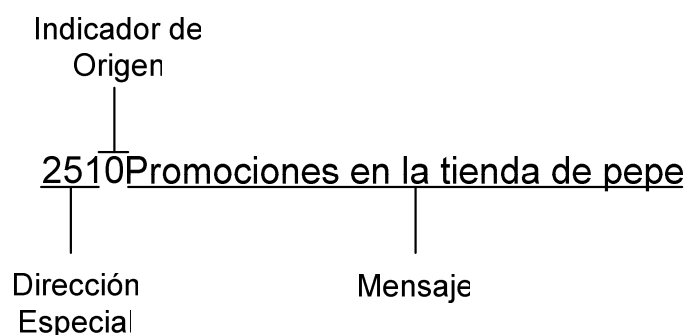
Como se puede ver en la figura la estructura de trama del mensaje enviado por el uServer, está compuesto por un encabezado concatenado al mensaje enviado por el nodo uServer. Este encabezado está compuesto por dos parámetros los cuales se pasan a describir a continuación:

- **Dirección Especial:** los primeros tres caracteres de esta trama representan la dirección Umaguma reservada para el envío de estos mensajes. Cuando un componente de la red lee esta dirección, interpreta la trama como un mensaje enviado desde un nodo.
- **Dirección Umaguma:** los siguientes tres caracteres representan la dirección Umaguma global del usuario al cual va dirigido este mensaje. Al transformar esta dirección a su forma de byte, se leen los tres bits menos significativos para determinar el destino del mensaje. En caso de que fuese "111", el mensaje es enviado a todos los celulares conectados a ese uAP.

El encabezado de este mensaje, tiene un largo fijo de seis caracteres y es concatenado delante del mensaje de texto a ser enviado al o a los terminales.

Cuando un nodo uAP recibe un mensaje 251 desde el uServer este modifica la trama, quitándole la dirección Umaguma y agregándole un identificador, el cual le permite al usuario final, determinar que nodo fue el originante del mensaje.

La figura a continuación es un ejemplo de la estructura de trama utilizada para el envío de mensajes desde un uAP.



Como se puede ver en la figura este mensaje cuenta con un encabezado compuesto por dos parámetros concatenado al mensaje.

A continuación se pasa a describir los parámetros del encabezado:

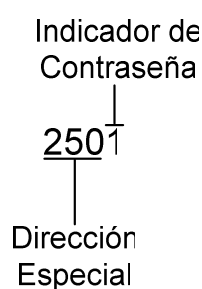
- **Dirección Especial:** los primeros tres caracteres del encabezado representan la dirección Umaguma especial, reservada para identificar estos mensajes.
- **Indicador de Origen:** este parámetro es utilizado por el software de los terminales para identificar el remitente del mensaje. En caso de ser 0, el mensaje fue originado por un uAP y en caso de que fuese un 1, el mismo fue originado por el uServer.

Como se puede ver el encabezado de este mensaje tiene un largo fijo de cuatro caracteres el cual luego se encuentra concatenado al mensaje.

3.3.3.6 Mensaje de Control 250

Este mensaje de control es utilizado por el uAP, para notificar al usuario si la contraseña enviada durante el proceso de conexión a la red Umaguma fue o no correcta. Como se puede ver en el título de esta sección la dirección especial reservada fue la 250.

La figura a continuación muestra un ejemplo de la trama utilizada para este mensaje:



Como se puede ver la estructura de trama de este mensaje está compuesto únicamente por dos parámetros, los cuales serán descriptos a continuación:

- **Dirección Especial:** los primeros tres caracteres del encabezado representan la dirección Umaguma reservada para estos mensajes.
- **Indicador de Contraseña:** es un carácter utilizado por el uAP para informarle al usuario si la contraseña antes introducida fue o no correcta. Se envía un 1 si es correcta, o un 0 si es incorrecta.

Como se pudo observar este mensaje tiene un largo fijo de cuatro caracteres.

3.3.3.7 Mensaje de Control 248

Como se puede ver en el título la dirección Umaguma reservada para este mensaje es la 248. Este es utilizado por los nodos de la red Umaguma, para realizar desconexión de usuarios. La causa puede variar y es informada en el mensaje.

La figura que se muestra a continuación es un ejemplo de trama enviada por el uServer, cuando este último desconecta un usuario de la red.



Como se puede ver en la figura esta trama cuenta con tres parámetros, los cuales se pasaran a explicar a continuación:

- **Dirección Especial:** los primeros tres caracteres de esta trama representan la dirección Umaguma reservada para estos mensajes de control.
- **Causa de Desconexión:** el cuarto carácter de esta trama representa la causa de desconexión. Existen dos causas que aplican al envío de este mensaje, las mismas son las siguientes:
 - 0 – Es utilizada cuando un uAP o un uServer, deciden desconectar a un usuario de la red
 - 1 – Esta causa es utilizada por los uAP, cuando un usuario se valida incorrectamente tres veces durante el proceso de conexión a la red.
- **Dirección Umaguma:** los últimos tres caracteres de la trama de la figura aplican solamente para los mensajes 248, provenientes desde el uServer. Estos sirven para indicarle al uAP que usuario desconectar.

Concluyendo se tiene una trama de largo fijo de siete caracteres.

Como ya se dijo, la trama de la figura es la que recibe un uAP desde el uServer, cuando este último decide desconectar un usuario. El uAP utiliza los últimos tres caracteres de esta trama para determinar el usuario a desconectar, para luego quitarlos de la misma. Por lo tanto la trama recibida por el usuario en el momento de la desconexión, es la misma que recibe en caso de que la desconexión fuese realizada por el nodo uAP.

Capítulo 4: Desarrollo de la red

4.1. uAP

4.1.1 Introducción

En las primeras etapas de desarrollo del proyecto, dedicamos la mayor parte de nuestro tiempo a implementar el envío de mensajes -mediante el protocolo Bluetooth- entre un teléfono celular y una computadora. El primer logro obtenido por el grupo fue poder desplegar en la pantalla de la computadora un mensaje de texto enviado desde el celular. A partir de ahí se fueron perfeccionando los programas en ambas puntas de la conexión hasta obtener una comunicación bidireccional y controlada entre el PC y el Terminal.

Finalmente (y luego de varios intentos fallidos), logramos perfeccionar nuestros programas hasta el punto de conseguir enviar mensajes entre dos teléfonos celulares, haciendo que el mismo pasara por la PC. Luego de alcanzar estas metas, se comenzó con el diseño del nodo uAP. Este fue el primer elemento de la red desarrollado por el grupo.

El primer diseño de este nodo era capaz de enrutar mensajes entre todos los celulares conectados a él, pero con ciertas restricciones en el mensaje, ya que este debía contener un número antes del mensaje, indicando el destinatario.

A medida que la red se fue perfeccionando, surgió la necesidad de que este uAP, fuese capaz de poder comunicarse con el uServer, de manera de poder tener una red con mayor área de cobertura.

Todo esto llevo a que se tuviera que diseñar para el uAP un sistema capaz de comunicarse con el Servidor mediante la utilización del protocolo TCP/IP. Ambos nodos intercambian entre sí una serie de mensajes de control y datos indicando por ejemplo el ingreso a la red de un nuevo usuario, así como la desconexión o la modificación de políticas, entre otros.

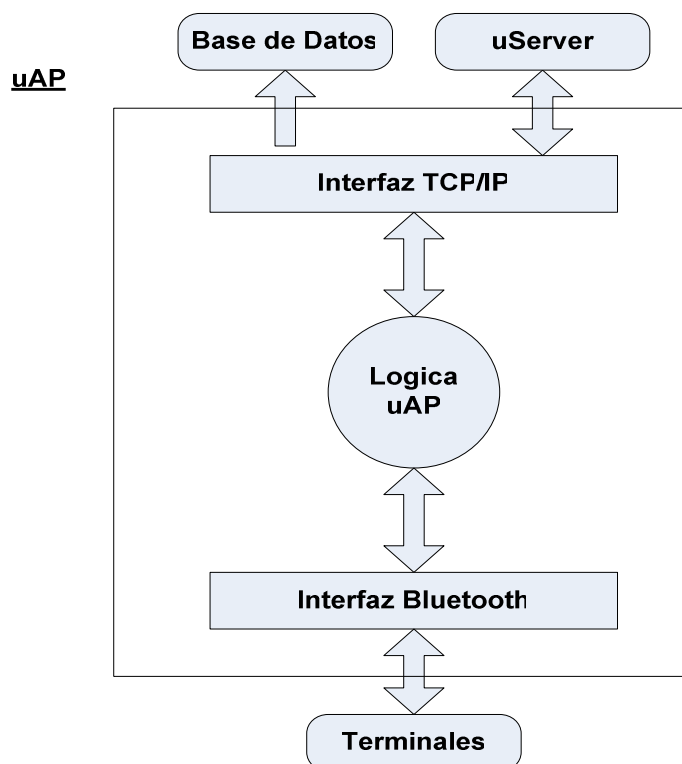
Como detalle final del nodo uAP, se diseño un procedimiento que extrae estadísticas de tráfico del nodo, datos que son enviados a la Base de Datos de manera periódica.

4.1.2 Modalidades de trabajo e Interfaces

Este nodo funciona como Punto de Acceso para todos los usuarios que pretendan utilizar los servicios provistos por la red. Se puede decir que este nodo cumple funciones similares a las que realizan las radiobases en las redes celulares, o los puntos de acceso en las redes Wifi. Cuenta con dos modalidades de funcionamiento, en primer lugar tenemos el caso Standalone en el cual el nodo trabaja solo y funciona como conmutador principal de la red, realizando todo el control de la misma. Esta modalidad de trabajo es eficiente para aplicar en sistemas que requieran un espacio de cobertura reducido.

En la segunda modalidad de trabajo, el uAP intercambia mensajes de datos y control a través de una conexión TCP/IP con un Servidor Central que se encarga de todo el control de la red. Esta modalidad es la utilizada cuando se pretenden desplegar sistemas de gran área de cobertura.

Como se puede ver en la siguiente figura, el nodo cuenta con dos interfaces hacia el exterior.



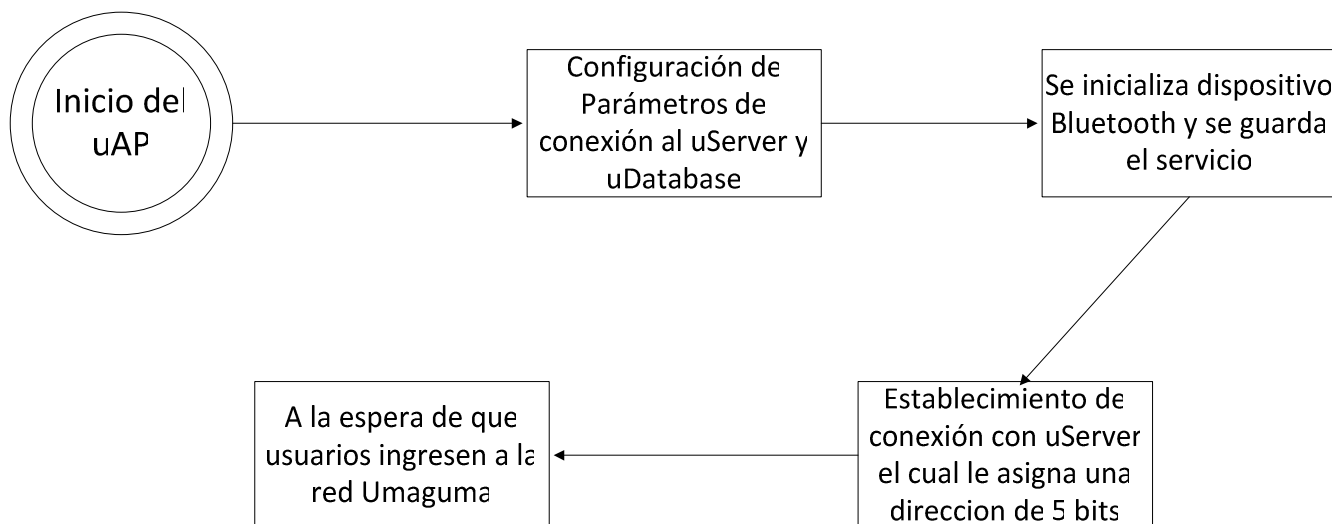
Este String tiene un formato similar al de una dirección http, y describe las características del servicio provisto por el nodo uAP. (Mas detalles serán dados en el Anexo).

Una vez desplegado el servicio Bluetooth, la clase principal crea una instancia de conUserver, (la cual hereda de la clase Thread), permitiendo de esta manera que el uAP establezca una conexión TCP/IP con el Servidor. Mientras esta conexión es establecida se bloquea el hilo que ejecuta la clase Main_AP, impidiendo de esta manera que usuarios ingresen a la red.

Luego de iniciada la conexión con el uServer, este le envía un mensaje de control 254, asignándole una dirección de cinco bits, e informándole si será necesario o no que los Terminales se autenticuen con el uAP mediante una contraseña de acceso para poder ingresar a la red. En caso de necesidad de contraseña, la misma también es enviada en el mensaje especial. Más detalles sobre este mensaje serán detallados en el capítulo 3, sobre el Protocolo de comunicaciones Umaguma.

Finalizado el establecimiento de conexión con el uServer, el hilo que maneja la instancia de la clase conUserver queda bloqueado a la espera de mensajes y la instancia de Main_AP continua su ejecución, corriendo el método Accept and Open (de la API JSR 82), permitiendo así que los usuarios puedan ingresar a la red Umaguma.

La siguiente figura resume el proceso de inicialización del nodo uAP.



4.1.3.2 Ingreso de usuarios a la red

Cuando un nuevo usuario ingresa a la red, la clase Main_AP obtiene un objeto que representa al nuevo Terminal, a partir de esto el uAP obtiene dos parámetros esenciales para la correcta identificación del usuario dentro de la red.

Los parámetros que se obtienen son los siguientes:

- **Friendly name:** Este parámetro es designado por el usuario del Terminal, y es desplegado en las listas de destinos que ven los usuarios de la red.
- **Dirección MAC:** Esta dirección identifica de manera única cualquier interfaz Bluetooth. Se designa durante la fabricación del dispositivo y no se permite su modificación.

Luego de que un usuario ingresa a la red, el nodo uAP le asigna una dirección global Umaguma (basándose en el Protocolo de Comunicaciones Umaguma), que le es informada mediante un mensaje de control “254”. Dentro del uAP, el Terminal es almacenado en una colección de la clase principal.

Una vez realizado este intercambio de información, la clase Main_AP ejecuta una instancia de la clase HiloC, la cual es ejecutada como un hilo y es la encargada de la recepción de datos desde este nuevo usuario. Se corre una de estas instancias por cada usuario que ingresa a la red. Esta instancia es la encargada de realizar la autenticación del nuevo usuario en la red. En caso de que el uServer lo haya predispuesto, el Terminal debe validarse en la red mediante una contraseña de seis dígitos para poder utilizar los servicios. En este caso el usuario tiene como máximo tres intentos para el ingreso de la contraseña, en caso de que falle en todas las oportunidades será desconectado de la misma mediante la utilización de un mensaje de control 248, y el hilo será desechado. Si el uServer no exigió validación al uAP, la conexión de los usuarios se realiza sin autenticación alguna.

Finalmente, la instancia de HiloC, ejecuta un método que informa al uServer de la presencia del nuevo usuario en la red, esto lo realiza mediante un mensaje de control 254. En respuesta a esto, el uAP recibe un mensaje de control 255, el cual contiene el Friendly Name y la dirección Umaguma de los usuarios que mantienen sesión en la red. Este mensaje es enrutado por el uAP hacia el nuevo terminal.

Los mensajes recibidos desde el uServer son manipulados por la instancia de la clase conUserServer, la cual crea una instancia de la clase HiloLan_Tx, siendo esta la encargada de enrutar de manera correcta los mensajes hacia el usuario de destino correspondiente, una vez que la instancia de esta clase envía el mensaje, la misma es desechada.

Luego de recibir desde el uServer el mensaje conteniendo la tabla, este envía a todos los uAP de la red un mensaje de control “252” informando sobre la conexión del nuevo usuario. Como se vio en el capítulo 3 (sobre el Protocolo de Comunicaciones), este mensaje contiene la dirección Umaguma global y el Friendly Name del nuevo usuario.

Fue durante el desarrollo de estas clases que el proyecto se topó con uno de los primeros grandes problemas, básicamente no se estaba consiguiendo conectar más de dos usuarios a la red simultáneamente. Al momento de que un tercer usuario intentaba ingresar alguno de los que ya se encontraba conectado perdía la sesión. Este problema parecía contradecirse con las características del perfil SPP, ya que teóricamente el protocolo permite la conexión simultánea de hasta siete Terminales con un mismo punto de acceso. En nuestro caso ni siquiera nos lográbamos acercar a ese número.

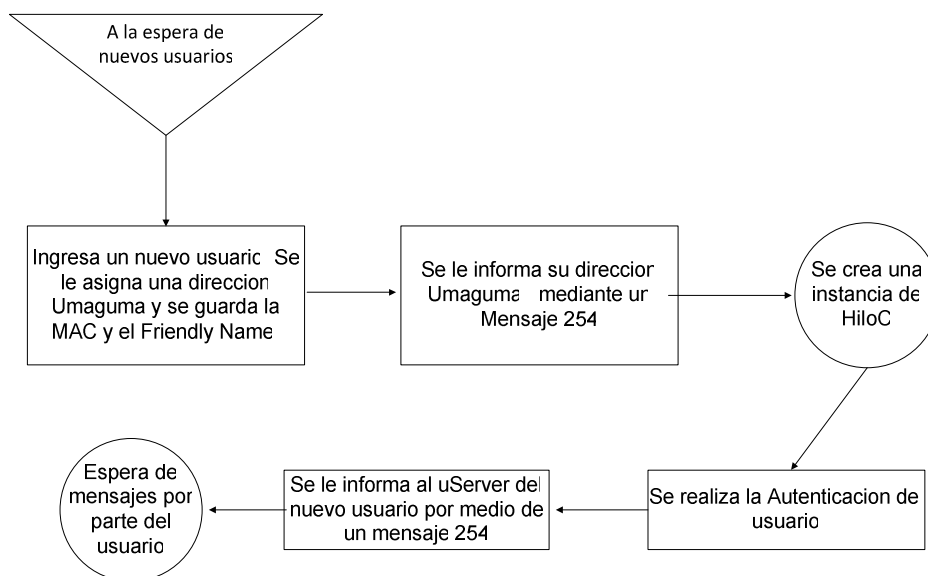
Lo primero se intentó para solucionar este inconveniente fue analizar el código utilizado por el software del Terminal y el uAP para establecer la conexión. Se modificaron las técnicas de negociación de conexión entre ambos puntos, intentando utilizar distintos esquemas de negociación pero no se pudo hallar una solución al problema.

Luego de una serie de pruebas fallidas, tomamos la decisión de estudiar el problema indagando sobre los que estaba sucediendo en las capas bajas de la pila Bluetooth, decisión que nos llevo utilizar el Sistema Operativo Linux (en particular la distribución Ubuntu). De esta manera conseguimos analizar los paquetes intercambiados entre las capas bajas de las dos puntas de la comunicación utilizando un sniffer Bluetooth denominado Hcidump.

Fue de esta manera que dimos con la raíz del problema, básicamente se debía a un intercambio de roles (en el enlace Bluetooth) causado por el proceso de búsqueda que realiza el software del Terminal antes de establecer una conexión. Lo que ocurría era que esta búsqueda forzaba a que el uAP conmutara su rol de Master a Slave, impidiendo con esto la conexión de más de dos Terminales simultáneos.

Como se verá en la sección dedicada al desarrollo del uCel, la solución a este problema consistió en no realizar una búsqueda de uAP's al momento de establecer la conexión, se optó en cambio por un esquema en el que se seleccione el Punto de Acceso desde una lista prefijada.

La siguiente figura resume el proceso de conexión de usuarios a la red.



4.1.3.3 Envío y Recepción de Mensajes de Texto

A medida que los usuarios ingresan a la red y comienzan a utilizar los servicios provistos por la misma, se hace necesario que los nodos uAP sean capaces de discernir el destino de los mensajes enviados por los Terminales.

Para esto se creó un sistema de enrutamiento de mensajes que utiliza una estructura de asignación de direcciones para cada uno de los usuarios de la red basándose en el protocolo de comunicaciones Umaguma (Capítulo 4).

Cuando un nuevo mensaje arriba al uAP, es manipulado por la instancia de la clase HiloC correspondiente a su remitente, luego se crea a su vez, una nueva instancia de la clase HiloC_Tx que efectúa el envío de mensajes a los destinatarios.

Para esto, en primer lugar se transforman los tres primeros bytes del mensaje a caracteres. Esta cadena representa, o bien la dirección Umaguma correspondiente a un destino, o bien una dirección especial, indicando de esta manera que se trata de un mensaje de control. Esto se detallara con más profundidad en el Capítulo 3.

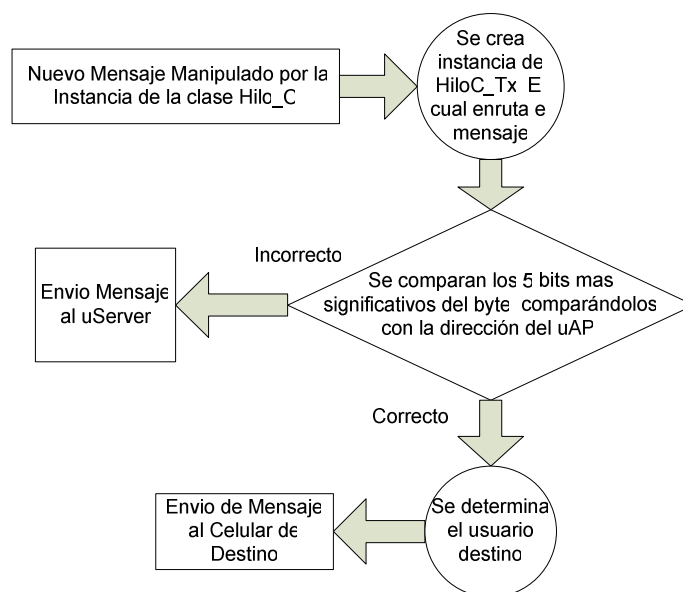
Una vez obtenida esta dirección, se ejecuta un método que realiza la conversión (de base decimal) a binario (8 bits). Los cinco bits más significativos de esta dirección indican el uAP al cual se encuentra conectado el destino, y los tres bits menos significativos indican el Terminal de destino. En el capítulo 3 se discutió la asignación de direcciones y la estructura de las mismas.

A partir de este byte se ejecuta un método que compara los primeros cinco bits de la dirección de destino del mensaje con la dirección propia (asignada por el uServer). En caso de que fuese la misma, se procede a ubicar el usuario de destino a partir de los últimos tres bits de la dirección de destino, obteniéndolo de la colección que guarda de los usuarios conectados en su área de cobertura. En caso de que la dirección del uAP de destino fuese diferente a la propia, se enruta el mensaje hacia el uServer, para que este envíe el mensaje hacia el uAP correspondiente.

Como ya fue mencionado, cuando el uAP recibe un mensaje proveniente desde el uServer, la instancia correspondiente a la clase HiloLan_TX es la encargada de realizar el correcto enrutamiento del mensaje. Para esto se analizan los tres primeros caracteres del mensaje, y se procede de la misma manera que la instancia de la clase HiloC_TX, asumiendo en este caso que el destino de este mensaje no será el uServer.

Como se explicó en el capítulo 2, la red Umaguma es capaz de transportar mensajes de texto y multimedia. Para estos últimos se reservaron dos direcciones especiales, una correspondiente a la transmisión de imágenes y otra para los mensajes de audio. Además de esto, fueron reservadas para el envío de mensajes especiales ciertas direcciones de destino, utilizados para el control de la red por parte de los nodos. Más detalle sobre estos mensajes especiales, así como las direcciones Umaguma reservadas y la estructura de trama utilizada se encuentran en el Capítulo 3.

La figura siguiente resume el proceso de enrutamiento realizado por el uAP durante el envío de mensajes originados por un usuario.



4.1.3.4 Mensajes de Datos Especiales

Al elaborar los objetivos iniciales del proyecto se proyectó el envío de mensajes de datos multimedia. En un principio se pensó únicamente en el caso de imágenes pero finalmente se logró superar estos objetivos, logrando también implementar el envío de mensajes de audio.

Como se explicó en el capítulo anterior, estos mensajes especiales tienen un entramado diferente a los mensajes de texto ordinarios.

Cuando un mensaje multimedia arriba al uAP, la instancia de HiloC_Tx procede de la misma manera que con los mensajes de texto. Al analizar los tres primeros caracteres del mensaje (y comprobar que se trata de un “247” ó “246”), esta instancia ubica la dirección del destino según lo estipulado en el Protocolo Umaguma, para finalmente proceder de la misma manera que con los mensajes de texto, en base a la dirección obtenida.

El procedimiento en el caso de los mensajes multimedia que llegan desde el Servidor es análogo al que se realiza con los mensajes de texto.

Por último solo resta describir el envío de mensajes Store and Forward, este servicio permite a los usuarios del sistema enviar mensajes de texto a usuarios que no se encuentren conectados a red. Los mensajes permanecen almacenados en la Base de Datos central y son entregados cuando el usuario ingresa nuevamente.

Cuando un mensaje Store and Forward llega al uAP desde un usuario, la instancia de HiloC_Tx analiza los tres primeros caracteres y al determinar que se trata de un mensaje especial “249”, lo envía directamente al Servidor, siendo este último el que realiza el procesamiento del mismo.

Por otro lado, si un mensaje Store and Forward llega desde el uServer, la instancia de HiloLan_TX obtiene la dirección de destino (ubicada en la trama del mensaje según lo estipulado por el protocolo Umaguma), y realiza un enrutamiento análogo al de los mensajes multimedia, con la única diferencia de que el mensaje re-enviado no contiene la dirección de destino, ya que la esta no tiene ningún tipo de utilidad una vez recibido el mensaje.

4.1.3.5 Mensajes de Control

En la medida que el proyecto comenzó a mostrar avances, paulatinamente fue resultando necesario intercambiar información de control entre los distintos componentes de la red. Si bien estos mensajes no presentan contenido relevante para los usuarios, resultan de suma necesidad para llevar un manejo ordenado y dinámico de los sucesos de la red. Al establecer el formato de las tramas para estos mensajes de control se definió utilizar las direcciones más altas del rango. La función de cada uno de estos mensajes fue explicada en el capítulo 3.

4.1.3.6 Envío de mensajes desde el uAP

Durante la etapa de diseño del uAP, se consideró la posibilidad de que el nodo fuese capaz de enviar mensajes de texto a los usuarios que se encontraran bajo su dominio. Esta funcionalidad apunta al envío de mensajes con anuncios o advertencias, los mismos pueden ser enviados a un usuario específico o realizando broadcast hacia todos los usuarios conectados a ese uAP. Para estos mensajes fue reservada la dirección 251.

4.1.3.7 Envío de estadísticas hacia la Base de Datos

Entre los objetivos fijados para el proyecto se pensó en la implementación un sistema de estadísticas que permitiera relevar el estado de los nodos y tener así un panorama global del funcionamiento de la red.

Fue con este fin que se desarrolló la clase Stats, a la cual se le ingresan ciertos datos estadísticos relevantes para evaluar el estado del nodo. Cuenta también con métodos que son llamados por las instancias de HiloC_Tx y HiloLan_TX en el momento que reciben mensajes de datos y son los encargados de modificar estos parámetros.

Los datos relevados por el nodo son los siguientes:

- Cantidad de mensajes de texto que circulan por el uAP
- Cantidad de mensajes de imágenes que circulan por el uAP
- Cantidad de mensajes de voz que circulan por el uAP
- Cantidad de bytes de mensajes de texto que circulan por el uAP
- Cantidad de bytes de mensajes con imágenes que circulan por el uAP
- Cantidad de bytes de mensajes de voz que circulan por el uAP

- Cantidad de bytes de todos los tipos de mensajes que circulan entre celulares bajo el dominio de un mismo uAP.
- Cantidad de bytes de todos los tipos de mensajes que van desde ese uAP hacia el uServer.

Estos datos son guardados periódicamente en la Base de Datos central junto con información sobre la fecha y hora correspondientes, así como la dirección Umaguma del uAP en cuestión.

4.1.4 Modalidad de trabajo Stand-Alone

Durante los primeros meses de trabajo, el proyecto se enfocó más que nada en el diseño y desarrollo del uAP. Esto llevó a que las primeras versiones del nodo funcionen únicamente en modalidad Standalone, dado que no se disponía aún de un Servidor desarrollado. Para estos casos, el uAP se convierte en el nodo con mayor jerarquía del sistema, utilizando una dirección configurada previamente.

La inicialización de este nodo comienza con la puesta en servicio de la interfaz Bluetooth correspondiente, de igual manera que en la modalidad completa (con Servidor). Se registran también los mismos datos de usuario cada vez que un nuevo Terminal ingresa a la red.

En esta modalidad el uAP, debido a que encabeza la jerarquía de la red, es el encargado de la creación, procesamiento y transmisión de todos los mensajes de control utilizados. No se incluye en este caso el servicio de mensajería de Store and Forward, ya que no se dispone de Base de Datos para el almacenamiento, por esto mismo tampoco se implementa el sistema de reporte de Estadísticas.

4.2. uCel

4.2.1 Introducción

Para finalizar con esta descripción de los nodos de la red, nos abocamos a los detalles de la implementación del terminal de usuario. Este nodo tiene la particularidad de que su desarrollo se construyó utilizando una plataforma de programación ligeramente diferente al resto: Java 2 Micro Edition, la cual se caracteriza por tener clases y funcionalidades

enfocadas al desarrollo de aplicaciones que corren sobre dispositivos móviles, contemplando por ejemplo el manejo del teclado utilizado por los teléfonos celulares.

4.2.2 Establecimiento de conexión

El primer objetivo que nos planteamos al encarar esta aplicación (junto con el nodo uAP), fue el de establecer conexiones Bluetooth entre terminales y computadoras que sean capaces de enviar cadenas de texto entre sí. Para esto utilizamos el protocolo RFCOMM (Radio Frequency Communication), implementado mediante el perfil SPP (Serial Port Profile), con el cual podemos establecer conexiones punto a punto que se comportan como un canal serial. Estas funciones se encuentran incluidas en la API de Java JSR 82, implementada por la librería BlueCove.

Inicialmente utilizamos un mecanismo mediante el cual el terminal realizaba una búsqueda de posibles puntos de acceso para luego seleccionar con cual iniciar la conexión. Para la búsqueda de dispositivos utilizamos SDAP (Service Discovery Application Profile), perfil que implementa el protocolo SDP (Service Discovery Protocol). Este esquema parecía funcionar correctamente en un comienzo, pero luego resultó tener dos desventajas importantes que nos hicieron optar por un esquema de conexión distinto con listas fijas de puntos de acceso.

En primer lugar, las búsquedas y establecimiento de conexión resultaron demasiado lentos e inestables para lo que era nuestra idea de funcionamiento. En segundo lugar surgió un problema que alteraba el correcto funcionamiento del resto del sistema. Para realizar una búsqueda de servidores SPP el terminal encuesta a todos los dispositivos Bluetooth visibles, y en este proceso obliga al dispositivo encuestado a cumplir un rol de Esclavo en la conexión. Es problema es que cuando se realiza la encuesta a un nodo uAP, este se ve obligado a invertir los roles (Maestro-Esclavo) en las restantes conexiones. Y dado que los dispositivos Bluetooth admiten hasta 7 conexiones simultáneas (una Piconet) únicamente estando en estado Maestro, no solo se imposibilitaba establecer la nueva conexión, sino que además se comenzaban a caer las existentes. La única salida que encontramos a este problema fue la de monitorear constantemente el estado de las conexiones (mediante un Sniffer Bluetooth como Hcidump) y utilizar dinámicamente comandos que modificaran los roles hacia su estado correcto. Esto resultó definitivamente poco práctico para el manejo del sistema, lo cual nos volcó a decidimos por el otro esquema de conexión, utilizando listas fijas.

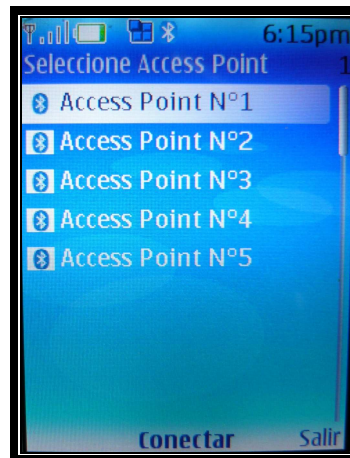


Figura 4.1 – Pantalla de selección de punto de acceso

Una vez definido el punto de acceso a utilizar, podemos comenzar el inicio de sesión, utilizando el método de Java `Conector.open(String direccionURL)` que corresponde al perfil SPP. A este método se le debe pasar como parámetro la dirección URL del uAP, que en nuestro caso tiene siguiente formato:

btsp://direccionMAC:1;authenticate=false;encrypt=false;master=false

Con esto estamos indicando que iniciaremos una conexión utilizando el perfil SPP (RFCOMM), hacia un dispositivo con cierta dirección MAC y que utilizaremos el canal de frecuencia número 1. El término “authenticate=false” indica que se no solicitará clave de autenticación para establecer la conexión. También estamos especificando que no utilizaremos encriptado de la información, en tanto que con “master=false” estamos indicando que el servidor SPP está dispuesto a ser tanto maestro como esclavo en la comunicación.

Luego de establecer la conexión a nivel de Bluetooth, estamos en condiciones de abrir los flujos de entrada y salida para los datos (desde y hacia el punto de acceso), mediante los cuales realizamos las transferencias de bytes con la red. Estas funcionalidades se encuentran implementadas en el programa dentro de las clases `HiloTx` y `HiloRx`, en donde encontramos los métodos correspondientes al envío y recepción de mensajes Umaguma. Lo primero que se transmite durante la comunicación es un mensaje especial desde el uAP con el siguiente formato:

250	Bit de autenticación	Dirección global del nuevo usuario
-----	----------------------	------------------------------------

En este mensaje se le informa al terminal cual será su dirección global dentro de la red y si debe o no solicitar contraseña de ingreso, esto último se informa a través de un bit

de autenticación. En caso afirmativo, el terminal solicitará al usuario el ingreso de la contraseña correspondiente, la cual se envía directamente al uAP (sin ningún entramado) para su comprobación. Luego éste devuelve un mensaje especial en el que informa si la contraseña fue correcta o no. El usuario dispone de 3 intentos para realizar la autenticación, de ser incorrecta la conexión es dada automáticamente de baja por el uAP, de ser correcta la sesión finalmente se da por iniciada en la red Umaguma, con lo cual estamos en condiciones de iniciar las comunicaciones.

4.2.3 Manejo de usuarios

Una vez solucionado el tema del establecimiento de conexiones y transmisión de paquetes hacia el uAP, debemos fijar criterios para manejar los destinos de la red en el terminal. Lo primero que recibe un usuario recién autenticado es una lista -proveniente del Servidor- en la que se le informa el estado actual de la red (mensaje especial "255"), en ella se encuentran especificados todos los usuarios conectados y sus respectivas direcciones globales. Al recibir este mensaje el Terminal activa todas sus funciones y despliega la tabla de destinos mediante la cual el usuario puede comenzar a enviar mensajes a través de la red.

El mensaje especial en el cual se reporta al terminal la lista de destinos alcanzables es también utilizado por el sistema para actualizar de manera rápida la lista completa de destinos en momentos posteriores, esto por ejemplo ocurre cuando se modifican las políticas de visibilidad de un uAP. Se dispone además de dos mensajes adicionales con funciones más granulares para la modificación de las listas, por un lado el mensaje especial "252", con el cual se le informa al Terminal que se debe agregar un destino a la lista y el mensaje especial "253" para informarle que debe retirarse un destino. El funcionamiento y estructura de estos mensajes fue descrito anteriormente en el capítulo número 3.

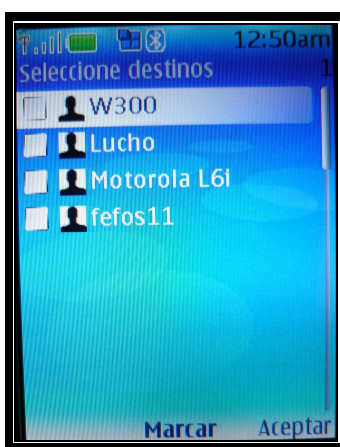


Figura 4.2 – Pantalla de selección de destinos

Con la lista de destinos del Terminal implementada y actualizada estamos en condiciones de comenzar a traficar mensajes entre los distintos usuarios, para lo cual se envía a través de la red (como fue definido en el protocolo de comunicaciones Umaguma) una trama en la que se especifican las direcciones globales de origen y destino junto con el mensaje. Para esto el usuario tiene acceso a una lista en la que puede seleccionar uno o múltiples destinos para realizar el envío del mensaje. El destinatario recibe los mensajes y los despliega en la pantalla principal ordenados según su llegada junto con el nombre del remitente. Se dispone también de una opción para hacer limpieza de pantalla, eliminando todos los mensajes recibidos anteriormente.

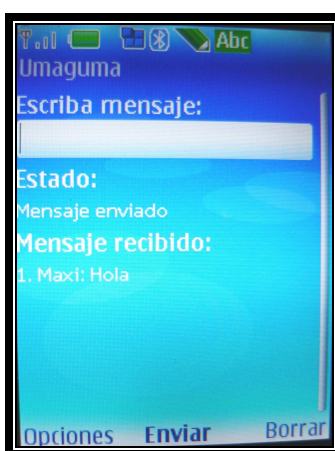


Figura 4.2 – Pantalla principal

4.2.4 Mensajes de texto especiales

Como casos particulares de envío de mensajes de texto tenemos dos ejemplos: los mensajes del sistema y los mensajes Store and Forward. En el primero de los casos llega al terminal un mensaje que en lugar de provenir desde otro usuario, lo hace directamente desde la administración de la red, pudiendo ser tanto desde el uAP como del uServer. Estos mensajes son generados desde la interfaz gráfica del nodo origen y se envían encapsulado dentro del mensaje especial N° 251. Dentro del entramado se envía la información sobre el origen del mensaje.

El segundo caso son los mensajes Store and Forward, su particularidad consiste en que el destinatario puede encontrarse desconectado al momento de realizar el envío, lo cual provoca que el mensaje quede almacenado en la Base de Datos central a la espera de ser enviado ni bien el usuario destino inicie sesión. Para hacer uso de este servicio el usuario escribe un mensaje como lo hace usualmente e ingresa a la opción “Mensaje SnF”, finalmente se debe digitar la dirección física (MAC) del destino efectuando así el envío del

mensaje. Una vez que el destino ingresa a la red recibe el mensaje que se encontraba almacenado en la Base de Datos y lo despliega como una pantalla emergente. Junto con el mensaje se muestra información sobre el remitente, la fecha y la hora de envío.

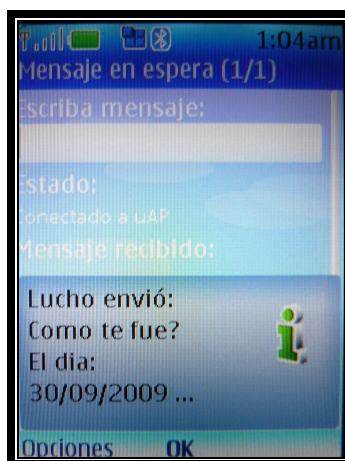


Figura 4.3 – Recepción de mensaje Store and Forward

Durante la definición del mecanismo de envío de mensajes Store and Forward surgió el planteo de mejorar el sistema de ingreso del destino para, en lugar de utilizar la dirección MAC, utilizar o bien una nueva numeración que se asocie de manera única a cada dispositivo que alguna vez haya ingresado al sistema, o bien el número MSISDN correspondiente al teléfono celular sobre el que se ejecuta la aplicación. Esto último presentaba la complicación de tener que interactuar con el operador celular para averiguar dicho número en cada usuario de la red, cosa que a priori no resulta sencilla. En resumen, no se encontró para este problema una solución cuantitativamente más práctica o accesible que la de ingresar la dirección MAC, por lo que decidimos de momento utilizar este sistema, quedando planteado como una posible mejora a futuro.

4.2.5 Mensajes multimedia

Uno de los servicios implementados sobre la red Umaguma es el envío y recepción de mensajes multimedia. Para esto se hace uso de los dispositivos hardware embebidos en los terminales sobre los que corremos la aplicación. Los archivos son capturados utilizando las opciones “Enviar Audio” y “Enviar Foto”, presentes en la pantalla principal de la aplicación, con lo cual iniciamos el asistente de captura para cada caso respectivamente. La interacción entre la aplicación y los dispositivos se hace mediante la API de Java JSR 135 (MMAPI).

Para la transmisión de grabaciones de voz utilizamos el mensaje especial “246” en tanto que para las imágenes utilizamos el “247”. El entramado y transmisión se realiza de manera muy similar para ambos casos. Cuando uno de estos mensajes es recibido en el terminal se despliega una pantalla emergente que muestra el archivo recibido junto con el nombre del remitente. Para el caso de los mensajes de audio se dispone de un botón Reproducir que puede ser ejecutado tantas veces como se desee.

La codificación utilizada para las imágenes tomadas por la cámara del terminal es PNG (Portable Network Graphics), este formato gráfico está basado en un algoritmo de compresión sin pérdida para mapas de bits no sujeto a patentes. Este formato fue desarrollado en buena parte para solventar las deficiencias del formato GIF y permite almacenar imágenes con una mayor profundidad de contraste y otros importantes datos [1].

En el caso de las grabaciones de voz se implementó un sistema de codificación inteligente capaz de adaptarse a la compatibilidad del dispositivo. En primer lugar la aplicación intenta utilizar el códec de compresión AMR (Adaptative MultiRate), en caso de no estar disponible en del terminal el mensaje se envía utilizando PCM (Pulse-Code Modulation), que representa un estándar mucho mas compatible por no aplicar compresión sobre la señal digitalizada de voz. AMR realiza una codificación de la señal con tasa adaptativa, optimizada para la compresión de voz. Es por ejemplo el sistema de codificación utilizado por la tecnología celular GSM. Al recibirse el mensaje se detecta la codificación utilizada y se aplica la descompresión correspondiente [2].

4.2.6 Fin de sesión

El mecanismo de finalización de la sesión es considerablemente más sencillo que el de inicio. Básicamente podemos separarlo en tres casos:

- El usuario desea abandonar la red
- El sistema da de baja la conexión del Terminal
- Se termina la conexión debido a un fallo en la comunicación

En el primer caso el terminal sencillamente envía un mensaje al uAP (mensaje especial “248”), para notificar la baja al sistema y posteriormente se cierra el programa. En el resto de la red se distribuye la información sobre el retiro de este usuario.

Cuando la finalización es impuesta por el sistema, el Terminal recibe el mensaje especial “248” en el cual se incluye un campo que indica el motivo de la desconexión. Actualmente manejamos dos posibles motivos que son:

- Por decisión del administrador de la red a través de la interfaz gráfica
- Porque el usuario se equivocó 3 veces al ingresar la contraseña de autenticación.

Queda planteada la posibilidad de agregar a futuro otros motivos de desconexión. Cuando el Terminal recibe este mensaje da por terminada la conexión, deshabilitando los comandos de la aplicación y avisando al usuario de la baja.

En el caso de fallos, el terminal es capaz de detectar -al cabo de pocos segundos- la ausencia de conectividad, por lo que asume como finalizada la sesión, deshabilitando las opciones de la aplicaciones.

4.2.7 Interfaz grafica de usuario

Para la implementación de la interfaz grafica del Terminal se procuró realizar un diseño de las distintas pantallas y menús que facilite lo más posible la interacción con el usuario. En vista de esto intentamos ordenar los botones, combos, menús e imágenes de una manera conveniente y priorizando principalmente el fácil acceso a las funcionalidades.

Se utilizaron para este desarrollo las diferentes clases que J2ME dispone para este fin, en particular las que se encuentran en el paquete lcdui. En la mayoría de los casos se utilizaron las funcionalidades de alto nivel que proveen las clases Form, Textbox, List y Alert, subclases de Screen. En el capítulo 8 se discutirá mucho más a fondo los detalles de las interfaces gráfica de cada una de las aplicaciones.

4.3. uServer

4.3.1 Introducción

Luego de desarrollar la primera versión del uAP, donde se logró enrutar mensajes de texto entre Terminales conectados a dicho nodo, surgió la necesidad de expandir la red. Para esto, como se explicó en el Capítulo 2, se pensó en una arquitectura centralizada, que consistía en un Servidor al que se conectarían todos los AP de la red. Es en este punto, que se comienza el desarrollo del uServer.

Inicialmente se debió estudiar el establecimiento de conexiones TCP/IP entre dos computadoras (Punto de Acceso y Servidor), ya que hasta el momento solo se había

logrado establecer conexiones entre un celular y una computadora a través de Bluetooth. Luego de ahondar en el funcionamiento de las conexiones y el envío de datos utilizando TCP/IP, se consiguió enviar un mensaje de texto entre dos Terminales conectados a dos puntos de acceso distintos, los cuales se encontraban a su vez interconectados por el Servidor. Este “enrutamiento” era exitoso solo siguiendo un orden adecuado en la conexión de los distintos elementos a la red (Puntos de Acceso y Terminales), pero fue fundamental para observar el funcionamiento de las conexiones y el envío de datos a través de TCP/IP, y así poder comenzar con el diseño del uServer.

La primera versión del uServer solamente era capaz de enrutar mensajes de texto entre dos o más uAP’s e intercambiar con éstos la mínima cantidad necesaria de mensajes de control como para estar al tanto de las conexiones. Llegado este punto se comenzó a enriquecer el funcionamiento del Servidor, hasta llegar -luego de un largo proceso de desarrollo- a la versión final del uServer que –entre otras cosas- es capaz de:

- Enrutar mensajes de texto, imágenes y archivos de voz.
- Administrar y gestionar la red por medio de mensajes de control que permiten dividir la red en zonas y asignar nombres y contraseñas a los uAP, establecer políticas de visibilidad entre los distintos nodos y dar de baja cualquier conexión (Punto de Acceso o Terminal).
- Interactuar con una Base de Datos remota para almacenar estadísticas sobre el funcionamiento del Servidor e implementar un sistema de mensajería Store and Forward que permite a los usuarios enviar mensajes de texto a Terminales que no se encuentren conectados a la red.
- Interactuar directamente con los Terminales conectados a la red, por medio del envío de mensajes de texto e imágenes desde el uServer.

4.3.2 Funcionamiento General

La función principal del uServer es la de interconectar todos los puntos de acceso pertenecientes a la red de manera de que Terminales conectados a distintos uAP puedan interactuar entre sí, pero como se mencionó en la introducción, también juega un rol importante a la hora de administrar y gestionar la red. Para poder llevar a cabo todas las funciones mencionadas, el uServer debe mantener constantemente una conexión TCP/IP con cada uno de los uAP involucrados en la red. Teniendo en cuenta las dos variantes posibles en el orden de la conexión entre dos nodos, se decidió que el Servidor fuera quién estuviera permanentemente escuchando conexiones de nuevos puntos de acceso, y que estos se encargaran de iniciar el establecimiento de la conexión con el Servidor.

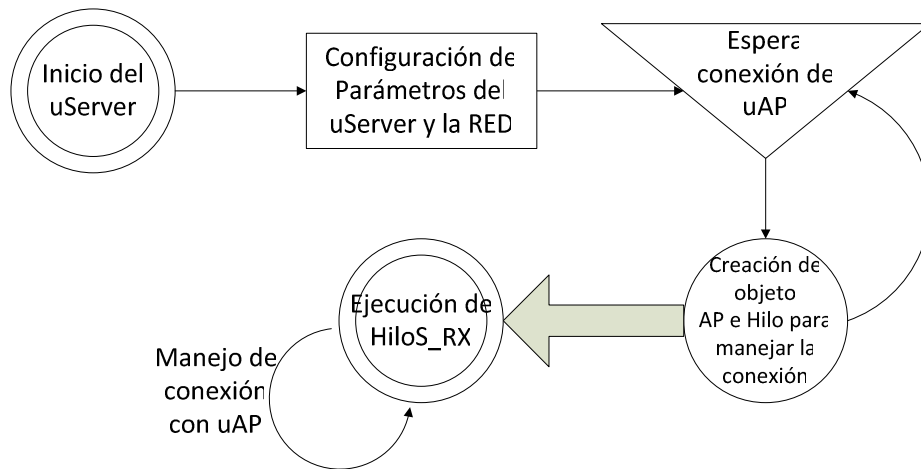
Es debido a lo mencionado en el párrafo anterior que el uServer debe ser iniciado antes que cualquier uAP que vaya a formar parte de la red, y por lo tanto, para poner en funcionamiento una red Umaguma que cuente con Servidor central (o sea, que tenga dos o más uAP), el primer paso es iniciar el Servidor.

Fue así que se comenzó a diseñar el uServer considerando que la red Umaguma iba a entrar en funcionamiento cuando se iniciara el Servidor, y es por esto que la primer tarea que cumple dicho nodo (incluso antes de aceptar conexiones de los uAP) es establecer, por medio de un cuadro de configuración que aparece al ejecutar el software, los siguientes parámetros:

- Nombre de la red.
- Puerto TCP en que se escucharán conexiones de los uAP.
- Máximo de uAP permitidos en la red.
- Dirección IP en que se encuentra la Base de Datos.
- Puerto TCP en que se realizará la conexión con la Base de Datos.
- Activación o desactivación del uso de contraseñas en los uAP.

Una vez realizada la configuración de los parámetros del Servidor (y la red), este queda a la espera de conexiones desde los distintos uAP que integrarán la red. Cada vez que se conecta un nuevo nodo, el uServer le asigna una dirección global de cinco bits y crea un objeto para representar a dicho uAP.

Inmediatamente después de crear el nuevo uAP (instancia de la clase DispAP, ver Capítulo 2), el uServer crea y ejecuta un Hilo (Thread) para manejar la conexión con dicho uAP, y así poder volver al estado de escucha de nuevas conexiones lo antes posible. En el siguiente diagrama se ilustra el funcionamiento básico del uServer mencionado en los párrafos anteriores.



Figura

4.4 – Esquema de funcionamiento del uServer

4.3.3 Manejo de conexión con un nuevo uAP

Como se mencionó en la sección anterior, ni bien se conecta un nuevo uAP a la red (y se crea un objeto para representar la nueva conexión), se instancia y ejecuta un objeto de clase HiloS_RX para manejar la conexión con el nuevo nodo y liberar cuanto antes los recursos destinados a la “escucha” de conexiones de nuevos AP.

El funcionamiento de este hilo es muy similar al funcionamiento general del uServer. En primera instancia se pasa por un proceso de configuración, en el cual se establecen los siguientes parámetros propios de cada uAP:

- Nombre del uAP.
- Zona en que se encuentra el uAP.
- Contraseña de 6 dígitos para restringir las conexiones al uAP en cuestión (en caso de que se decida utilizar contraseña en el AP y que al iniciar el uServer se haya activado el uso de contraseñas en los uAP).

Una vez que se setean los parámetros descritos, el uServer le envía al nuevo uAP una trama “254” (ver Capítulo 4) conteniendo su dirección y la contraseña para restringir la conexión de Terminales (en caso de que se utilice ésta política). Realizada esta negociación, el uAP se considera conectado a la red, por lo que recién en éste punto se agrega el objeto que representa a este nodo a una colección que contendrá a todos los uAP conectados (colección mapaAP) y se despliega el nuevo nodo en el árbol en que se ilustran los elementos conectados en la interfaz gráfica.

Superado el establecimiento de la conexión con el nuevo uAP, la instancia de HiloS_RX correspondiente al nodo en cuestión se bloquea en espera de datos provenientes de dicho nodo.

Cada vez que llega una trama desde el uAP, se crea y ejecuta una instancia de HiloS_TX cuya función es únicamente la de procesar la trama recibida. De manera similar al caso de la clase HiloS_RX se utilizó un hilo que se ejecuta en paralelo al resto del programa para manejar cada una de las tramas recibidas para que, de esta manera, la instancia que maneja la conexión con el AP vuelve al estado de espera lo antes posible.

A continuación se ilustra el manejo de la conexión con un uAP.

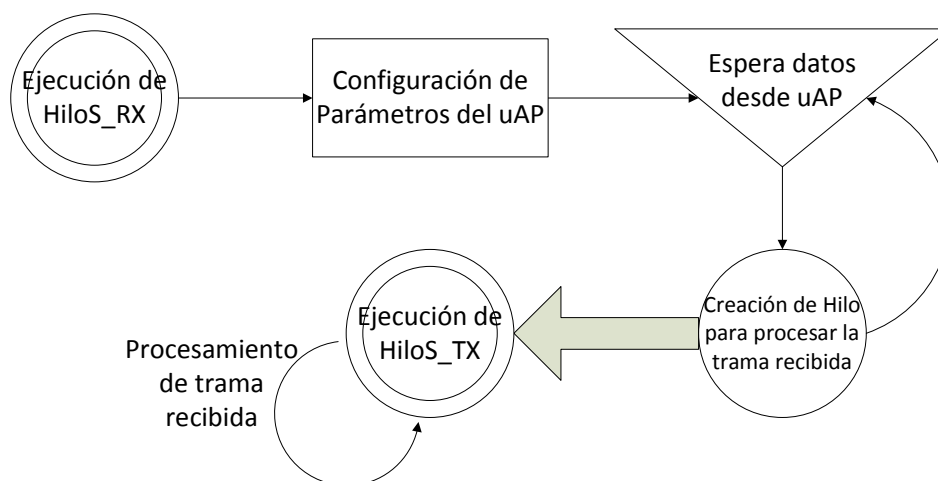


Figura 4.5 - Esquema de manejo de conexión con uAP

4.3.4 Procesamiento de nuevas tramas

Las tramas provenientes de los uAP son leídas por las instancias correspondientes de la clase HiloS_RX y procesadas en una instancia de HiloS_TX que se crea exclusivamente para cada trama. La única verificación que se realiza dentro de Hilo_RX es comparar el primer byte leído con “-1”, ya que esto implicaría que se perdió la conexión con el uAP y por lo tanto, el uServer deberá proceder con la “desconexión” del mismo (ver Sección 4.3.5).

Si la información recibida es válida, se crea y ejecuta una instancia de HiloS_TX a la que se le pasa como parámetro la trama a procesar. Para esto se convierten los 3

primeros bytes a un String que, como se vio en el Capítulo 3, indican si se trata de un mensaje especial (y de qué tipo) o de un mensaje de texto ordinario.

A continuación se describe como reacciona la instancia de HiloS_TX dependiendo del valor de los 3 primeros bytes de la trama.

➤ **Mensaje Especial “246” o “247”**

246	Dirección de destino	Dirección de origen	Archivo de voz
247	Dirección de destino	Dirección de origen	Imagen

Cuando los primeros 3 bytes coinciden con el String “246” o “247”, la trama recibida corresponde a un mensaje de voz o una imagen. En este caso la función del Servidor se reduce a reenviar el mensaje hacia el Terminal de destino. Para esto toma los 3 bytes siguientes de la trama (que corresponden a la dirección Umaguma del destino) y la reenvía hacia el uAP correspondiente.

➤ **Mensaje Especial “254”**

254	Friendly Name	Dirección Umaguma	Dirección MAC
-----	---------------	-------------------	---------------

Un mensaje especial “254” se recibe desde un uAP cada vez que un nuevo Terminal se conecta a la red, con esto se notifica al uServer de la conexión del nuevo usuario, para que tome las siguientes medidas:

- Crear un objeto de la clase *DispCel* para poder representar al nuevo Terminal, a partir de la información enviada por el uAP (friendly name, dirección Umaguma y MAC).
- Crear y enviar al uAP una tabla conteniendo los friendly names y las direcciones Umaguma de todos los Terminales conectados a la red en el momento.
- Avisar a todos los uAP presentes sobre la conexión del nuevo uCel para que lo difundan hacia sus respectivos Terminales (utilizando el *mensaje especial “252”*, ver Capítulo 4).
- Agregar una nueva instancia de *DispCel* (correspondiente al nuevo uCel) a la colección *mapaCel* que contendrá todos los Terminales conectados a la red, y desplegar el nuevo elemento en el árbol de la interfaz gráfica.
- Consultar en la Base de Datos si existen mensajes Store and Forward almacenados para el Terminal en cuestión. En caso afirmativo se deben enviar dichos mensajes al uAP correspondiente (utilizando el mensaje especial “249”).

➤ **Mensaje Especial “253”**

253	Dirección Umaguma del Terminal desconectado
-----	---

A diferencia del caso anterior, el mensaje especial “253” se recibe cada vez que un usuario se desconecta de la red. La notificación proviene del punto de acceso correspondiente al Terminal desconectado. Cuando esto sucede, el uServer:

- Reenvía el mensaje recibido al resto de los uAP para que estén al tanto y difundan hacia los uCel la desconexión del usuario.
- Obtiene la dirección Umaguma del Terminal a partir de la trama recibida para poder referenciar el objeto con el cual se representa dicho Terminal, y así eliminarlo de la colección en que se almacena y del árbol de la interfaz gráfica.

➤ **Mensaje Especial “249”**

249	Dirección MAC del destino	Dirección Umaguma de origen	Mensaje
-----	---------------------------	-----------------------------	---------

El caso de la dirección especial “249” corresponde al envío de mensajes Store and Forward. La primera acción que realiza el Servidor en estos casos es verificar que efectivamente el Terminal de destino no se encuentra conectado a la red. Para esto, se obtiene la MAC del destino a partir de la trama recibida y se busca en la colección de Terminales conectados. Llegado éste punto, se distinguen 2 casos:

- **Primer Caso:**
El Terminal al cual se envía el mensaje se encuentra conectado a la red. Este caso no es el usual, pero puede darse, por ejemplo, si el usuario que envía el mensaje no es consciente de que el usuario de destino se encuentra conectado.
De darse esta situación, el uServer convierte la trama (Store and Forward) en un mensaje de texto ordinario y lo envía al uAP correspondiente para que el destino lo reciba como si fuera un mensaje de texto común, obteniendo la información necesaria de la trama recibida.

- **Segundo Caso:**

Este caso corresponde al uso normal de los mensajes Store and Forward, en el cual el Terminal con la MAC de destino (incluida en la trama) no se encuentra conectado a la red. Para estos casos el uServer realiza lo siguiente:

1) Obtiene a partir de la trama (y de la información que el mismo Servidor tiene almacenada) los datos necesarios para insertar el mensaje en la tabla de Store And Forward de la Base de Datos (MAC del destino, MAC y friendly name del origen, y mensaje de texto).

2) Se guarda en la Base de Datos la información obtenida en el punto anterior junto con la fecha y hora en que se realiza dicha acción.

➤ **Mensaje de Texto “ordinario”**

Dirección Umaguma de destino	Dirección Umaguma de origen	Mensaje
------------------------------	-----------------------------	---------

Por último, si los primeros 3 bytes de la trama recibida no corresponden a ninguno de los mensajes especiales mencionados, se deduce que la trama corresponde a un mensaje de texto ordinario y se procede a enrutar dicha trama al uAP correspondiente. En este caso, los 3 primeros bytes de la trama corresponden a la dirección Umaguma del destino, y por lo tanto, la única función que realiza el Servidor es la de reenviar el mensaje hacia el uAP correspondiente.

4.3.5 Pérdida de conexión con un uAP

Como se explicó en la sección anterior, cuando desde una instancia de HiloS_RX se recibe el byte correspondiente a “-1”, indica que se perdió conexión con el uAP que corresponde a dicha instancia. Cuando esto sucede, el uServer (desde la clase HiloS_RX) maneja la desconexión del uAP de la siguiente manera:

- Setea como ‘falso’ la variable “corriendoSR” para que la instancia de HiloS_RX salga del loop que está constantemente esperando nuevos datos del uAP correspondiente.
- Envía a todos los uAP que quedaron conectados un mensaje especial “253” por cada uno de los Terminales que se encontraban conectados al nodo que perdió conexión.

- Elimina de la colección mapaAP al nodo que se desconectó y de mapaCel a todos los Terminales que estaban conectados a dicho nodo. También se sacan estos elementos del árbol que se muestra en la interfaz gráfica.

4.3.6 Interacción con la Base de Datos

Como ya se mencionó, el uServer interactúa con una Base de Datos para llevar a cabo los reportes estadísticos y el manejo de mensajería Store And Forward. Para esto, como se vio en el capítulo 2, el Servidor cuenta con una clase (BdD) que se encarga de ingresar, consultar y borrar registros de la Base de Datos, esta clase se instancia y ejecuta (en paralelo al resto del programa ya que se trata de un Hilo) inmediatamente después de que se inicia el Servidor.

➤ Manejo de Estadísticas

Para realizar las funciones deseadas, la clase BdD cuenta con diversos contadores que son incrementados o decrementados a partir de datos que recibe o envía el uServer.

Estos contadores controlan:

- Trafico de mensajes de texto a través del uServer.
- Trafico de bytes debidos a los mensajes de texto.
- Trafico de imágenes a través del uServer.
- Trafico de bytes debidos a las imágenes.
- Trafico de archivos de voz a través del uServer.
- Trafico de bytes debidos a los archivos de voz.
- Mensajes que se almacenan en el Store And Forward.
- Bytes que se almacenan en el Store And Forward.
- Mensajes que se obtienen del Store And Forward.
- Bytes que se obtienen del Store And Forward.
- Mensajes almacenados en el Store And Forward.
- Bytes almacenados en el Store And Forward.

Para incrementar o decrementar dichos contadores se utilizan métodos públicos que son llamados desde las clases en que se reciben, procesan o envían datos. En general, la orden de modificar un contador es dada desde las instancias de HiloS_TX correspondientes a cada trama. De esta manera, cuando se recibe una trama se sabrá de

que tipo de mensaje se trata al procesarla, y por lo tanto, es en dicha instancia que se decide por ejemplo, incrementar la cantidad de mensajes y bytes de mensajes de texto, o incrementar la cantidad de mensajes y bytes almacenados en el Store And Forward, o los contadores que corresponda.

El “esqueleto” de la clase BdD se utiliza para manejar el ingreso de estadísticas en la Base de Datos. Para esto, el objeto que instancia esta clase permanece en un estado de loop infinito que realiza las siguientes funciones:

- Espera 15 minutos, intervalo de tiempo durante el que se recogen las estadísticas en el uServer (modificando los contadores).
- Llama al método “ingresarEst” para guardar en las tablas correspondientes de la Base de Datos el valor de los contadores en ese momento.
- Resetea los contadores correspondientes (todos, menos los que cuentan los mensajes y bytes almacenados en la tabla de Store and Forward).

➤ Manejo de mensajes Store and Forward

Dado que para utilizar la funcionalidad Store and Forward no se necesita ningún procesamiento adicional, la clase BdD únicamente cuenta con métodos para ingresar, consultar y eliminar los registros de la tabla correspondientes en la Base de Datos. Estos métodos son llamados desde las instancias de HiloS_TX cada vez que el Servidor recibe un mensaje de texto con formato SnF o un mensaje de notificación de conexión de un nuevo Terminal.

➤ Manejo de Gráficas.

Como se verá más adelante, el nodo uServer cuenta con la posibilidad de graficar en tiempo real el tráfico que circula a través de él. Para realizar esta función correctamente, los contadores necesarios son también implementados en la clase BdD, luego son consultados periódicamente por la clase encargada de realizar las gráficas.

4.4 uDataBase

4.4.1 Introducción

Para alcanzar algunos de los objetivos propuestos en el proyecto, se encontraron limitaciones en las herramientas de desarrollo utilizadas hasta el momento, por lo cual se debió recurrir a la búsqueda de nuevas posibilidades.

Entre esas limitaciones apareció la necesidad de trabajar con extensas tablas que manejen gran variedad de datos relacionados, lo que llevó a pensar en la posibilidad de la utilización de un sistema de base de datos.

Analizando esta opción se vio que además de cumplir con el objetivo buscado, también permitiría lograr una mejor eficiencia de la red. De esta forma se liberaría de responsabilidad al nodo que debe utilizar esos datos, delegando el manejo de estos a la Base de Datos y logrando por ende una menor complejidad en el desarrollo del software de dicho nodo, pero sobretodo una menor exigencia de hardware de este.

Desde un punto de vista más global, la utilización de una base de datos permitiría concentrar todas las tablas de datos necesarias no solo para la utilización de un nodo en particular, sino también para todos los nodos de la red que la requieran.

Por todo lo mencionado anteriormente se decidió implementar la base de datos como un nodo independiente, al cual el resto de los nodos de la red recurrirán para realizar sus gestiones correspondientes, ya sea ingresar, consultar o borrar los datos necesarios.

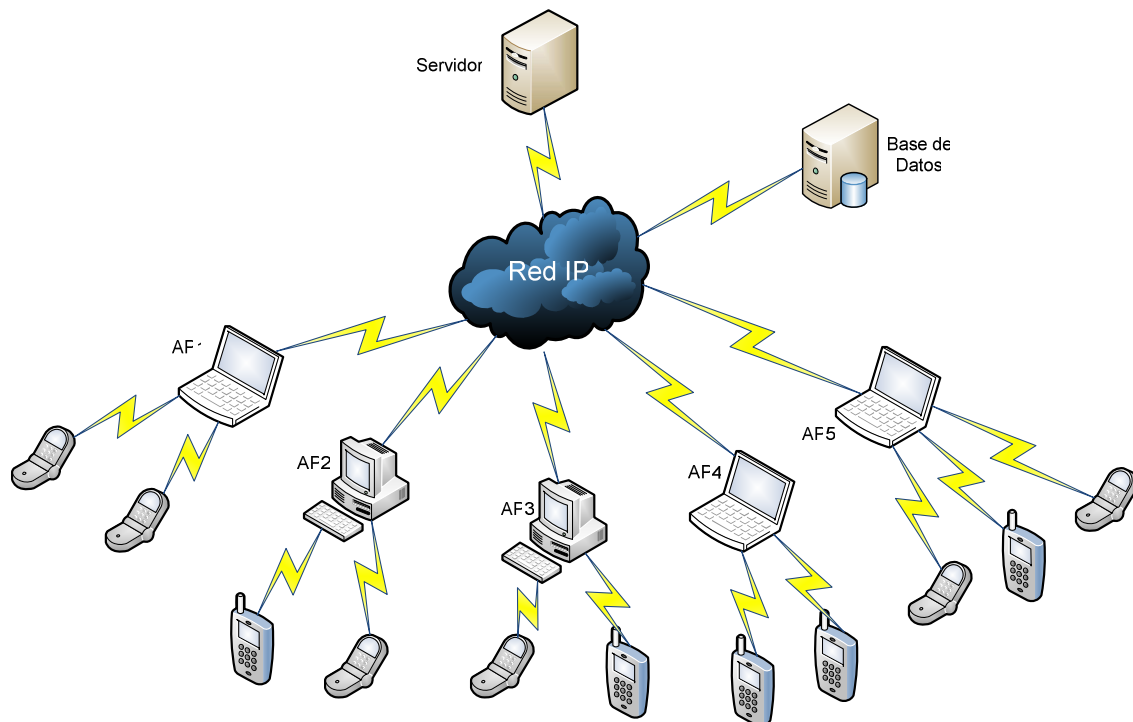


Figura 4.6 - Diagrama de la red con la Base de Datos como nodo independiente

4.4.2 Elección de la Base de Datos y adaptación al sistema

Tomada la decisión, el siguiente punto fue decidir que tipo de base de datos utilizar. Por su flexibilidad, potencia y variedad de operaciones se decidió que sea SQL (ver anexo) el lenguaje de acceso a base de datos utilizado. Además el hecho de ser uno de los lenguajes más utilizados hoy en día facilitaría la obtención de información para capacitarse sobre el uso del mismo.

Con el lenguaje de acceso a base de datos ya definido, solo restó elegir un sistema de gestión que soporte dicho lenguaje. Para esto fue necesario establecer ciertas condiciones tanto desde el punto de vista del funcionamiento, la compatibilidad y el costo.

Fundamentalmente estas condiciones se basaron en:

- Debería tratarse de un software libre, no solo para reducir costos sino también para continuar con el uso exclusivo de herramientas de software libre que se venían utilizando para el desarrollo del proyecto.

- Ser accesible desde bloques de código en lenguajes de programación (en nuestro caso Java) para así tener la posibilidad de acoplarlo con el resto del software y de esta manera acceder a ella desde cualquiera de los nodos de la red.
- Poder manejar varios tipos de variables, ya que los datos que contendrán sus tablas no serán todos de un mismo tipo (la justificación de este requisito así como también del siguiente, serán explicados mas adelante en el presente capítulo).
- Manejar texto de una extensión considerable.
- Tener la posibilidad de lograr múltiples accesos simultáneos a una misma tabla, para así permitir el acceso de varios nodos en simultáneo y evitar retardos, datos a destiempo u otros posibles inconvenientes.
- Ser compatible con el sistema operativo Linux (usado para el desarrollo del software del proyecto), y -de ser posible- con otros sistemas operativos (no cerrando definitivamente la posibilidad de extender el proyecto a otros sistemas).

Analizando estos factores, las diferentes opciones de sistemas de gestión de bases de datos se fueron reduciendo hasta que entre restantes se terminó optando por el sistema PostgreSQL.

PostgreSQL es un sistema de gestión de bases de datos relacional orientada a objetos, publicado bajo la licencia BSD. Se trata de un proyecto “open source”, el cual no es dirigido por una sola compañía sino por una comunidad de desarrolladores y organizaciones comerciales que trabajan para su desarrollo.

El cumplimiento por parte de PostgreSQL de las condiciones impuestas, se detalla a continuación:

- Como se mencionó anteriormente, se trata de un software libre de tipo “open source”.
- Es accesible desde varios lenguajes de programación (entre ellos C, C++, Java, Python, Ruby, etc.) utilizando la potencia propia de cada lenguaje. Esto permite realizar desde operaciones básicas hasta complejas operaciones de programación orientada a objetos o programación funcional.
- Maneja todos los tipos de variables tradicionales de los lenguajes de programación, distinguiendo además tipos mas específicos como Arrays, Direcciones IP, figuras geométricas, variables de formato Fecha/Hora (en formato ANSI), etc. identificando también orden de precedencia en aquellos tipos en los que tiene sentido.
- Maneja texto de largo ilimitado.

- Mediante un sistema llamado MVCC (Acceso concurrente multiversión, por sus siglas en inglés) permite que, mientras un proceso escribe en una tabla, otros accedan a la misma tabla sin necesidad de bloqueos. En otros sistemas de bases de datos, la tabla a la que se le está ejecutando un proceso se bloquea (o en el mejor de los casos se bloquean ciertas filas de la tabla) frente a otros procesos que la requieran.
- PostgreSQL se encuentra disponible en versiones para los sistemas operativos FreeBSD, Linux, Mac OS X, Solaris y Windows.

En el caso de este proyecto, el acceso a la base de datos desde el lenguaje de programación utilizado (Java) se realiza utilizando una librería llamada JDBC (Java Database Connectivity).

Esta API permite la ejecución de operaciones desde el lenguaje Java independientemente del sistema de gestión de bases de datos que se utilice, siempre y cuando este sistema utilice el dialecto SQL. Dicho de otra manera, JDBC permitirá, entre otras cosas, utilizar el lenguaje SQL embebido dentro de Java.

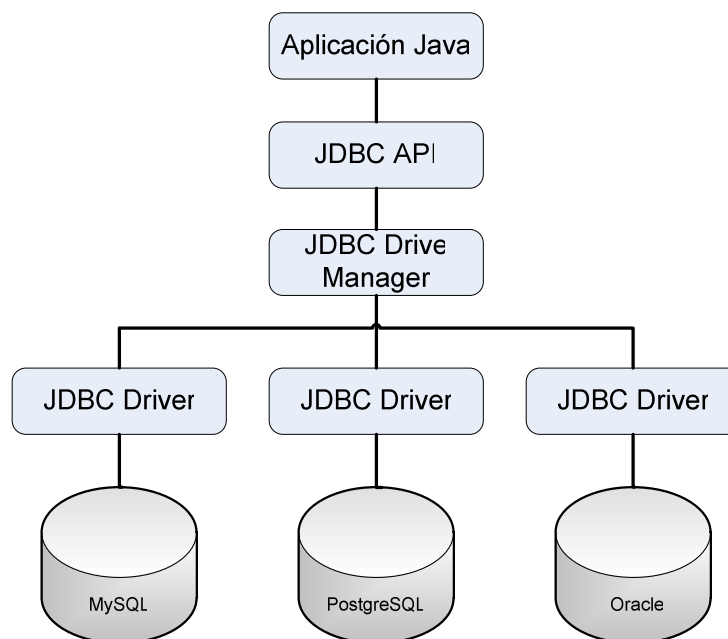


Figura 4.7 - Diagrama de la estructura de JDBC

JDBC cuenta con una colección de interfaces Java y métodos de gestión para manejadores de conexión hacia cada modelo específico de base de datos. Estos manejadores de conexión son conjuntos de clases que implementan las interfaces Java y

que utilizan los métodos de registro para declarar los tipos de localizadores de bases de datos que pueden manejar.

Con un localizador de base de datos (URL) provisto por un método de registro y otros parámetros de conexión (como pueden ser usuario y contraseña) se establece la conexión entre el programa desarrollado en Java y la Base de Datos. Luego, por medio de la utilización de las interfaces provistas por JDBC, se procederá al embebido de los comandos SQL dentro del código Java, lográndose a partir de esto la interacción entre ambos lenguajes. Asimismo, JDBC provee las interfaces necesarias para la introducción al programa (desarrollado en Java) de los datos obtenidos a partir de las consultas realizadas.

En consecuencia, a partir de todo lo mencionado hasta aquí, quedan establecidas las herramientas necesarias para cumplir el objetivo del manejo de grandes tablas de datos relacionados, propuesto al principio del presente capítulo, quedando además definidos los mecanismos, ya sea para ingresar datos, borrar o realizar cualquier tipo de consulta en dichas tablas, desde la plataforma en donde se desarrollaron el resto de los nodos de la red.

4.4.3 Usos de la Base de Datos

4.4.3.1 Uso de la Base de Datos en Store and Forward

Concretamente en este proyecto, la Base de Datos fue diseñada para llevar a cabo parte del proceso implicado en dos de las funcionalidades de la red Umaguma.

Uno de estos usos, fue la implementación de un sistema que permita la realización de Store and Forward de mensajes. Store and Forward (en español, Almacenamiento y Envío) es una conocida técnica de telecomunicaciones que consiste en retener la información enviada en una estación intermedia entre origen y destino, para luego ser enviada desde allí y seguir su curso, basándose en ciertas reglas establecidas previamente. Este nodo o estación intermedia, almacena la información recibida desde otros nodos y siguiendo las políticas fijadas, determinará hacia donde y en que momento la reenviará.

Específicamente en lo que respecta a este proyecto, el sistema de mensajería Store and Forward, es implementado para brindar al usuario la posibilidad de enviar mensajes a destinos cuyos dispositivos no se encuentran conectados en la red en el momento en que desea realizar en envío. Esta técnica posibilitará almacenar dichos mensajes en cierto nodo y poder ser enviados a destino una vez que el destinatario conecte su dispositivo en la red.

Al momento de analizar cómo desarrollar esta funcionalidad, en un principio se pensó en que quien almacene estos mensajes sea el propio Servidor, pero pronto se vio que esto podría producir algunas consecuencias negativas como:

- Mayor exigencia de hardware para el servidor
- Posibles complicaciones en la agilidad de la red, ya que mientras el servidor realiza búsquedas en tablas no podría atender solicitudes de otros nodos
- Mayores retrasos en general, que perjudican la eficiencia de la red

Fue por esto que surgió la necesidad de realizar una estación separada donde se almacenen los mensajes junto con otros datos necesarios para permitir la implementación de esta funcionalidad.

Cuando un dispositivo conectado a la red envía un mensaje a otro que no se encuentra conectado, el mensaje sale del Terminal acompañado de los datos del destinatario (dirección MAC y Friendly name) y una dirección especial que lo identifica como mensaje para Store and Forward, la cual hará que tanto su uAP asociado como el Servidor sepan que deben hacer él.

El mensaje llega al uAP y este, al verificar su dirección especial, lo envía directamente al Servidor quien, comprobando también la dirección especial, levantará una conexión con la Base de Datos e ingresará (mediante un comando SQL) el mensaje junto con los datos necesarios, en una tabla de la Base creada con este fin.

La tabla que se menciona está conformada por los siguientes campos:

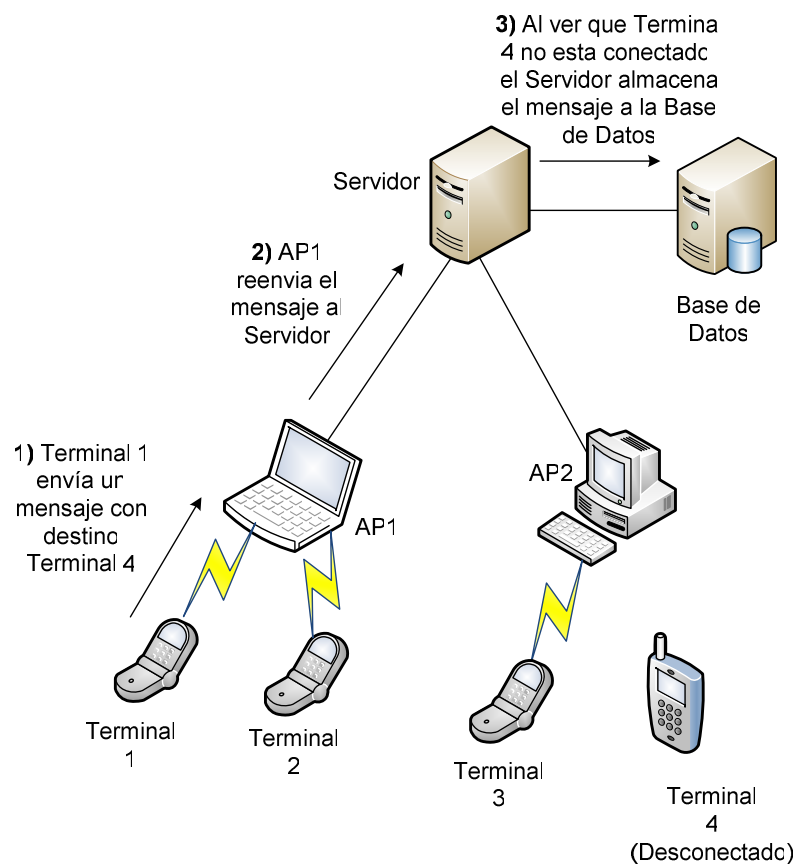
Fecha y Hora	Dirección MAC del móvil de origen	Dirección MAC del móvil de destino	Friendly Name del móvil de origen	Mensaje a enviar
--------------	-----------------------------------	------------------------------------	-----------------------------------	------------------

Una vez ingresados los datos en los respectivos campos de la tabla, estos permanecerán almacenados en ella.

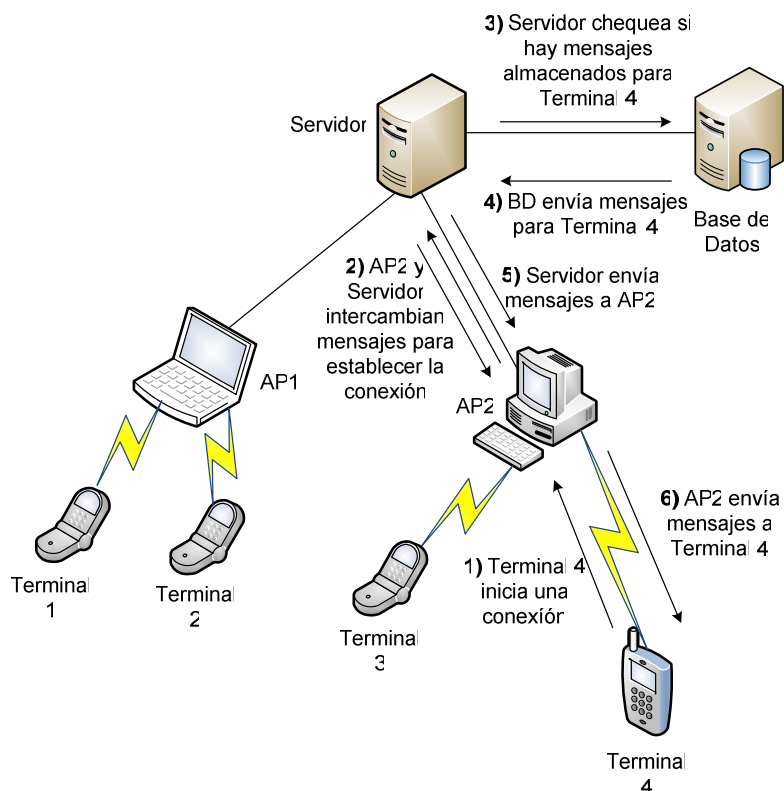
Cuando un nuevo Terminal ingresa la red, además de todo el intercambio de mensajes de control que ocurre para poder establecer la sesión, se debe agregar un paso más. El servidor realiza una consulta a la Base de Datos para comprobar si existen mensajes de Store and Forward almacenados para dicho móvil. Esta consulta se hace por medio de la dirección MAC del Terminal, recorriendo la tabla y quedándose con aquellas filas en las que dicha dirección coincida con la ingresada en el campo "Dirección MAC del móvil de destino.

Una vez ubicadas las filas, el Servidor las recibe e inmediatamente ejecuta un comando SQL en el que solicita que se borren de la Base de Datos todas aquellas líneas que contenían los mensajes recientemente seleccionados. A continuación se realiza un proceso de desconexión del servidor con la Base de Datos.

Luego, el Servidor preparará a partir las líneas recibidas, cada una de las tramas a enviar con la estructura del protocolo diseñado para la red. Las tramas son enviadas al uAP correspondiente y por medio de este al Terminal destino.



Situación 1 – Envío de mensaje a Terminal desconectado



Situación 2 – Terminal se conecta a la red y recibe mensajes de *Store and Forward*

Cada uno de los mensajes de Store and Forward recibidos irán acompañados de la hora y fecha en que fueron enviados por el uCel de origen, junto con su respectivo Friendly Name, para poder ser identificado por el destinatario.

4.4.3.2 Uso de la Base de Datos en Sistema de Estadísticas

La otra funcionalidad para la que fue necesaria la Base de Datos fue el desarrollo de un sistema de estadísticas de la red.

Este sistema -que más adelante en el presente documento se explicará con mayor detalle- permitirá monitorear el tráfico cursado por los nodos de la red, distinguiendo además, entre los tipos de mensajes que se transmiten (mensajes de texto, imágenes y voz). Esto dará, entre otras cosas, nociones de cuales son los nodos más congestionados y para qué tipo de mensajes se usan con mayor frecuencia, para -de ser necesario- plantear diferentes configuraciones en cuanto a la distribución de los nodos.

Al igual que en el caso de los mensajes Store and Forward, en un principio se planteó la posibilidad de que cada nodo guarde -él mismo- un registro del tráfico que cursa.

Análogamente a lo concluido para el caso anterior, se observó que esto resultaba poco eficiente argumentando las mismas razones de mayor exigencia de hardware y mayores retrasos en las funciones del servidor.

Por tal motivo se creyó necesario que el almacenamiento de los datos se hiciera dentro de tablas especialmente creadas para este uso en la Base de Datos. Esto además traería ventajas adicionales, por ejemplo:

- Se tendría un nodo independiente donde se concentrasen todas las estadísticas de todos los nodos de la red en lugar de tener que ir a cada uno de ellos por separado para conseguir sus datos.
- En el caso de querer obtener estadísticas de uno o varios nodos de la red, se ganaría rapidez, ya que la información se encontraría almacenada en una tabla común, lo cual acorta la búsqueda.
- Se da la posibilidad de un acceso remoto a la información desde cualquier computadora con acceso a la red que cuente con el software de gestión (que se detallará en capítulos posteriores) y las respectivas direcciones IP, puerto y contraseña de la Base de Datos.

El sistema de gestión de estadísticas se explicará en este documento en un capítulo posterior, analizado desde el punto de vista del usuario. A continuación se explicará el funcionamiento del sistema desde el punto de vista de la recolección de datos.

En lo que respecta a la red desarrollada en este proyecto, y como se explicó en capítulos anteriores, se distinguen desde el punto de vista del enrutamiento de mensajes dos tipos de nodos: Access Points (uAPs) y Servidor (uServer). Es de ellos que nos interesa tener un control de estadísticas de tráfico, ya que son quienes están directamente involucrados en el tema.

Se crearon en la Base de Datos dos tablas especialmente dedicadas al almacenamiento de datos estadísticos, una para cada tipo de nodo. Estos datos son necesarios para que posteriormente el software de gestión pueda presentarle las cifras finales al usuario.

El criterio que se empleó para la recolección de datos es similar al utilizado en los sistemas de estadísticas de las redes de telefonía celular. Consiste en un reporte de datos en intervalos de 15 minutos. Esto significó implementar dentro del software (uAP y uServidor), diferentes variables que actúen como contadores y que se incrementen cada vez que un mensaje es cursado por ellos. Los contadores acumularán información sobre la cantidad de mensajes conmutados y el tamaño (en bytes) de cada tipo de mensaje que pasa por el nodo.

Al cumplirse un lapso de 15 minutos, cada nodo realiza una conexión con la base de datos donde reporta los valores acumulados por sus contadores en las tablas correspondientes, ingresándolas mediante un comando SQL. A continuación lleva sus variables a cero y posteriormente solicita un pedido de desconexión con la base de datos.

Tal como se mencionó anteriormente, para cada tipo de nodo se creó una tabla diferente. La razón por la cual se realiza de esta manera es que los nodos tienen funciones diferentes y por lo tanto los parámetros que interesa analizar son distintos.

La tabla que recibe los contadores del Servidor posee los siguientes campos:

Fecha y Hora	Mensajes E/S de Texto	Mensajes E/S de Imágenes	Mensajes E/S de Voz	Mensajes S&F Entrantes	Mensajes S&F Salientes	Mensajes S&F Acumulados
Bytes E/S de mensajes de Texto	Bytes E/S de mensajes de Imágenes	Bytes E/S de mensajes de Voz	Bytes de mensajes S&F Entrantes	Bytes de mensajes S&F Salientes	Bytes de mensajes S&F Acumulados	

Análogamente, los campos llenados por los contadores aportados por los APs son los siguientes:

Fecha y Hora	Dirección de AP	Mensajes E/S de Texto	Mensajes E/S de Imágenes	Mensajes E/S de Voz
Bytes E/S de mensajes de Texto	Bytes E/S de mensajes de Imágenes	Bytes E/S de mensajes de Voz	Bytes Locales	Bytes hacia el Servidor

El criterio con el que se definen cada uno de estos campos será explicado (para cada una de los contadores) en el capítulo dedicado especialmente al sistema de gestión de estadísticas (Capítulo 6) que se verá mas adelante.

Gracias al MVCC (Acceso concurrente multiversión, en inglés) que, como se explicó anteriormente posee PostgreSQL, se puede garantizar que no ocurrirán eventuales inconvenientes en el caso de que se realice una consulta desde el software de gestión a la vez que en la misma tabla se están ingresando datos por medio de alguno de los nodos, lo que asegura una mayor eficiencia del sistema desarrollado haciéndolo además mas robusto frente a las exigencias del usuario.

Capítulo 5: Ejemplos de uso

En este capítulo se detallan los diferentes eventos que se dan entre los actores de la red en determinados casos de uso de la misma. En primer lugar se explicará en que momentos se dan estos eventos, para luego enumerar en orden cronológico los distintos mensajes intercambiados entre los componentes de la red.

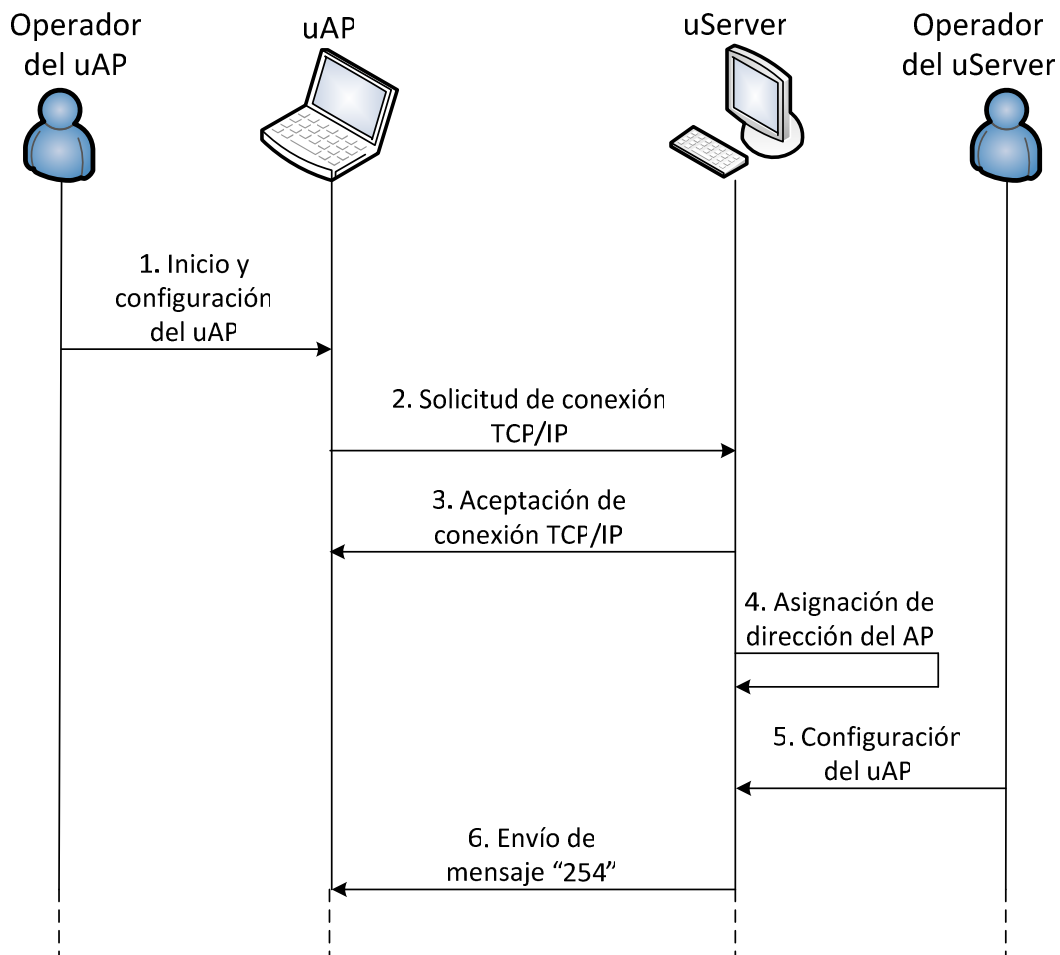
5.1 Conexión de un uAP al uServer

Si la red desplegada utiliza uServer, al iniciar el uAP éste intenta establecer una conexión TCP/IP con el Servidor, lo que lleva a un intercambio de mensajes entre ambos, necesario para un correcto funcionamiento del uAP dentro de la red. Siguiendo un orden cronológico, los distintos pasos en el establecimiento de la conexión son:

1. Seteo del uAP para trabajar conectado al uServer. Para esto se configuran los siguientes parámetros desde la Interfaz gráfica del AP:
 - Dirección IP del uServer.
 - Puerto en que el uServer se encuentra a la espera de nuevas conexiones AP.
 - Dirección IP y puerto en que se encuentra la uDataBase.
2. El uAP inicia una conexión TCP/IP hacia el uServer.
3. Si el uServer se encuentra funcionando correctamente, aceptará la conexión del uAP, ya que este se encuentra constantemente escuchando y aceptando conexiones por el puerto seleccionado al iniciar dicho nodo.
4. Una vez establecida la conexión, el uServer asigna automáticamente una dirección Umaguma para el nuevo uAP.
5. Se ingresan manualmente (desde la Interfaz gráfica del Servidor) las siguientes propiedades del uAP:
 - Nombre del AP
 - Zona a la cual pertenecerá el AP

- Contraseña de 6 dígitos para autorizar las conexiones de Terminales al nuevo AP (en caso de que se haya seteado el uso de contraseñas para dicho AP).
6. Recién llegado este punto se considera establecida la conexión con el uAP para el uServer, por lo que se procede al envío de la trama “254” hacia el AP, la cual contiene la dirección Umaguma del nuevo nodo y la contraseña que deberá utilizar para validar los nuevos usuarios.

La siguiente figura ilustra la secuencia descrita.



5.2 Pérdida de conexión entre un uAP y el uServer

Cuando se pierde la conexión entre un uAP y el uServer, el Servidor debe notificar al resto de los elementos conectados a la red de la desconexión de los Terminales conectados al AP desconectado. Además, este Punto de Acceso debe notificar a los Terminales con los que mantiene conexión directa que ya no mantienen conexión con los Terminales del resto de la red.

El proceso que siguen los elementos de la red, a partir de que el uAP involucrado y el uServer se percatan de la desconexión entre ambos, es el siguiente:

1. El uServer notifica al resto de los uAP que quedaron conectados a la red de la desconexión del AP. Para esto, envía a todos los uAP un mensaje especial “253” por cada uno de los Terminales que se encontraban conectados en el uAP que se desconectó.

En paralelo, el uAP desconectado crea una nueva tabla de usuarios conectados conteniendo solamente a los Terminales conectados a dicho AP, para poder notificar a los usuarios en cuestión que ya no mantienen conexión con los usuarios del resto de la red.

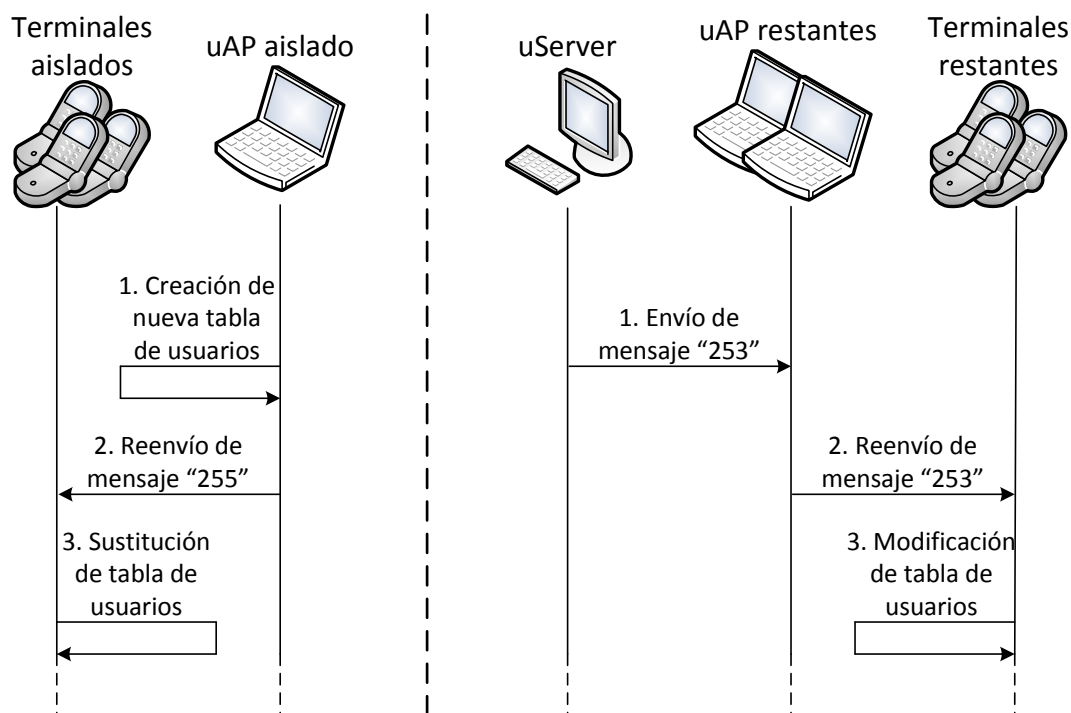
2. Los uAP que continúan conectados a la RED, reciben los mensajes “253” enviados por el uServer y los reenvían a todos los Terminales con los que mantienen conexión.

En paralelo, el uAP con el que se perdió conexión envía la tabla de usuarios creada en el punto anterior, mediante el mensaje especial “255”, a todos los Terminales con los que mantiene conexión.

3. Los Terminales que mantienen conexión con el uServer (a través del correspondiente uAP) reciben el mensaje “253” y actualizan sus tablas de usuarios conectados eliminando de ella a los usuarios conectados al uAP con el que se perdió conexión.

En paralelo, los Terminales conectados al uAP con el que se perdió conexión reciben el mensaje “255” y sustituyen la tabla de usuarios por la nueva, que solo contiene a los Terminales que están conectados al nodo en cuestión que quedó trabajando en modo Stand-Alone.

Estas secuencias en paralelo que se describieron en los pasos anteriores, se enseñan a continuación.



5.3 Conexión de un celular a un AP

Un Terminal que cuente con el software Umaguma, podrá intentar conectarse a todo uAP que mantenga menos de 7 conexiones con otros Terminales (máximo impuesto por el estándar Bluetooth). Cuando esto sucede, se produce un intercambio de mensajes de control entre los elementos involucrados, y dependiendo de las políticas configuradas en la red (ver Capítulo 7), se establecerá o no la conexión entre el uCel y el uAP.

A continuación se describen los eventos realizados por ambos componentes, dependiendo de las políticas y configuraciones de la red.

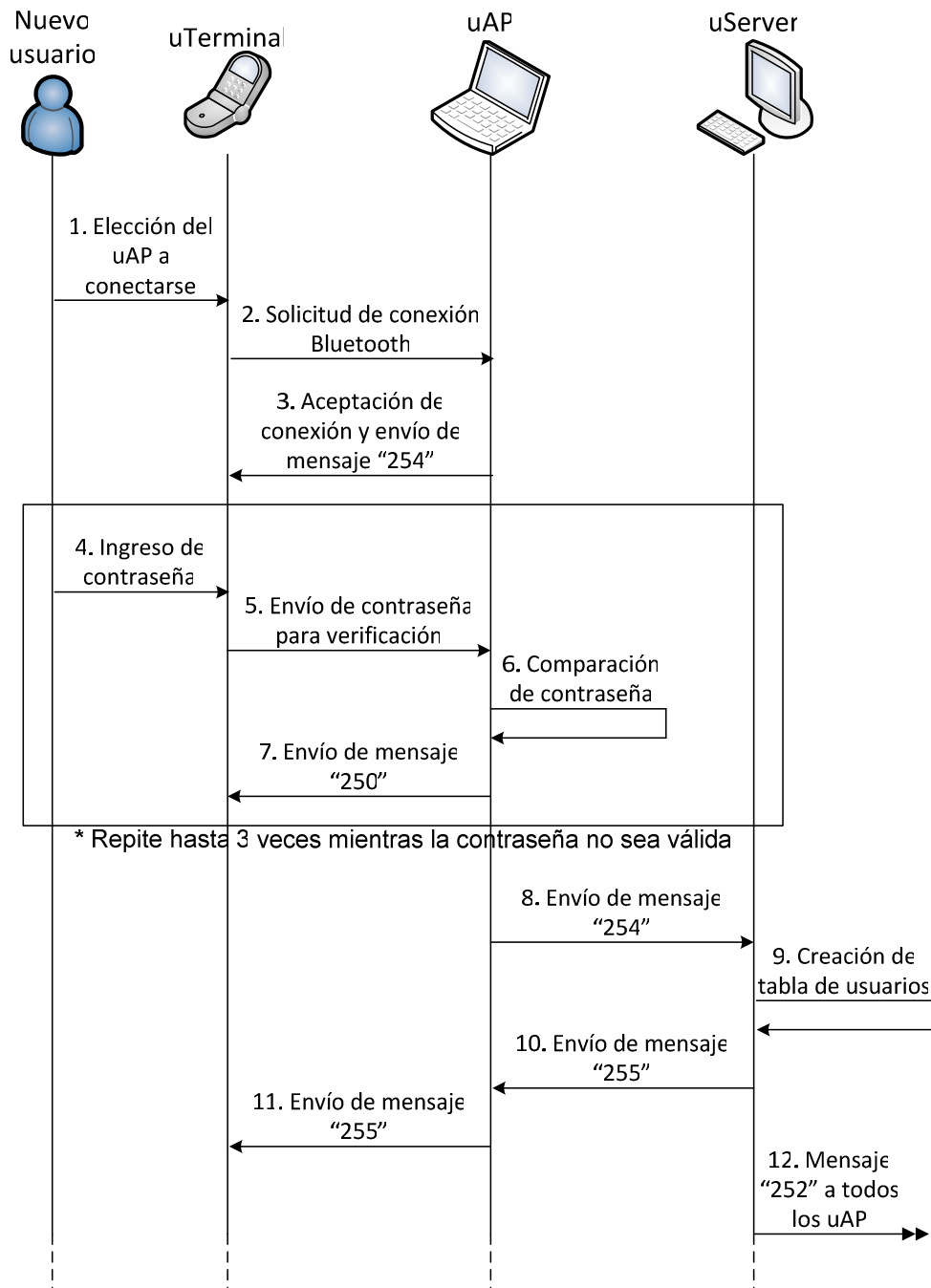
- **Conexión exitosa a un uAP con contraseña:**

1. El usuario elige desde el uCel el uAP al que desea conectarse mediante una lista incluida en el software umaguma (de esta manera se evita realizar la búsqueda de dispositivos y servicios que se mencionará más adelante).
2. Luego se intenta establecer la conexión bluetooth con el uAP utilizando la MAC de dicho nodo.
3. Si el uAP aún no ha llegado al máximo de conexiones con uCel's, aceptará la conexión del Terminal, y le enviará al mismo el mensaje "254" que contendrá una bandera indicando que el AP requiere contraseña, su correspondiente dirección umaguma y la contraseña (como se mencionó en el Capítulo 3).
4. El nuevo usuario ingresa desde el uCel la contraseña.
5. El uCel envía al uAP la contraseña ingresada para su verificación.
6. El uAP compara la contraseña ingresada por el usuario con la correcta.
7. Envío de mensaje especial "250" desde el uAP al uCel indicando si la contraseña ingresada es válida o no.

Los pasos 4 a 7 pueden repetirse hasta 3 veces en caso de que el usuario ingrese mal la contraseña.

8. Una vez que el Terminal recibió su dirección umaguma y se ingresó correctamente la contraseña, el uAP le envía al uServer un mensaje especial "254" notificando la conexión del nuevo Terminal, el cual incluye el friendly name, dirección umaguma y MAC del nuevo usuario.
9. Luego de recibir la notificación de conexión del nuevo uCel, el uServer crea una tabla con los friendly names y las direcciones umaguma de todos los usuarios conectados a la red en ese momento.
10. El uServer envía la tabla generada en el paso anterior al uAP en que se encuentra conectado el nuevo Terminal mediante el mensaje especial "255".
11. El uAP reenvía la tabla recibida al nuevo Terminal mediante un mensaje "255" entre dichos elementos.
12. Por último, el uServer envía a todos los uAP una notificación de conexión del nuevo Terminal mediante el mensaje especial "252", donde se incluye su friendly name y dirección umaguma, para que los AP distribuyan la notificación a todos los Terminales de la red.

A continuación se ilustra el proceso de conexión de un uCel.



- **Conexión fallida con un uAP con contraseña (por contraseña incorrecta):**

El proceso en éste caso es idéntico al anterior hasta el punto 7.
Luego de repetir 3 veces los pasos 4 a 7 sin que el usuario ingrese correctamente la contraseña, el uAP finaliza la conexión con el uCel.

- **Conexión a un uAP sin contraseña:**

Si el uAP fue configurado de manera de no utilizar contraseña para restringir la conexión de Terminales, el proceso de conexión es igual al ilustrado en el caso con contraseña, sacando los pasos 4 a 7.

- **Conexión a un uAP funcionando en modo Stand-Alone:**

Si el uAP está funcionando en el modo Stand-Alone, se observan las siguientes modificaciones en el proceso de conexión descrito:

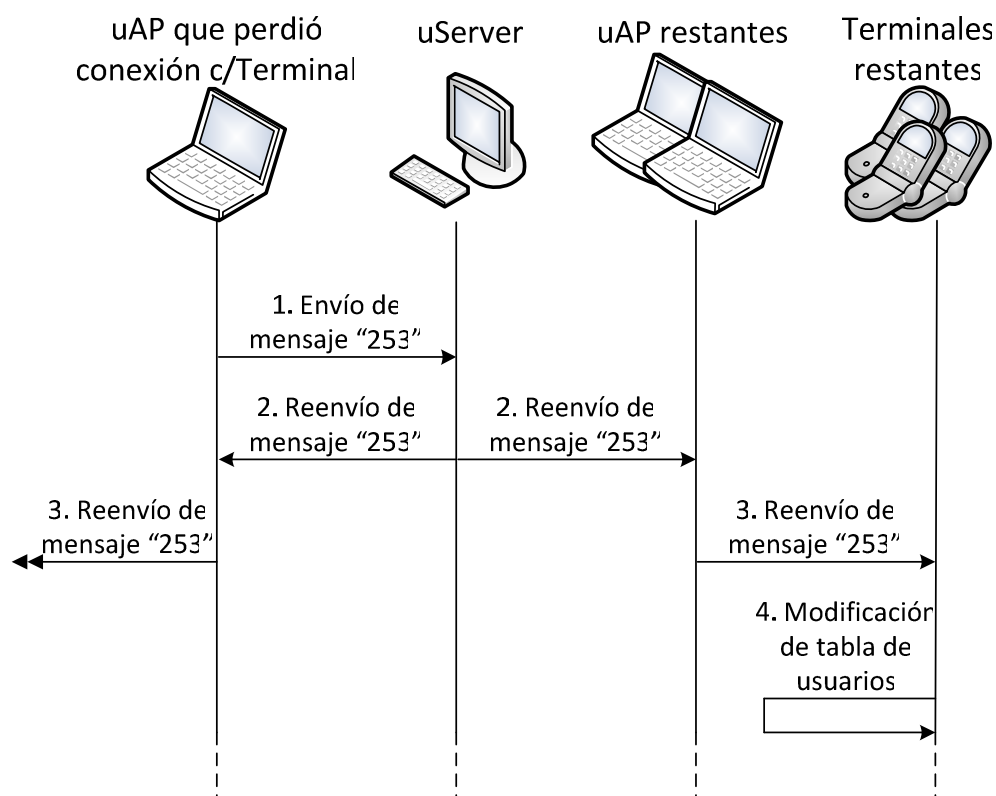
- Se eliminan los pasos 4 a 7, por no utilizar contraseña en este modo de funcionamiento.
- Se eliminan los pasos 8 a 10, ya que no existe uServer para notificar del nuevo usuario conectado, y la tabla para enviar en el mensaje especial “255” es generada por el uAP mismo.
- Se elimina el paso 12, ya que no hay otros uAP conectados a la red para notificarles del nuevo usuario conectado.

5.4 Perdida de conexión con un Terminal

Cuando se pierde la conexión entre un uCel y el uAP al que se encontraba conectado, se llevan a cabo las siguientes acciones para notificar al resto de los elementos de la red sobre la desconexión:

1. El uAP que perdió conexión con el Terminal envía un mensaje “253” con la dirección umaguma del Terminal desconectado al uServer.

2. El uServer reenvía el mensaje "253" a todos los AP conectados, incluyendo al AP en que estaba conectado el Terminal con el que se perdió conexión.
3. Los uAP reenvían el mensaje "253" a todos los Terminales con los que mantienen conexión.
4. Los Terminales reciben el mensaje "253", y modifican sus listas de usuarios eliminando al Terminal desconectado de la misma.



5.5 Envío de mensaje de texto desde un uCel

Cualquier usuario de Umaguma puede enviar un mensaje de texto a otro usuario de la red siempre que las políticas de visibilidad de los AP involucrados lo permitan (como se verá en el Capítulo 7).

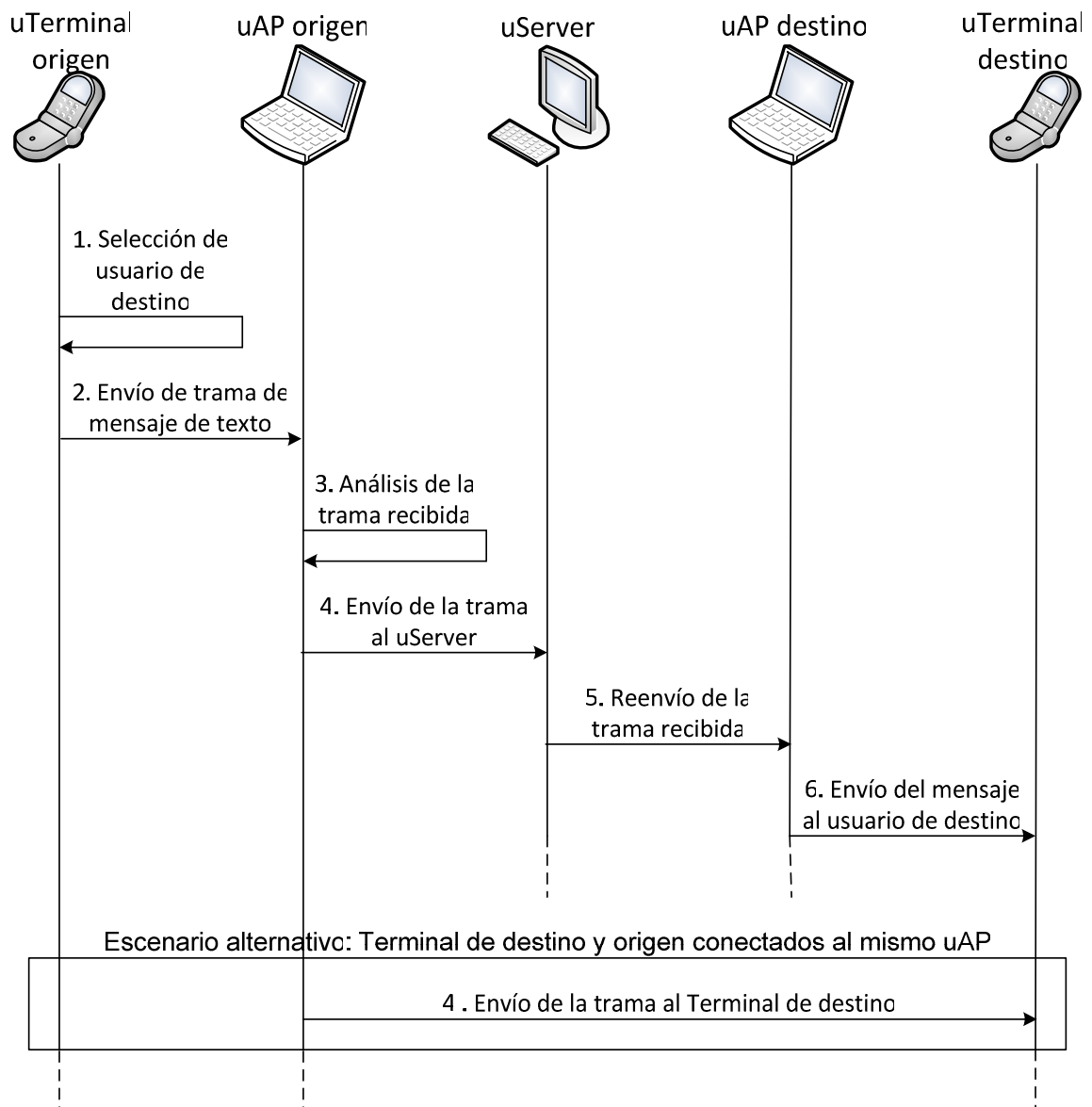
En caso de que la comunicación entre el usuario remitente y el usuario destinatario esté permitida, el proceso de envío del mensaje de texto es el siguiente:

- **Envío de mensaje entre usuarios conectados a distintos Puntos de Acceso:**

1. El usuario remitente selecciona a partir de una lista almacenada en el uTerminal el usuario al cual desea enviarle el mensaje.
2. El uCel de origen genera una trama conteniendo la dirección umaguma del usuario de destino, la dirección Umaguma del usuario de origen y el mensaje a transmitir, y lo envía hacia el uAP al cual se encuentra conectado.
3. El uAP al que se encuentra conectado el usuario que envía el mensaje recibe la trama enviada por el uCel y analiza la dirección Umaguma del Terminal de destino.
4. Del análisis de la trama recibida, el uAP deduce que el Terminal de destino se encuentra conectado a otro AP de la red, por lo que envía el mensaje al uServer.
5. El uServer vuelve a analizar la dirección umaguma de destino del mensaje y lo reenvía hacia el uAP en que se encuentra conectado el uCel de destino.
6. Por último, el uAP mencionado en el punto anterior envía el mensaje al Terminal de destino, obteniendo la dirección de dicho dispositivo de la trama en cuestión.

- **Envío de mensaje entre dos Terminales conectados al mismo uAP:**

En este caso, el procedimiento es similar al descrito en el caso anterior, con la diferencia de que en el paso 4, cuando el uAP analiza la trama enviada por el usuario remitente, deduce que el Terminal de destino se encuentra conectado a éste mismo AP, y por lo tanto, en vez de enviar el mensaje al uServer para que continúe el enrutamiento, envía el mensaje directamente al Terminal de destino. Las acciones realizadas en ambos casos por cada componente se ilustran a continuación.



5.6 Envío de Imágenes desde un Terminal

Al igual que para los mensajes de texto, cualquier usuario podrá mandar una imagen capturada con la cámara fotográfica de su Terminal a otro usuario siempre que las políticas de la red lo permitan.

Como se explicó en el Capítulo 3, la trama generada por el Terminal para enviar imágenes es muy similar a la generada para enviar mensajes de texto. La diferencia, aparte de la sustitución del mensaje de texto por la imagen, es la adición de 3 bytes al

principio de la trama para indicar que corresponde a una imagen (mensaje especial "247").

Es por lo mencionado en el párrafo anterior, que el proceso que atraviesa una trama utilizada para enviar una imagen desde un Terminal a otro, es similar al que atraviesa una trama que contiene un mensaje de texto. La única diferencia, es que cada nodo debe realizar un procesamiento adicional debido a los 3 bytes extras que contiene la trama en cuestión.

5.7 Envío de archivos de voz desde un Terminal

Nuevamente, siempre que las políticas de la RED lo permitan, un Terminal podrá enviar un archivo de voz hacia otro Terminal que también esté conectado a Umaguma.

En este caso, las tramas generadas son idénticas a las generadas para enviar una imagen (a excepción de la carga útil), por lo que cada componente de la red involucrado en el envío se comportará de la misma manera que en el punto anterior.

5.8 Store And Forward

Un usuario conectado a la red Umaguma puede enviar un mensaje de texto a cualquier uCel, que se encuentre conectado a la red o no, a partir de la MAC del Terminal de destino. Para esto, Umaguma utiliza una Base de Datos que funciona como Store And Forward en caso de que el Terminal de destino no se encuentre conectado a la RED en el momento en que se envía el mensaje.

A continuación se describen las acciones realizadas por cada uno de los elementos de la red cuando un usuario manda un mensaje a un Terminal identificándolo por su MAC, y cuando un usuario que recién se conecta a Umaguma recibe un mensaje desde el Store And Forward.

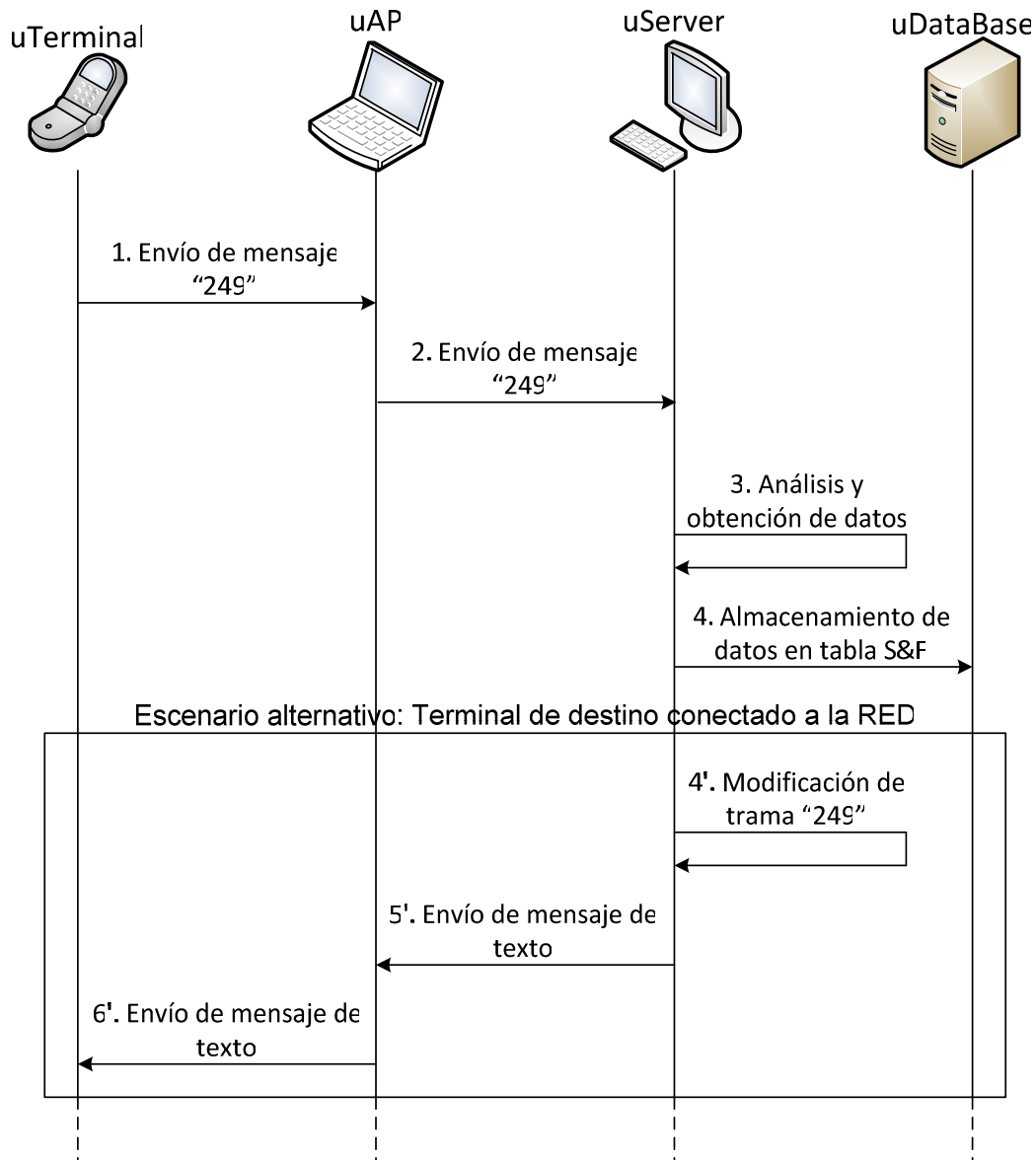
5.8.1 Envío de un mensaje utilizando MAC de destino

- **Caso en que el usuario de destino no está conectado a la red:**
 1. El uCel genera una trama conteniendo la MAC del Terminal de destino, la dirección Umaguma de origen y el mensaje a enviar, y la envía al uAP al que se encuentra conectado a través del mensaje especial "249".
 2. El uAP reenvía la trama al uServer.
 3. El uServer analiza la MAC del Terminal de destino, para verificar que el Terminal con dicha MAC no se encuentra conectado a umaguma, y obtiene, a partir de la trama recibida y la información almacenada en dicho nodo sobre los Terminales conectados, los parámetros necesarios para almacenar en la Base de Datos (friendly name y MAC del Terminal de origen, MAC del Terminal de destino, fecha y hora en que se recibió el mensaje S&F y el mensaje a transmitir).
 4. El Servidor establece una conexión TCP/IP con la Base de Datos y almacena en la tabla correspondiente los parámetros descritos en el punto anterior.

- **Caso en que el usuario de destino se encuentra conectado a la red:**

En caso de que el Terminal de destino se encuentre conectado a la red umaguma cuando el uServer recibe el mensaje "249", el Servidor modifica dicho mensaje para convertirlo en un mensaje de texto ordinario y lo envía al uAP en que se encuentra conectado dicho Terminal.

En la siguiente figura se observan los 2 casos mencionados.



5.8.2 Recepción de un mensaje Store And Forward

1. Cada vez que el uServer recibe una notificación de que un nuevo Terminal se conectó a la red, busca, a partir de la MAC del uCel, si hay mensajes almacenados en el Store And Forward para dicho usuario.
2. En caso de que existan registros para el nuevo Terminal en la tabla correspondiente al Store And Forward de la Base de Datos, el uServer obtiene la información almacenada en los campos de la misma para generar una trama por

registro, conteniendo: el mensaje a enviar, la dirección umaguma del Terminal de destino, la MAC y el friendly name del Terminal remitente y la fecha y hora en que se almacenó el mensaje.

3. El uServer envía las tramas al uAP en que se encuentra conectado el Terminal en cuestión.
4. Por último, el uAP enruta cada una de las tramas provenientes desde el uServer hacia el uCel de destino, que se encarga de procesar y desplegar el mensaje.

Capítulo 6: Sistema de Estadísticas de la Red

6.1 Motivación

Tal como fue explicado en capítulos anteriores, la red cuenta con un sistema de estadísticas el cual almacena su información en la base de datos de la red.

Un sistema de estadísticas surge por la necesidad de poder tener una noción sobre el comportamiento de los nodos de la red, respecto al tráfico que cursan y las posibles congestiones de estos. De esta manera se podrá identificar cuales son aquellos nodos mas utilizados y con que carga de tráfico.

La idea principal, es crear un sistema estadístico en analogía a los utilizados en las redes de telefonía celular, los cuales aportan información necesaria tanto para conocer el funcionamiento de la red como así también para basar procesos de optimización de esta.

Entre otras cosas, la información que aportan las estadísticas de cada nodo, posibilita que se planifiquen cambios en la ubicación de los nodos, brindando mayor capacidad en las zonas más necesitadas. La gran demanda de mensajes en un nodo, puede afectar el servicio de la red ya que introduciría mayores retrasos en los envíos así como también otros problemas de funcionamiento. Esto puede ser percibido mediante estadísticas y basado en ello se puede proceder a la colocación de un nuevo punto de acceso que balance el tráfico entre él y el nodo congestionado. Por el contrario, nodos de tráfico nulo o prácticamente nulo, pueden ser considerados innecesarios y ser quitados o relocalizados en zonas mas necesitadas.

Otra ventaja de poseer este tipo de herramienta, es la posibilidad de mediante ella poder identificar posibles problemas en el funcionamiento de la red. Por ejemplo, si un nodo ubicado en una zona de alto tráfico presenta una disminución importante en sus estadísticas, podría percibirse allí una falla la cual puede ser tanto a nivel de software como de hardware. Así mismo una caída de un nodo presentará estadísticas nulas durante su inactividad, lo que puede dar una noción del período que estuvo sin proporcionar servicio.

6.2 Implementación

6.2.1 Software de Gestión de Estadísticas

En contrapartida con lo visto anteriormente, en esta sección se describirá el funcionamiento de un sistema de estadísticas de tráfico de la red, visto desde el lado del usuario. Consiste en la creación de un software de gestión capaz que entregarle al usuario las cifras finales procesadas de los datos almacenados en la base, dándole una visión lo suficientemente clara de la situación del tráfico en la red.

Este software de gestión posee la gran ventaja de que no requiere obligatoriamente de un equipo en particular, sino que es flexible a poder ser lanzado desde cualquier equipo. Puede ser ejecutado tanto en un nodo en particular, en paralelo con el software que realiza las funciones de dicho nodo, así también como remotamente desde una computadora ajena a la red con la única condición de tener conexión con la base de datos. Esto es posible ya que la información que maneja se basa exclusivamente en lo almacenado en la base de datos y que a su vez esta permite conexiones remotas.

Otra gran ventaja que ofrece este tipo de acceso, es que se pueden ejecutar varias copias en simultáneo en varios equipos, dando la posibilidad de proporcionar una gestión no exclusiva a un usuario ni a una sola ubicación.

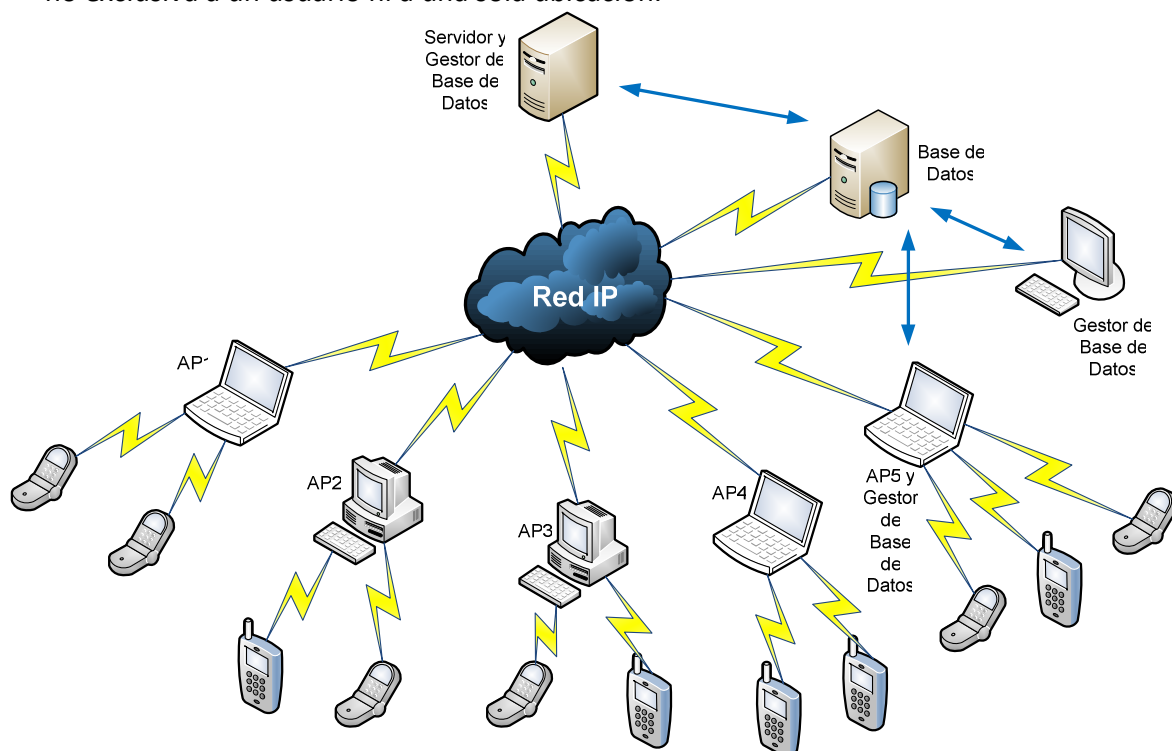


Figura 6.1 - Diagrama de la red con diferentes posibilidades de ubicación del software de gestión de estadísticas

6.2.2 Definición de criterios de medición de estadísticas

Como ya fue mencionado anteriormente, cada nodo de la red (Access Points y Servidor) se conecta a la base de datos para ingresar, mediante comandos SQL, una serie de contadores. Estas conexiones se dan cada 15 minutos e inmediatamente después de ser entrados los datos se procede a la desconexión. La información proporcionada por dichos valores ingresados servirá para el cálculo de estadísticas de tráfico en cada uno de los nodos. Estos datos serán tomados por el software de gestión según criterios establecidos por el usuario, para luego ser procesados y ser presentados ante estos los resultados finales.

Dado que esta red es íntegramente de mensajería, se tomaron en cuenta para medir estadísticamente cantidades de mensajes y sus tamaños en bytes.

Es de gran importancia hacer una distinción entre los tipos de mensajes que se envían y reciben entre los nodos de la red. Los diferentes tipos de mensajes (texto, imágenes y voz) poseen tamaños bastante desiguales en bytes (el ejemplo más claro es que los de texto suelen ser varias veces más pequeños que los de imágenes o voz) por lo que carecería de sentido contar todos los mensajes por igual sin distinguir tipo, ya que no aportaría ninguna noción de tráfico precisa. Por este motivo, fue que se decidió implementar contadores diferentes para cada tipo de mensaje tanto para registrar la cantidad de mensajes como sus respectivos tamaños en bytes.

Los tamaños de los mensajes en bytes son medidos a nivel de capa de aplicación. Esto quiere decir que la cantidad de bytes registrada por cada mensaje incluye lo ocupado por el cuerpo del mensaje mismo y los encabezados propios del protocolo desarrollado en este proyecto. Dicho de otra manera, no se incluyen en esta medida los encabezados propios del protocolo TCP/IP que son utilizados dentro del core de la red.

También es relevante notar, que no todos los mensajes que se dan entre los nodos de la red son registrados en el sistema estadístico. No se añaden a las estadísticas el tráfico generado por aquellos mensajes especiales entre nodos de la red, necesarios para el correcto funcionamiento de la red (por ejemplo, mensajes de control, mensajes donde se envían listas de usuarios conectados, consultas a la base de datos, etc).

Las estadísticas son exclusivamente del tráfico generado por mensajes de carga útil para el usuario (texto, imágenes o voz) cuyo destino final sea el propio usuario. Entre estos mensajes se incluyen únicamente los generados por usuarios, y no así los generados por algún nodo en particular (de forma tal de hacer un broadcast local con los terminales que se conectan a él). Estos últimos no serán contabilizados, ya que la idea del sistema de estadísticas es tener un control del tráfico "impredecible" (es decir, del cual no se puede

tener un control ya que depende exclusivamente de los usuarios), mientras que el tráfico generado por los nodos puede ser controlado y gestionado por la propia red.

Análogamente, los mensajes generados por el Servidor (cuyo objetivo puede ser hacer un broadcast ya sea global de la red o específico de ciertos APs que componen una zona) tampoco serán incluidos en la información estadística.

Al tener los dos tipos de nodos (Access Points y Servidor) funciones diferentes, a nivel estadístico deben clasificarse algunos mensajes en diversas categorías según su función. Es por ello que para cada uno de estos nodos se analizan algunos contadores diferentes, los cuales también deben ser procesados de una manera distinta al momento de entregar los resultados al usuario.

6.2.3 Funcionamiento

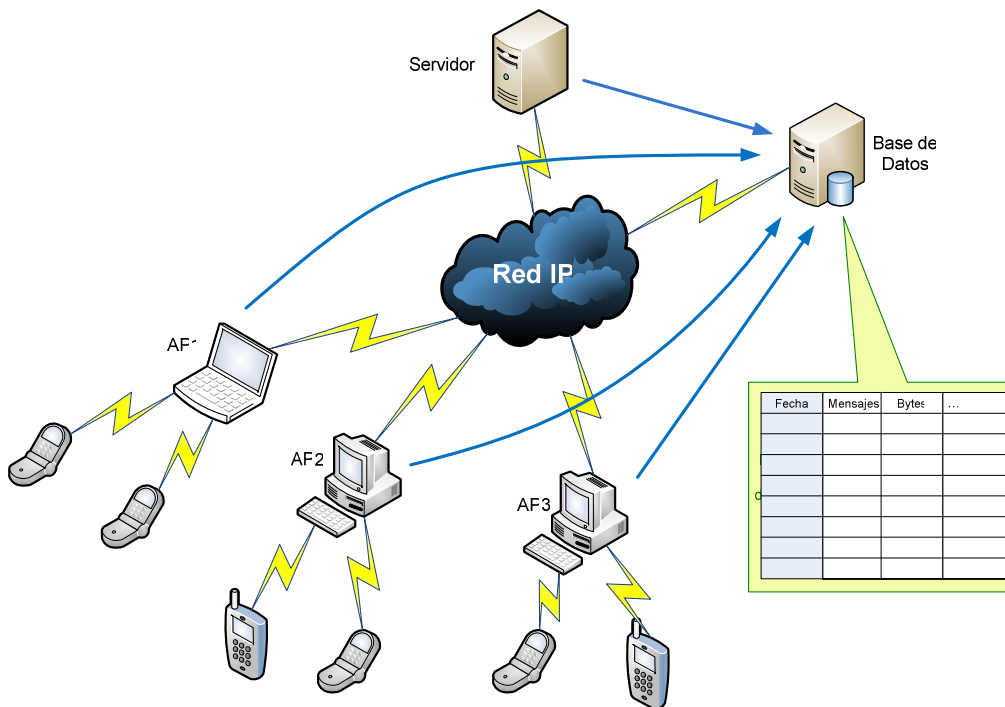
A continuación, se describirá el funcionamiento del sistema, sintetizando las posibles funcionalidades que ofrece el mismo.

En primer lugar, al iniciar el software, el usuario debe indicar los datos necesarios para realizar la conexión a la base de datos. Esto consiste en ingresar la dirección IP, puerto y contraseña. Consecuentemente con esto, deberá indicar de qué tipo de nodo (Access Point o Servidor) son las estadísticas que quiere analizar.

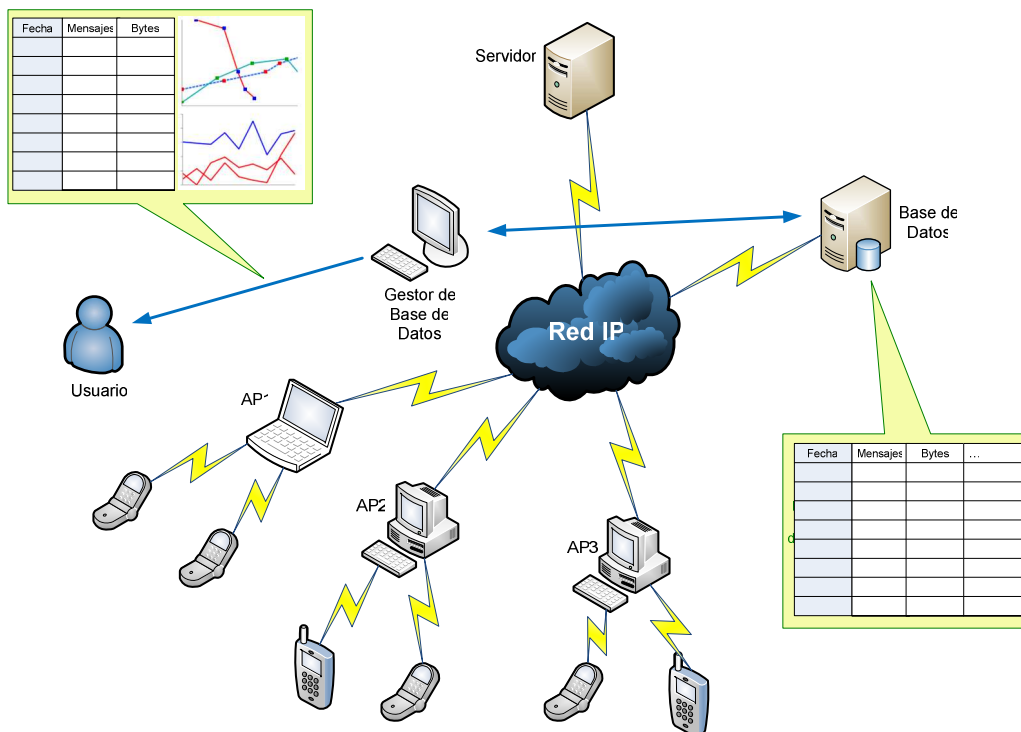
Una vez establecida la conexión e indicado el tipo de nodo, el programa ya conoce a qué tabla de la base de datos debe dirigirse para conseguir la información. Restaría entonces ahora, que el usuario indique el período en el cual desea obtener las mediciones, ingresando fecha y hora de comienzo y fin.

A partir de estos datos, se arma un comando SQL el cual seleccionara las filas de la tabla que corresponden al período de medición solicitado, para ser inmediatamente transferidas al software para su procesamiento.

En el caso de ser un Access Point el tipo de nodo seleccionado, se ofrece la posibilidad de visualizar las estadísticas de todos los APs dentro del período o de un AP en particular. En este último caso deberá indicar la dirección de dicho AP en la red, y a partir de esta se añadirá una restricción más al comando SQL que realiza la selección de las filas correspondientes.



Situación 1 – Servidor y APs ingresan sus contadores a la base de datos cada 15 minutos y llenan sus respectivas tablas



Situación 2 – El Gestor de Estadísticas, toma la información de la base de datos y luego de procesarla le presenta los datos finales al usuario

6.2.4 Procesamiento de datos y resultados finales

6.2.4.1 Tablas de resultados finales

Los cálculos correspondientes para el procesamiento final son distintos según el tipo de nodo seleccionado.

En una primera instancia se le presentará al usuario una tabla que expone las estadísticas solicitadas, la cual es producto de los datos obtenidos de la base luego de ser procesados por el software en cuestión.

Aquí hay que hacer una distinción ya que la información presentada para las estadísticas de los Access Points difiere de la del Servidor.

Analicemos primero el caso de la tabla presentada para las estadísticas del Servidor. Como se mencionó en un capítulo anterior, los datos aportados por el Servidor en períodos de 15 minutos se almacenan en una tabla especial de la base de datos la cual contiene los siguientes campos:

Fecha y Hora	Mensajes E/S de Texto	Mensajes E/S de Imágenes	Mensajes E/S de Voz	Mensajes S&F Entrantes	Mensajes S&F Salientes	Mensajes S&F Acumulados
Bytes E/S de mensajes de Texto	Bytes E/S de mensajes de Imágenes	Bytes E/S de mensajes de Voz	Bytes de mensajes S&F Entrantes	Bytes de mensajes S&F Salientes	Bytes de mensajes S&F Acumulados	

La explicación del significado de cada campo se detalla a continuación:

- **Fecha/Hora:** Fecha y hora en que fue ingresada dicha línea a la tabla.
- **Mensajes E/S de texto:** Cantidad de mensajes de texto de entrada/salida que fueron cursados por el nodo en el período de 15 minutos entre que fue ingresada la línea leída y la línea anterior.
- **Mensajes E/S de imágenes:** ídem anterior pero para mensajes de imágenes.
- **Mensajes E/S de voz:** ídem anterior pero para mensajes de voz.

- **Mensajes S&F entrantes:** Cantidad de mensajes que fueron ingresados (dentro del lapso de 15 minutos entre que fue ingresada la línea actual de la tabla y la anterior) a la base de datos para ser almacenados por el sistema de *store and forward*, con el objetivo de ser entregados cuando el Terminal de destino se conecte a la red.
- **Mensajes S&F salientes:** Cantidad de mensajes que se encontraban almacenados en la base de datos por el sistema de *store and forward* y fueron entregados a los terminales de destino dentro del período descrito en el punto anterior.
- **Mensajes S&F acumulados:** Cantidad de mensajes que quedaron acumulados en la tabla de la base de datos correspondiente al sistema de *store and forward*, en el momento que en el que la línea de la tabla leída fue ingresada.
- **Bytes E/S de texto:** tamaño en bytes correspondientes a los mensajes descriptos en el campo Mensajes E/S de texto.
- **Bytes E/S de imágenes:** tamaño en bytes correspondientes a los mensajes descriptos en el campo Mensajes E/S de imágenes.
- **Bytes E/S de voz:** tamaño en bytes correspondientes a los mensajes descriptos en el campo Mensajes E/S de voz.
- **Bytes S&F entrantes:** tamaño en bytes correspondientes a los mensajes descriptos en el campo Mensajes S&F entrantes.
- **Bytes S&F salientes:** tamaño en bytes correspondientes a los mensajes descriptos en el campo Mensajes S&F salientes.
- **Bytes S&F acumulados:** tamaño en bytes correspondientes a los mensajes descriptos en el campo Mensajes S&F acumulados.

El procesamiento de los datos consiste en una primera instancia en ampliar la información presentada al usuario, realizando simples cálculos requeridos para obtener nuevos campos de información, lo cuales representarán valores totales de cantidades de mensajes y cantidad de bytes cursados.

A continuación se muestran los campos de información de la tabla de resultados que se le presenta al usuario y los cálculos correspondientes para la obtención de estos:

Fecha y Hora	Mensajes E/S de Texto	Mensajes E/S de Imágenes	Mensajes E/S de Voz	Mensajes S&F Entrantes	Mensajes S&F Salientes	Mensajes S&F Acumulados	Bytes E/S de mensajes de Texto
Bytes E/S de mensajes de Imágenes	Bytes E/S de mensajes de Voz	Bytes de mensajes S&F Entrantes	Bytes de mensajes S&F Salientes	Bytes de mensajes S&F Acumulados	Total de Bytes Entrantes	Total de Bytes Salientes	

Todos los datos que se presentaran en cada línea de dicha tabla, corresponderán a eventos ocurridos dentro del período de 15 de minutos transcurrido entre la línea que se lee y la anterior. Sólo cabe realizar una aclaración y es que en los campos de Mensajes de Store and Forward acumulados y Bytes de Store and Forward acumulados, estos se refieren a aquellos mensajes y bytes que quedaron almacenados en la base de datos al momento de ser ingresada a la base de datos dicha línea en la tabla.

Los nuevos campos de información que se visualizan se obtienen a partir de simples cálculos acumulativos de datos almacenados en la tabla de la base de datos.

En el caso de las estadísticas del servidor los nuevos campos se componen de los totales de bytes que entran y salen del servidor. Para ello fue necesario analizar todas las posibilidades en donde se da que entren o salgan del servidor mensajes que contengan carga útil para el usuario (es decir, texto, imágenes o voz) ya que como se mencionó anteriormente son los únicos considerados por los criterios fijados para la realización de este sistema de estadísticas.

Analizando los mensajes entrantes se observo que no se componen únicamente de aquellos que llegan por medio de los Access Points, sino que también, al contar la red con un sistema de store and forward, en ciertos momentos llegaran al Servidor mensajes que habían sido almacenados en la base de datos para su posterior entrega. Por lo tanto el total de bytes entrantes al servidor se compone de la siguiente manera:

total de bytes entrantes = bytes E/S de mensajes de texto + bytes E/S de mensajes de imágenes + bytes E/S de mensajes de voz + bytes de mensajes que salen de store and forward

Análogamente, los mensajes salientes del servidor se componen de los que mensajes que salen hacia los Access Points (que pasaron por el Servidor provenientes de otros

Access Points), así como también aquellos que son enviados a la base de datos para ser almacenados cumpliendo las funciones del store and forward.

Es importante notar aquí una particularidad, y es que aquellos mensajes que salen del store and forward serán entrantes al servidor pero inmediatamente serán enviados a los Access Points correspondientes a los terminales de destino. Estos mensajes salientes del servidor, no se encuentran incluidos en entre los mensajes E/S almacenados en la tabla de la base de datos ya que justamente el sistema de store and forward no posee una salida del mensaje inmediata a la entrada de este al nodo. Además la entrada y salida de un mensaje de este tipo no tiene porque darse dentro del mismo lapso de 15 minutos que se da entre los almacenamientos de contadores en la base de datos, y por tanto no puede ser considerado mensaje de E/S aún si su salida no fuese estrictamente inmediata a su entrada. Por este motivo debe incluirse además al calculo de los mensajes salientes del Servidor, los mensajes que salieron de store and forward, dentro de ese lapso de 15 minutos.

De esta manera podemos decir que:

$$\text{total de bytes salientes} = \text{bytes E/S de mensajes de texto} + \text{bytes E/S de mensajes de imágenes} + \text{bytes E/S de mensajes de voz} + \text{bytes de mensajes que entran a store and forward} + \text{bytes de mensajes que salen de store and forward}$$

O resumiendo de otra manera:

$$\text{total de bytes salientes} = \text{total de bytes entrantes} + \text{bytes de mensajes que salen de store and forward}$$

De manera similar, se analiza ahora lo ocurrido con las estadísticas de los Access Points.

Como fue mostrado anteriormente en el presente documento, en la base de datos los campos de información de la tabla de almacenamiento de contadores de estadísticas de los Access Points son los siguientes:

Fecha y Hora	Dirección de AP	Mensajes E/S de Texto	Mensajes E/S de Imágenes	Mensajes E/S de Voz
Bytes E/S de mensajes de Texto	Bytes E/S de mensajes de Imágenes	Bytes E/S de mensajes de Voz	Bytes Locales	Bytes hacia el Servidor

Varios de los campos que presenta esta tabla, coinciden con los presentados en la tabla de estadísticas del servidor. Los otros son detallados a continuación:

- **Dirección uAP:** Dirección (en la red) correspondiente al AP cuyos contadores se muestran en el resto de la línea.
- **Bytes locales:** Total de bytes correspondientes a los mensajes de E/S que pasaron por el Access Point, cuyo origen y destino son terminales que se encuentran conectados a él (es decir, no pasan en ningún momento por el servidor).
- **Bytes hacia el Servidor:** Total de bytes correspondientes a los mensajes de E/S que llegan al Access Points provenientes de terminales conectados a el y son reenviados por este al Servidor.

Los campos de información de la tabla que se le presentará al usuario son:

Fecha y Hora	Dirección de AP	Mensajes E/S de Texto	Mensajes E/S de Imágenes	Mensajes E/S de Voz	Bytes E/S de mensajes de Texto
Bytes E/S de mensajes de Imágenes	Bytes E/S de mensajes de Voz	Bytes Locales	Bytes hacia el Servidor	Total de bytes E/S en el Servidor	

Para este caso, los datos representados en los nuevos campos no presentan complejidad, ya que en este nodo si ocurre que todos los mensajes que entran son reenviados inmediatamente.

Sólo se presenta un nuevo campo, y simplemente representa el total del bytes cursados por el Access Point en el período detallado anteriormente. Como se trata de un nodo exclusivamente de E/S, no tendría ningún sentido distinguir un total de entrantes de uno de salientes ya que serían lo mismo.

Si bien este total es representado por la suma de cada tipo de mensaje cursado, aprovechando de las posibilidades que nos ofrecen los campos descriptos, se lo puede representar de la siguiente manera:

$$\text{Total de bytes E/S en el Servidor} = \text{Bytes Locales} + \text{Bytes hacia el Servidor}$$

6.2.4.2 Gráficos de resultados finales

Descrito todo lo que respecta a las tablas presentadas al usuario, restaría ahora ver ahora la segunda etapa del procesamiento de la información obtenida de la base de datos.

En esta etapa, se le ofrece al usuario la posibilidad de visualizar gráficos con diferentes representaciones de lo ocurrido en lo que respecta al tráfico cursado en el período de tiempo seleccionado por el usuario.

El poder observar gráficos es de gran importancia, ya que posibilitan tener una visión mas simplificada de las variaciones en todos los intervalos de tiempo, a la vez que permite mostrar claramente tendencias y balances, algo que sería de gran dificultad si se lo intenta hacer desde la simple observación de una tabla.

Además, permite realizar comparaciones entre los diferentes tipos de mensajes, observando cuales son los mas demandados pudiendo también identificar horas pico donde hay mayor tráfico de mensajes de cada tipo.

Para cada tipo de nodo se presenta un grupo de gráficos estadísticos distinto. Se describen a continuación cada uno de estos gráficos según el tipo de nodo analizado.

Para el caso de las estadísticas del Servidor, los gráficos presentados son los siguientes:

- **Mensajes E/S en el Servidor:** donde se grafican la cantidad de mensajes en función del tiempo. En mismo recuadro son representados tres trazos, cada uno correspondiente a cada uno de los tres tipos de mensajes (texto, imágenes y voz).
- **Bytes E/S en el Servidor:** en este gráfico se representan en tres trazos los bytes correspondientes a los mensajes graficados en el punto anterior, en función del tiempo.
- **Mensajes en Store and Forward:** grafica en tres trazos diferentes, la cantidad de mensajes de *store and forward* entrantes, salientes y acumulados, en cada instante de tiempo dentro del período seleccionado.
- **Bytes en Store and Forward:** representa en tres trazos los bytes correspondientes a los mensajes graficados en el punto anterior, en función del tiempo.

- **Tráfico Total en el Servidor:** muestra en dos trazos diferentes, el total del bytes tráfico entrante y saliente, calculado según como se indicó en los campos Total de bytes entrantes y Total de bytes salientes, explicados anteriormente.

De la misma manera para el caso de los Access Points, se describen a continuación el grupo de gráficos presentados:

- **Mensajes E/S del Access Point:** representa en tres trazos la cantidad de mensajes de texto, imágenes y voz que pasan por el Access Point en función del tiempo dentro del período seleccionado por el usuario.
- **Bytes E/S del Access Point:** incluye tres trazos representando los bytes correspondientes a los mensajes graficados en el punto anterior.
- **Tráfico Total en el Access Point:** representa mediante tres trazos los bytes correspondientes a los mensajes locales, al servidor y totales, en función del tiempo. Los dos primeros corresponden a los campos de la tabla explicada anteriormente, mientras que el último es la suma de ambos.

Capítulo 7: Gestión de la red y Políticas de Conexión

7.1 Introducción

A medida que la red fue creciendo en tamaño y prestaciones, surgió la necesidad de implementar ciertas políticas para gestionar y controlar el estado de la misma.

Para esto, Umaguma implementa las siguientes funciones:

- Posibilidad de configurar una contraseña de 6 dígitos en cada uAP para controlar las conexiones de Terminales al AP.
- Posibilidad de restringir la visibilidad entre determinados uAP.
- Separación de la red en zonas.
- Posibilidad de desconectar cualquier elemento de la red desde un nodo de mayor jerarquía.
- Posibilidad de enviar información desde cualquier nodo de la red.

A continuación se dará una breve descripción de las funciones mencionadas, para luego terminar el capítulo con ejemplos de aplicación de las mismas.

7.2 Descripción

7.2.1 Acceso con Contraseña

Esta función permite restringir la conexión de los Terminales a ciertos Puntos de Acceso, por lo que puede considerarse una política de acceso a la red. Esta política solo estará disponible si en la configuración inicial del uServer se activó el uso de contraseñas en los uAP. Cada vez que se conecta un nuevo AP, es posible asignarle desde el uServer una contraseña de 6 dígitos.

La contraseña generada por el uServer es informada al uAP utilizando un mensaje especial “254”, junto con la dirección umaguma del nuevo AP. La contraseña es almacenada en el nodo de acceso, y solicitada a los nuevos usuarios que pretenden ingresar a la red.

Cuando un usuario pretende ingresar a la red Umaguma, el nodo de acceso le solicitara contraseña. El usuario tendrá como máximo 3 intentos para ingresar la contraseña, si el usuario supera esta cantidad no podrá ingresar a la red.

7.2.2 Visibilidad

Con esta función se logran aplicar políticas para restringir la visibilidad entre los Puntos de Acceso de la red. Este tipo de política solo aplica entre uAP y se configura desde el uServer, mediante la interfaz gráfica del mismo.

Esta política fue diseñada de manera de que los usuarios conectados a un AP con visibilidad restringida, no sean capaces de comunicarse con usuarios conectados a otros Puntos de Acceso. Además, la visibilidad se pensó para que fuera “bidireccional”, de manera que si se configura el AP1 para que no pueda ver al AP2, el AP2 tampoco podrá ver al AP1.

La visibilidad es implementada modificando la tabla de usuarios conectados que se despliega en la interfaz gráfica del uCel. Para esto, los elementos de la red realizan las siguientes acciones:

- El uServer vuelve a crear la tabla que contiene los friendly names y las direcciones umaguma de todos los usuarios conectados (tabla que, como se mencionó en capítulos anteriores, se pasa a cada Terminal cuando ingresa a la red), excluyendo a los usuarios conectados a los Puntos de Acceso que fueron configurados para dejar fuera de la visibilidad. Estas tablas también se vuelven a crear para los usuarios de los uAP involucrados en la modificación de la visibilidad para cumplir con la característica de “bidireccionalidad” explicada.
- El Servidor envía a los AP involucrados las nuevas tablas de usuarios, que contienen solo los Terminales que podrán ser visibles desde cada Punto de Acceso.
- Cada uAP reenvía las tablas recibidas a todos los Terminales conectados en dicho nodo.

- Los Terminales actualizan sus listas de usuarios conectados, finalizando así, el proceso de modificación de visibilidad entre APs.

7.2.3 Separación de la red en Zonas

Este punto refiere a una función exclusiva del uServer, que permite gestionar la red a partir del agrupamiento de Puntos de Acceso en Zonas. El hecho de que para el Servidor exista un objeto que represente a un conjunto de Puntos de Acceso, permite por ejemplo, distribuir rápidamente contenido a todos los Terminales que se encuentren dentro de una Zona, como un aviso de emergencia, información sobre algún evento en dicha Zona, etc.

Cuando un uAP se conecta, desde el Servidor se deben configurar parámetros antes de dar por establecida la conexión. Uno de estos parámetros es la Zona a la que pertenecerá dicho AP. En éste punto, quien tenga el control del uServer podrá elegir incluir al nuevo AP en alguna de las Zonas ya existente, o, crear una nueva Zona para el Punto de Acceso en cuestión.

7.2.4 Desconexiones Forzadas

Cualquier nodo de la red puede desconectar a cualquier elemento que se encuentre conectado a él sin previo aviso ni autorización. Así, un uAP puede desconectar a cualquier Terminal que se encuentre conectado a dicho nodo, mientras que el uServer cuenta con las siguientes opciones:

- Desconexión de un Terminal de la red.
- Desconexión de un uAP de la red.
- Desconexión de todos los AP pertenecientes a una determinada Zona.
- Desconexión de todos los AP de la red.

Para realizar la desconexión, se cuenta con mensajes especiales que indican al Terminal o al uAP la razón por la cual fueron desconectados. Esto ya fue mencionado en la descripción del protocolo en el Capítulo 3.

7.2.5 Envío de Contenido desde nodos

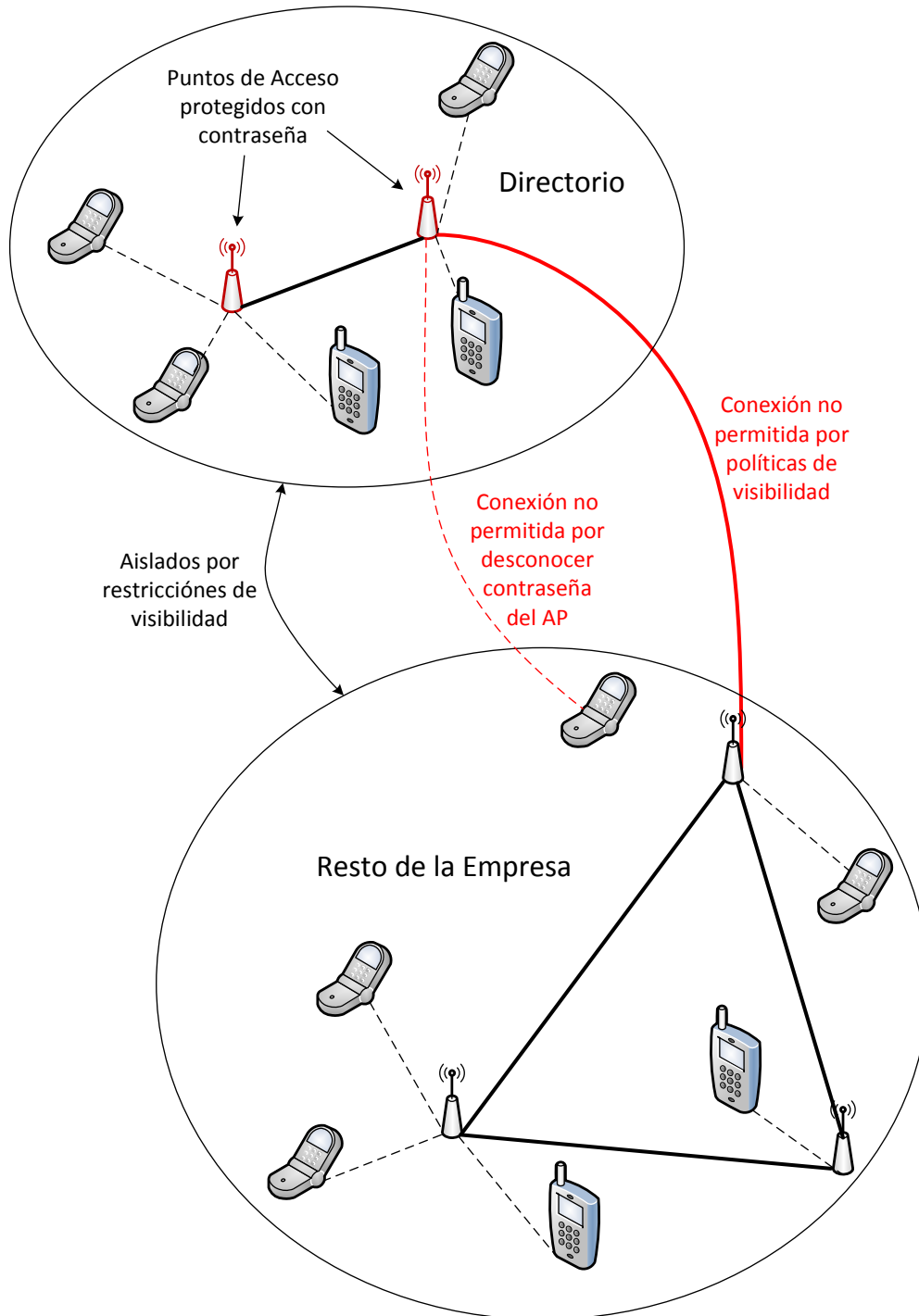
Todos los nodos de la red pueden comunicarse directamente con los Terminales a partir del envío de información utilizando mensajes especiales (ver Capítulo 3).

Los uAP son capaces de enviar únicamente mensajes de texto hacia cualquiera de los Terminales que se encuentren conectados a él, mientras que el uServer es capaz de enviar mensajes de texto e imágenes a cualquier Terminal de la red, a partir de las siguientes opciones de envío:

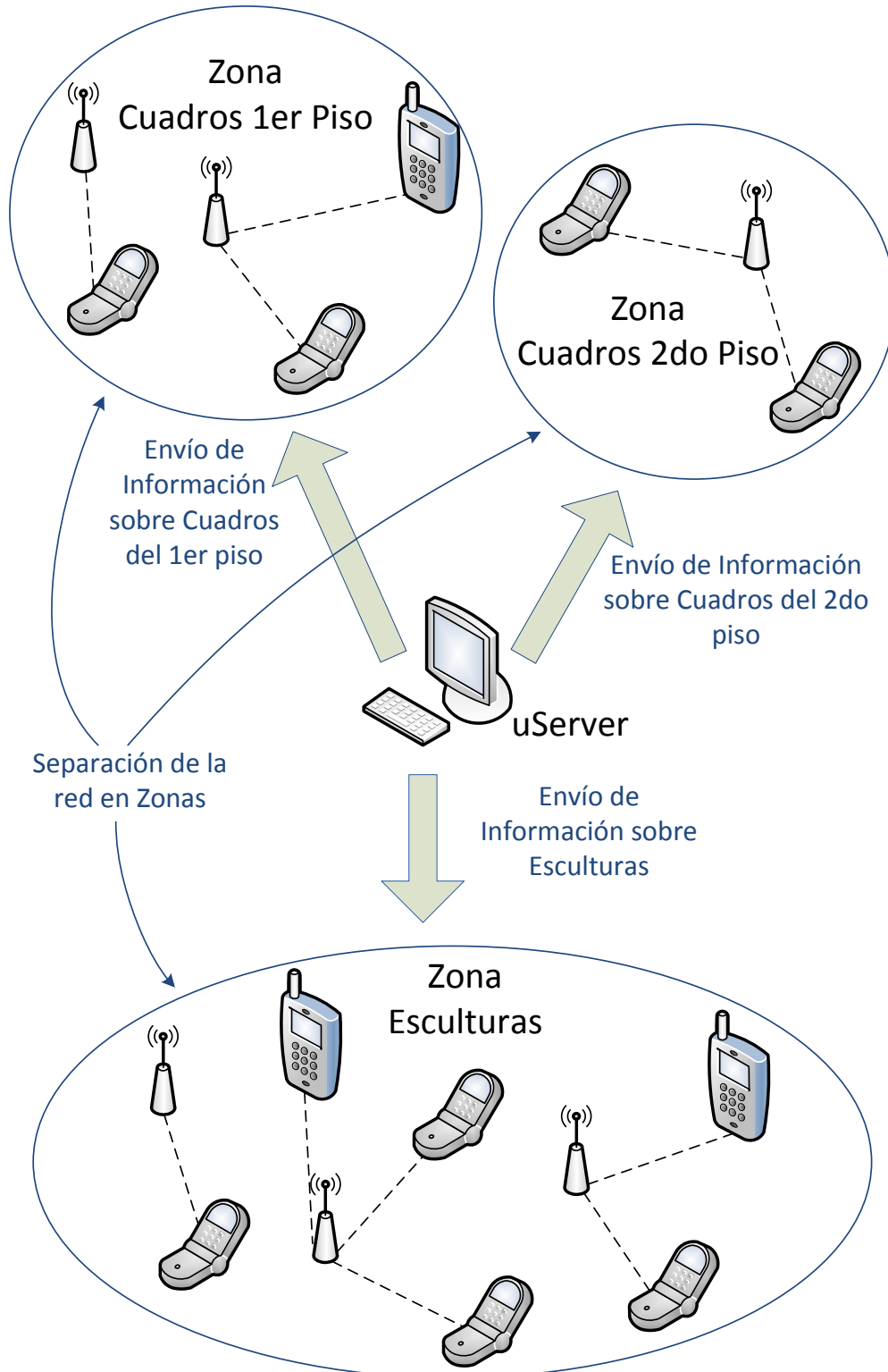
- A un solo Terminal en particular.
- A todos los Terminales conectados a un uAP.
- A todos los Terminales dentro de una Zona.
- A todos los Terminales de la red.

7.3 Posibles usos de las políticas

- **Ejemplo 1:** Utilización de políticas de visibilidad y contraseñas en algunos Puntos de Acceso (Aplicación a Empresas).



- **Ejemplo 2:** División de la red en Zonas y distribución selectiva de contenido (Aplicación a Museos).



Capítulo 8: Interfaz Gráfica

8.1. Introducción y objetivos

En el presente capítulo se detallará la aplicación gráfica realizada en cada uno de los nodos de la red, la cual posibilitará la interacción con el usuario, para cada uno de los sistemas desarrollados.

La principal premisa que se impuso para la realización de dichas interfaces gráficas, fue que fueran lo suficientemente “amigables” para el usuario (como suele decirse en la jerga de la programación: user friendly), en el sentido de que sean de fácil comprensión y operación para quien usa el sistema por vez primera.

Además, se analizaron diferentes opciones para la planificación de una interfaz lo más optimizada posible, desde el punto de vista de su rendimiento en función del hardware utilizado. Básicamente, se buscó tener una interfaz suficientemente atractiva y comprensible, consumiendo la menor cantidad posible de recursos de hardware.

8.2. Herramientas de desarrollo

Para entrar en el campo de las herramientas utilizadas para el desarrollo y diseño de las interfaces, se debe hacer una separación entre las que fueron utilizadas para la creación de la interfaz de los Terminales y las manejadas para el resto de los nodos (Access Points, Servidor y Gestor de Estadísticas).

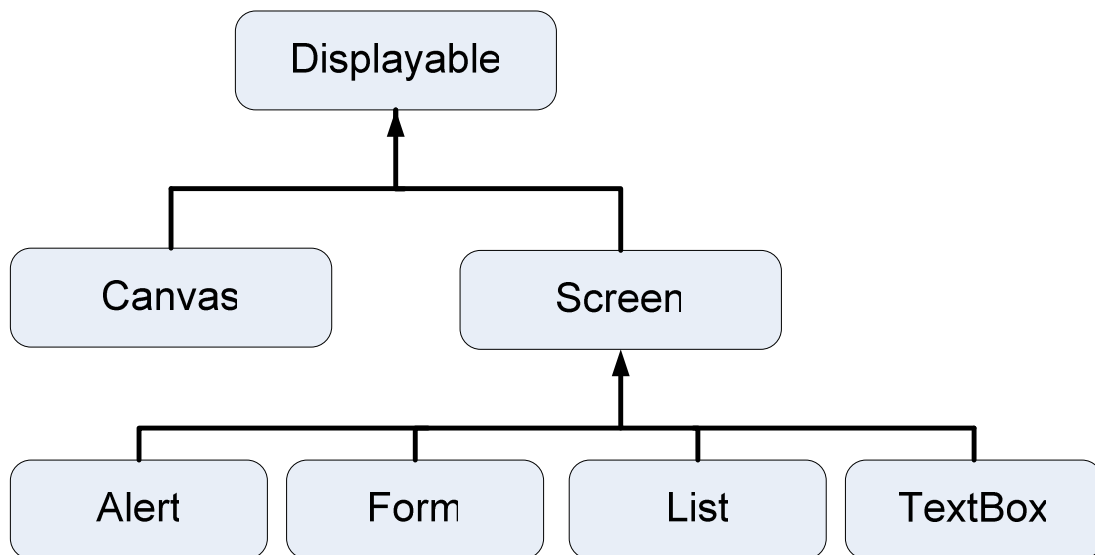
8.2.1 Interfaz gráfica de Terminales

La herramienta utilizada para la creación de la interfaz gráfica de los terminales se basó en J2ME (Java 2 Micro Edition). Tal como fue explicado en capítulos anteriores, J2ME es una la plataforma enfocada a la aplicación de la tecnología Java en dispositivos electrónicos con capacidades computacionales y gráficas muy reducidas, tales como teléfonos móviles, PDAs u otros dispositivos del estilo.

Dentro de esta plataforma, entre otros se encuentra el paquete javax.microedition.lcdui (Interfaz de usuario con pantalla LCD), el cual contiene una biblioteca de interfaz de usuario tanto de alto como de bajo nivel. De esta división es que

se distinguen las llamadas interfaces gráficas de alto nivel e interfaces de bajo nivel. Este paquete contiene un conjunto de clases y métodos necesario para el desarrollo de todo lo referente a interfaces gráficas, que ofrece esta plataforma para el tipo de dispositivo utilizado en este proyecto.

La estructura del mencionado paquete se compone de un grupo de clases básico en el cual se presentan los diferentes métodos utilizados para confeccionar las distintas pantallas y funcionalidades que compondrán la interfaz. Dicha estructura se detalla a continuación en el siguiente diagrama junto con un breve comentario sobre cada una de ellas.



- **Displayable:** es la clase base para todos los interfaces de usuario de una aplicación. De esta superclase heredarán todas las otras clases.

Las dos clases que heredan directamente de Displayable, corresponden con el API de alto nivel y bajo nivel de la interfaz de usuario. En este sentido, Screen se relaciona a la de alto nivel, mientras que Canvas a la de bajo nivel.

- **Canvas:** es una pantalla en la cual la aplicación puede dibujar directamente. Dicha pantalla no es tan estructurada como las tratadas en las interfaces de alto nivel.

- **Screen:** es la clase base para la creación de interfaces de alto nivel. De ella heredan todas las subclases que utilizan para la creación de interfaces gráficas estructuradas.
- **Alert:** esta clase crea pantallas de tipo emergente, utilizadas frecuentemente para dar mensajes de alerta.
- **Form:** representa una pantalla en la cual se pueden agregar distintos ítems ordenados de diversas maneras según se desee.
- **List:** esta clase implementa la creación de listas. Dentro de los tipos de listas se distinguen tres tipos básicos: exclusivo, múltiple e implícito.
- **TextBox:** es la clase que implementa pantallas donde se puede colocar texto y a su vez editar a éste.

8.2.3 Interfaz gráfica de nodos del core

Para crear las interfaces gráficas del resto de los nodos (Access Points, Servidor y Gestor de Estadísticas), en una primera instancia se realizó una investigación sobre las posibilidades que se ofrecen en Java para la realización de este tipo de interfaces.

Analizando diferentes posibilidades se encontró muy recomendable la opción de utilizar el paquete SWING, incluido en la JFC (Java Foundation Classes) en la plataforma Java.

SWING es una biblioteca gráfica para Java, la cual incluye widgets para interfaz gráfica de usuario tales como cajas de texto, botones, desplegables y tablas. Dentro de Java, dichos componentes se pueden encontrar en el paquete javax.swing.

Antes de la existencia de Swing, las interfaces gráficas con el usuario se realizaban a través de AWT (Abstract Window Toolkit), de quien Swing hereda todo el manejo de eventos. Usualmente, para toda componente AWT existe una componente Swing que la reemplaza, por ejemplo, la clase Button de AWT es reemplazada por la clase JButton de Swing (el nombre de todas las componentes Swing comienza con "J").

A partir de la versión 1.1 de JDK (Java Development Kit) se empezó a incluir SWING además de AWT.

Las componentes de SWING utilizan la infraestructura de AWT, incluyendo el modelo de eventos AWT. Es por esto, que la mayoría de los programas SWING necesitan importar dos paquetes AWT: `java.awt.*` y `java.awt.event.*`

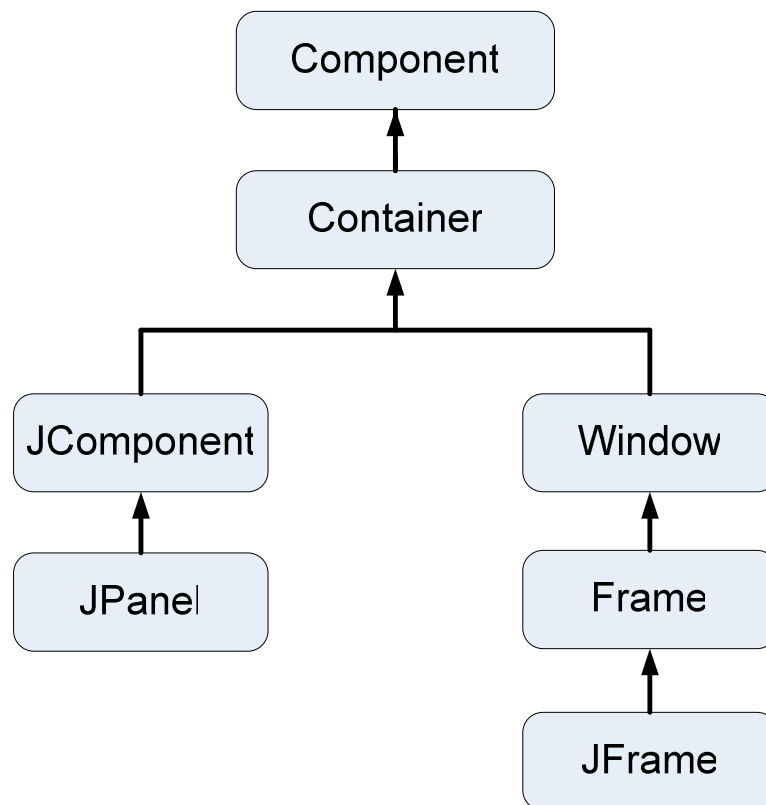
Algunas de las ventajas de SWING frente a AWT son:

- Los botones y las etiquetas SWING pueden mostrar imágenes en lugar de o además del texto.
- Se pueden añadir o modificar fácilmente los bordes dibujados alrededor de casi cualquier componente SWING. Por ejemplo, es fácil poner una caja alrededor de un contenedor o una etiqueta.
- Se puede modificar fácilmente el comportamiento o la apariencia de un componente SWING llamando a métodos o creando una subclase.
- Los componentes SWING no tienen por que ser rectangulares. Por ejemplo, los botones pueden ser redondos.
- Las tecnologías asistivas como los lectores de pantallas pueden fácilmente obtener información desde los componentes SWING. Por ejemplo, una herramienta puede fácilmente obtener el texto mostrado en un botón o en una etiqueta.

Otra característica de SWING es que se puede especificar el Aspecto y Comportamiento que utilice el GUI de nuestro programa. Por el contrario, los componentes AWT siempre tienen el aspecto y comportamiento de la plataforma nativa.

Lo mencionado en el párrafo anterior fue de gran importancia para es proyecto, ya que se pretendía tener la posibilidad de que el software creado pueda ser ejecutado en diferentes sistemas operativos y no estar ligado a sólo uno.

Como se mencionó anteriormente, SWING hereda de AWT, por lo que su estructura de clases se verá ligada a la de este último. A continuación se detallará dicha estructura, señalando además los principales componentes:



Las clases cuyo nombre comienza con “J” pertenecen a SWING, mientras que las restantes son parte de AWT.

- **Component:** es una clase abstracta que representa cualquier componente con representación gráfica.
- **Container:** es una componente que puede contener otros componentes gráficos.
- **JFrame:** permite representar ventanas. Son contenedores los cuales incluyen un “panel de contenido” (content pane) al cual se le pueden agregar diversos componentes gráficos como por ejemplo: etiquetas, botones, cuadros de texto, etc.

Algunos de los principales componentes que se pueden incluir (y que fueron utilizadas para la realización de las interfaces gráficas en este proyecto) en un JFrame son:

- **JLabel:** etiqueta para mostrar texto.

- **JTextBox:** cuadro para introducir texto.
- **JButton:** botón.
- **JCheckBox:** recuadro de comprobación.
- **JList:** lista de opciones.
- **JComboBox:** lista desplegable de opciones.
- **JScrollBar:** barra de scroll.
- **JTree:** árbol.
- **JTable:** tabla.
- **JMenu o JMenuBar:** diferentes tipos de menú.

Por otra parte, para la realización de gráficas, las cuales son utilizadas tanto en las interfaces gráficas del software Gestor de Estadísticas, así como también en la de los Access Points y el Servidor, se requirió de una herramienta adicional.

La herramienta utilizada es una librería para gráficas llamada JFreeChart, la cual esta escrita en su totalidad en Java. Esta librería posibilita la presentación de gráficas de calidad en plaicaciones Java, ya sean web o de escritorio.

Algunas de las caracterisiticas de JFreeChart son:

- Un API consistente con soporte para un amplio rango de tipos de gráficos.
- Soporte para varios tipos de salida, incluyendo componentes SWING como así también archivos de imagen PNG y JPEG, y formatos gráficos de vectores (incluyendo PDF, EPS y SVG).
- Es un software libre y open source, que permite su uso en aplicaciones propietarias.

8.3. Interfaces gráficas desarrolladas

8.3.1 Interfaz gráfica de Terminales

Al dar comienzo a la aplicación nos encontramos -en primer lugar- con una pantalla inicial que se muestra durante 2 segundos a modo de carátula, en ella se mencionan algunas características del proyecto. Luego se nos solicita que seleccionemos a cual punto de acceso nos deseamos conectar. En la figura 8.1 se muestra una captura de esta pantalla para el prototipo desarrollado, la idea es que para una aplicación práctica del sistema las opciones reflejen los nombres de las zonas en las que se encuentra cada punto de acceso, por ejemplo Oficinas, Deposito, Recepción, etc.

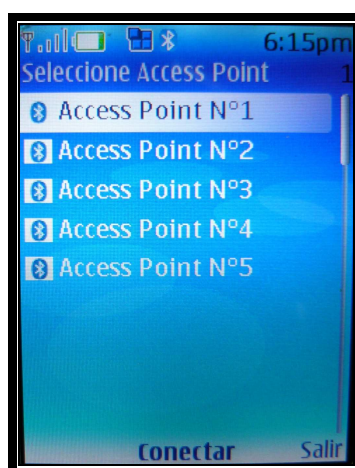


Figura 8.1 – Pantalla de selección de punto de acceso

Dependiendo de lo establecido por el servidor (o mejor dicho su administrador) la aplicación puede solicitarnos (o no) el ingreso de contraseña de autenticación, para esto deponemos de 3 intentos, tras lo cual la conexión es cerrada por el sistema, debiéndose reiniciar el programa para intentarlo nuevamente. La contraseña ingresada permanece oculta a la vista mediante asteriscos como se muestra en la siguiente imagen:

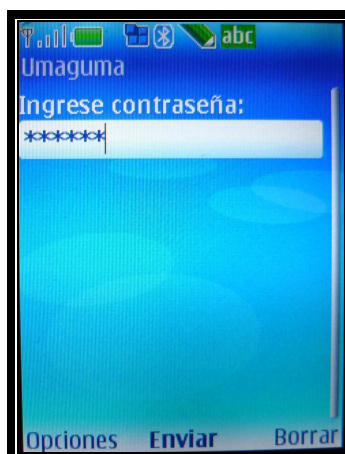


Figura 8.2 – Pantalla de ingreso de contraseña

Luego de ingresar correctamente la contraseña accedemos a la pantalla principal del programa en la que observamos tres campos principales:

- **Escriba mensaje:** Este es el campo en el que el usuario ingresa el mensaje a enviar, tanto en el caso de mensajes ordinarios como los mensajes Store and Forward.
- **Estado:** Aquí se muestra información sobre el estado de la conexión así como también sobre las acciones que el usuario va realizando, reportando por ejemplo cuando hay fallos.
- **Mensaje recibido:** En este espacio se van desplegando los mensajes recibidos a lo largo de la comunicación en orden creciente de llegada. El último mensaje recibido se muestra en la parte superior y los mensajes viejos se acumulan hacia abajo, debiéndose realizar scroll para visualizarlos si se acumula la suficiente cantidad. También se muestra el nombre del remitente junto al mensaje.

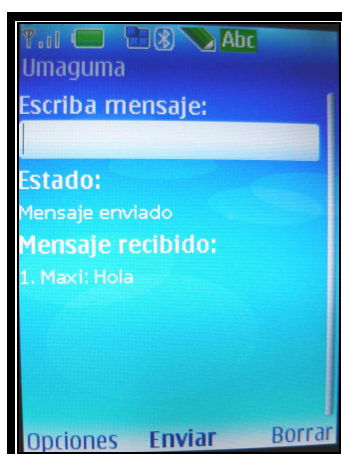


Figura 8.3 – Pantalla principal

Para enviar un mensaje de texto a uno o varios destinos se lo debe ingresar mediante el teclado alfanumérico, visualizándolo en el campo superior y luego se debe oprimir el botón “Enviar”. Previamente se deben seleccionar los destinos a los cuales se enviará el mensaje. Para esto debemos utilizar el botón “Destinos” dentro del menú de opciones desde donde se accede a las distintas funcionalidades de la aplicación. Este menú se muestra a continuación en la imagen 8.4.

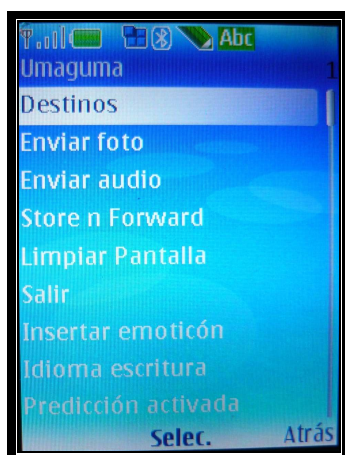


Figura 8.4 – Pantalla de opciones

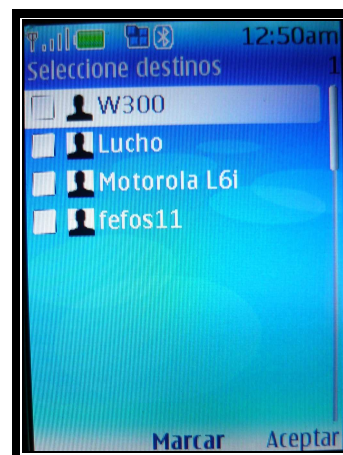


Figura 8.5 – Pantalla de selección de destinos

Dentro del menú de opciones también tenemos acceso a otras funcionalidades como por ejemplo el envío de mensajes Store and Forward, para esto se nos solicita ingresar la dirección MAC del destino. Esta dirección se debe ingresar como una palabra de 12 caracteres, no distinguiéndose entre mayúsculas y minúsculas.

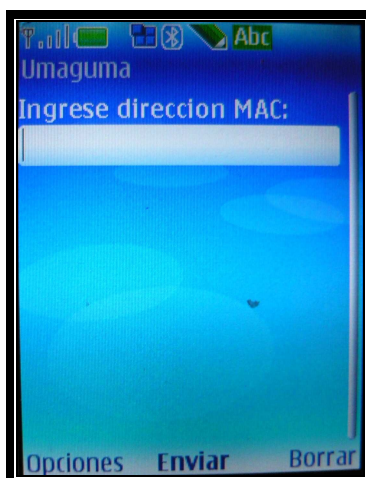


Figura 8.6 – Pantalla de ingreso de destinos para mensaje

A continuación se muestra el modo en el cual son recibidos los mensajes Store and Forward por el destinatario. Se observa una ventana emergente en la que se muestra el mensaje recibido junto con el nombre del remitente, así como también información de la fecha y hora de envío.

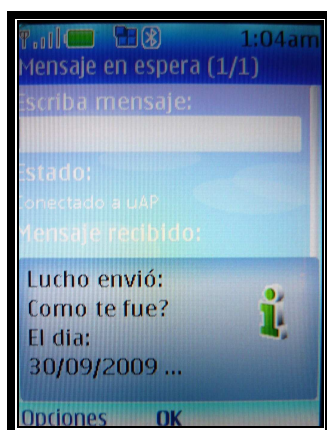


Figura 8.7 – Pantalla de recepción de mensaje del Store and Forward

De un modo similar al caso anterior (en una pantalla emergente) son desplegados los mensajes del sistema recibidos por el Terminal, estos mensajes pueden provenir tanto desde el punto de acceso como del servidor central, esto se indica en el título de la pantalla. En la imagen 8.8 se muestra el caso de un mensaje que proviene del uAP.

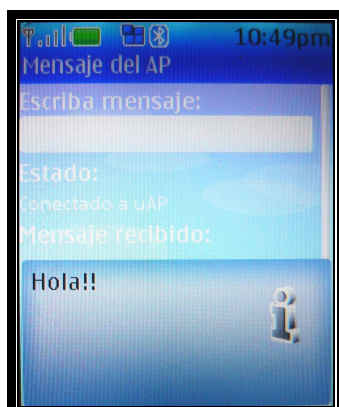


Figura 8.8 – Pantalla de recepción de mensaje del sistema

Los restantes botones del menú de opciones corresponden a los servicios de envío multimedia, en primer lugar tenemos la transmisión de imágenes por la red. Al seleccionar la opción “Enviar foto” accedemos a la cámara fotográfica del dispositivo (en caso de existir y estar disponible), con lo cual podemos capturar una imagen y luego enviarla a los destinos que se desee, estos la reciben junto con el nombre del remitente. A continuación se muestra el ejemplo de una imagen capturada, se observa el botón que permite realizar el envío.



Figura 8.9 – Pantalla de recepción de imagen

Por último tenemos el envío de mensajes de voz. Para hacer uso de este servicio se debe ingresar a la sección “Enviar audio” del menú de opciones, con esto se ingresa a la pantalla de captura de audio. Para comenzar con la grabación debemos seleccionar la opción “Grabar” y luego “Detener” cuando hayamos finalizado, a continuación se despliega la lista de destinos posibles para realizar el envío. La recepción se realiza

mediante una pantalla en la que se puede reproducir el mensaje tantas veces como se desee mediante el botón “Reproducir”.

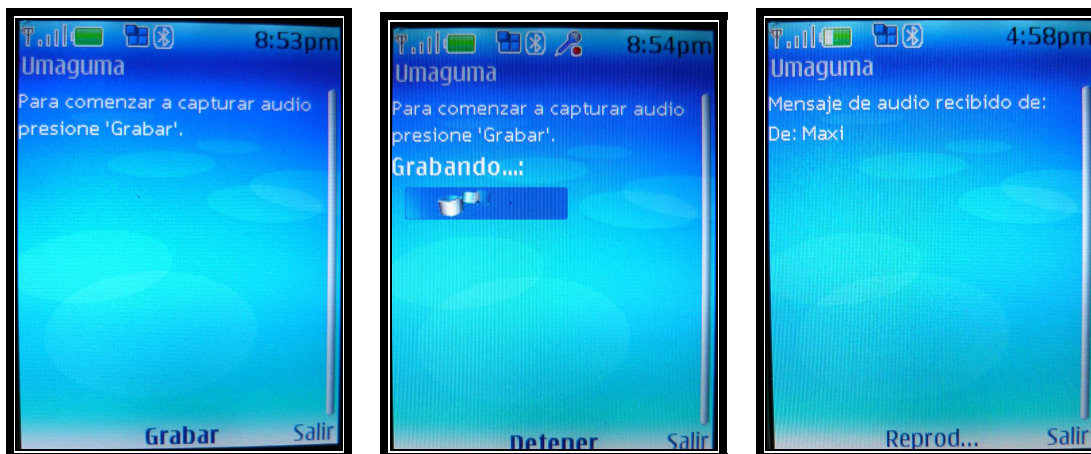


Figura 8.10 – Pantallas de grabación de mensaje de audio

Las imágenes mostradas pertenecen a capturas de la aplicación siendo ejecutada en un teléfono celular Nokia 6131.

8.3.2 Interfaz gráfica de Puntos de Acceso

Durante el desarrollo del proyecto la manera de desplegar información relevante por parte del software de los nodos fue cambiando, así como también la forma de interactuar con el usuario.

En un principio se utilizó únicamente la consola ordinaria del programa utilizado para el desarrollo del software (Eclipse), ya que en ese momento no se consideró necesario crear una interfaz amigable con el usuario debido a que el correcto desarrollo de las funcionalidades del nodo eran consideradas más relevantes.

Luego de haber logrado un correcto desempeño del nodo, se procedió a crear una interfaz gráfica simple la cual desplegó los usuarios conectados a la red por medio de ese uAP. Para esto se utilizaron las clases provistas por la librería AWT (Ver Anexo J2SE), la interfaz gráfica diseñada era muy simple ya que la misma desplegaba únicamente los parámetros de los usuarios conectados sin permitir que el usuario pudiese interactuar con la red.

Debido a la extrema simplicidad de esta interfaz grafica fue que se decidió desarrollar una mejor, la cual desplegara además de los usuarios conectados al uAP, graficas en vivo del tráfico que circula por el nodo, una consola la cual informara de eventos relevantes para el controlador del nodo, y la posibilidad de poder intervenir en la red.

A continuación se pasara a describir la interfaz grafica final desarrollada para este nodo. El orden en el cual se explicaran las diferentes ventanas, es el mismo orden con el que aparecen cuando se inicia el nodo uAP.

➤ Ventana de Inicio

La primera ventana desplegada cuando se inicia un nodo uAP, tiene como función permitir al controlador del nodo introducir ciertos datos relevantes para el funcionamiento del nodo.

La imagen a continuación es un ejemplo de esta ventana de inicio.

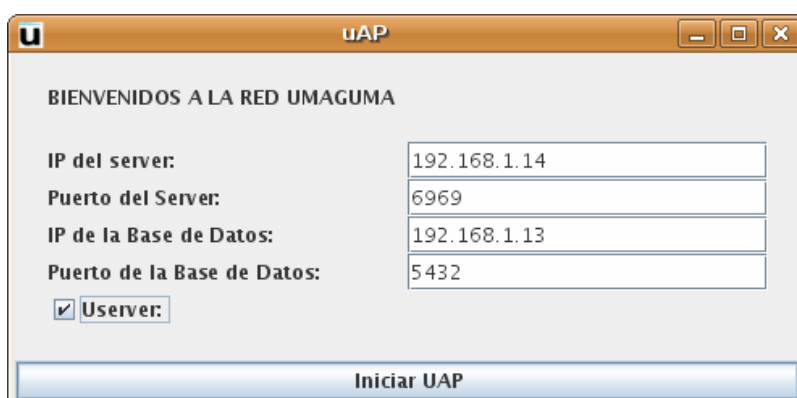


Figura 8.11 – Ventana inicial del uAP

Los datos solicitados por la ventana de inicio son detallados a continuación:

- **CheckBox uServer:** Es utilizado para indicarle al software cual será su modalidad de trabajo, en caso de que se encuentre marcado (Modalidad de Trabajo con uServer descrita en el capítulo 4), el usuario deberá rellenar el resto de los parámetros. En caso contrario (Modalidad de Trabajo Stand-Alone descrita en el capítulo 4), el resto de los parámetros no son necesarios.

- **IP del Server:** Ventana de texto en la cual se debe introducir la dirección IP del uServer, al cual se debe conectar el uAP. El formato de la dirección a introducir debe incluir los puntos (Ejemplo: **192.168.1.1**).
- **Puerto del Server:** Ventana de texto en la cual se debe introducir el puerto TCP del uServer al cual se conectara el uAP.
- **IP de la Base Datos:** Ventana de texto en cual se debe introducir la dirección IP de la Base de Datos a la cual el uAP le enviara las estadísticas del nodo. El formato a introducir debe incluir puntos. (Ejemplo: **192.168.1.1**).
- **Puerto de la Base de Datos:** Ventana de texto en la cual se debe introducir el puerto TCP de la base de datos.

Luego de haber introducido los datos necesarios para el funcionamiento del nodo uAP y haber presionado el botón “Iniciar uAP”, el software cierra esta ventana y despliega la ventana principal del mismo.

➤ **Ventana Principal**

Esta es la ventana principal del nodo, la misma despliega información de gran interés para la persona que se encuentra relevando el estado del mismo, esta también permite que el controlador pueda intervenir en la red ya sea por ejemplo desconectando usuarios o enviando mensajes a los mismos.

La imagen a continuación es un ejemplo de esta ventana principal.

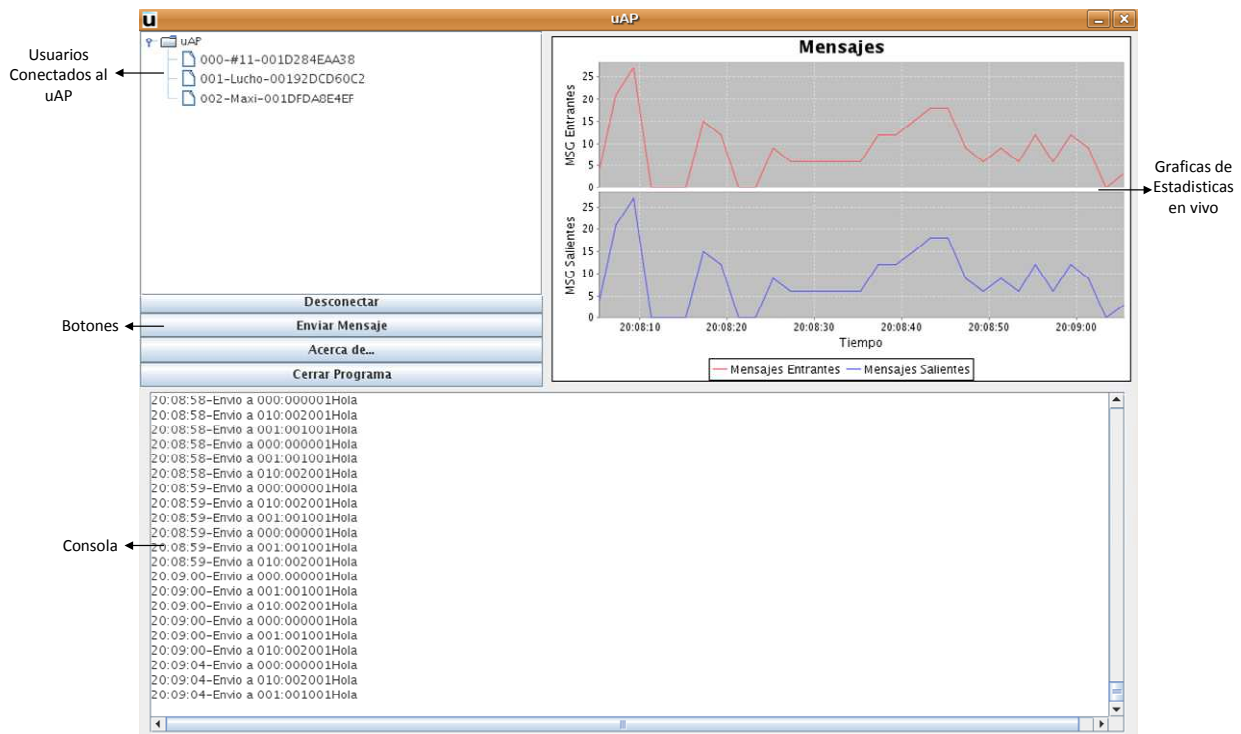


Figura 8.12 – Ventana principal del uAP

A continuación se detallará cada una de las zonas que se presentan en la imagen.

- **Usuarios Conectados al uAP:** Como se puede ver en la imagen, esta zona de la ventana muestra en forma de árbol desplegable cada uno de los usuarios que se encuentran conectados al uAP en ese momento, el árbol es actualizado de manera dinámica a medida que los usuarios se conectan o desconectan del uAP. Para cada uno de los usuarios conectados se tiene una fila la cual muestra Dirección Umagama Global, Friendly Name y dirección MAC del usuario.
- **Botones:** Debajo de la zona anterior se encuentran un conjunto de botones los cuales permiten al controlador de la red ejecutar acciones sobre los usuarios conectados, desplegar información relativa al proyecto o terminar el funcionamiento del uAP. A continuación se pasa a detallar la acción realizada al presionar cada uno de estos botones:
 - **Desconectar:** Este botón permite a la persona que se encuentra delante de la interfaz desconectar cualquier usuario conectado a

este uAP. Para esto primero se debe seleccionar un usuario en la sección de arriba y al presionar este botón el mismo es desconectado de la red Umaguma.

- **Enviar Mensaje:** Este botón permite enviar mensajes de texto a los usuarios conectados al uAP. Para esto el controlador de la red en primer lugar debe seleccionar un usuario o en caso de querer enviar un mensaje de broadcast debe seleccionar la fila donde dice “uAP”, todo esto en la sección de arriba. Al presionar este botón se le desplegara la siguiente ventana:



Figura 8.13 – Ventana de envío de mensajes del uAP

En la misma se debe introducir el texto a enviar y luego presionar el botón enviar, lo que hace que el mensaje sea enviado. Estos mensajes son enviados como mensajes especiales 251, lo que permite al software del usuario determinar que el mensaje provino desde el uAP. (En caso de dudas leer nuevamente el capítulo 3).

- **Acerca de:** Este botón despliega una ventana la cual contiene información sobre este proyecto y sus autores.
- **Cerrar Programa:** Al presionar este botón finaliza el programa, lo que causa la desconexión de los usuarios que se encontraban conectados a la red a través de este nodo uAP.
- **Consola:** En esta sección de la ventana se van desplegando eventos que suceden en el nodo, como por ejemplo ingreso de nuevos usuarios, mensajes de texto enviado por usuarios, mensajes que provienen desde el uServer, entre otros.
- **Graficas de Estadísticas en vivo:** En esta sección se tienen graficas, las cuales se van modificando a medida que transcurre el tiempo desplegando el grado de congestión del nodo. Esto permite al controlador del nodo poder percibir con una simple mirada cual es el nivel de tráfico en ese momento.

8.3.3 Interfaz gráfica del Servidor

➤ **Ventana de Inicio**

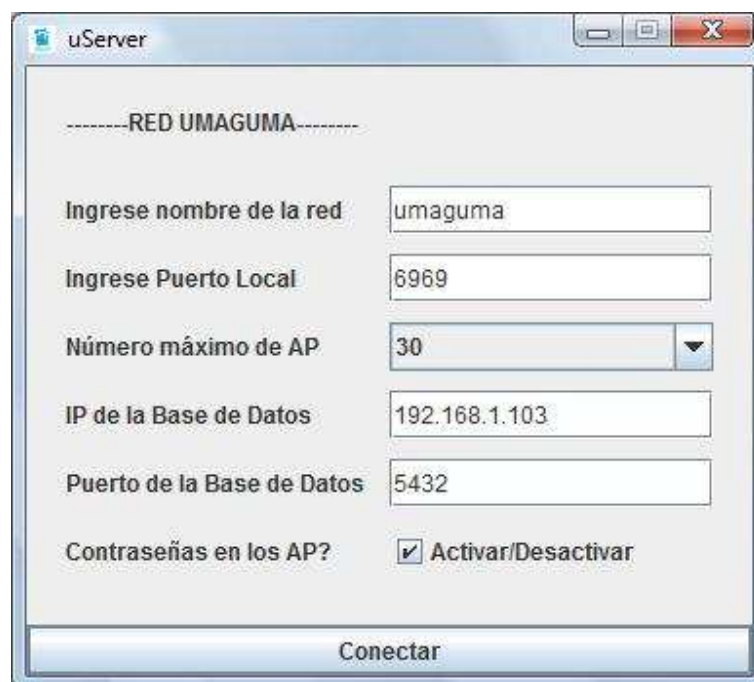


Figura 8.14 – Ventana inicial del uServer

Esta ventana es la primera en aparecer cuando se inicia el uServer.

Como ya se mencionó, permite al usuario configurar parámetros del uServer y de la red.

Estos parámetros son:

- **Nombre de la red:** Permite al usuario asignarle un nombre a la red.
- **Puerto Local:** Es el puerto en que se aceptarán las conexiones de los uAP.
- **Número Máximo de AP:** Permite al usuario seleccionar la cantidad máxima de Puntos de Acceso que se podrán conectar al Servidor.
- **IP de la Base de Datos:** Dirección IP a la cual el Servidor intentará establecer una conexión TCP/IP cuando necesite interactuar con la Base de Datos.

- **Puerto de la Base de Datos:** Puerto para establecer las conexiones TCP/IP con la Base de Datos.
- **Contraseñas en los AP?:** Activa o Desactiva el uso de contraseñas en los AP para restringir la conexión de Terminales a los mismos.
Si se activa (checkbox seleccionado), se ofrecerá la posibilidad de elegir configurar una contraseña o no para cada uAP que se conecte.
Si se desactiva (checkbox no seleccionado), quedará deshabilitada la opción de configurar contraseña para los uAP que se conecten.

Una vez que se ingresan los parámetros en su totalidad y se oprime el botón “Conectar”, la ventana de inicio se cierra dando lugar a la Interfaz principal del software uServer.

➤ **Ventana Principal:**

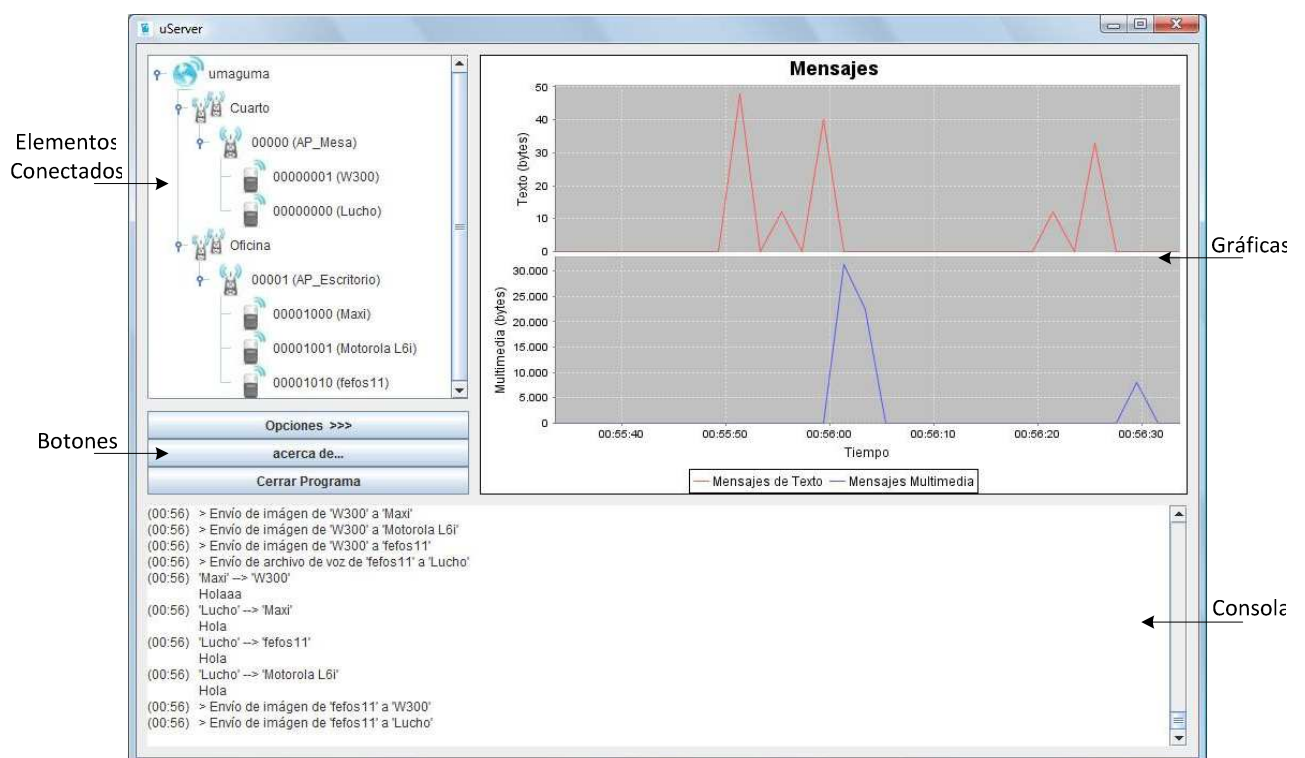


Figura 8.15 – Ventana inicial del uServer

Esta es la ventana principal del software uServer, y es desde donde el administrador podrá controlar el estado de la RED y tomar las acciones que crea necesarias.

Como se observa en la figura, la ventana principal del uServer tiene 4 zonas:

1. **Elementos Conectados:** En esta parte de la ventana se muestran los distintos elementos conectados a la RED en forma de árbol desplegable, representados de la siguiente manera:
 - El nodo principal del árbol es el uServer. Se muestra el nombre de la red.
 - Los nodos primarios son las Zonas en que está dividida la RED. Se muestra el nombre de cada Zona.
 - De cada una de las Zonas se desprenden los uAP que se encuentran en cada una de ellas. Se muestra la dirección umaguma de cada AP seguido del nombre entre paréntesis.
 - Los Terminales se desprenden del uAP al que están conectados. Se muestra la dirección umaguma de cada Terminal seguido del friendly name entre paréntesis.
2. **Botones:** Debajo del árbol utilizado para mostrar los elementos conectados, se encuentran 3 botones que permiten:
 - “Opciones >>>”: Realizar distintas acciones sobre el elemento seleccionado. (se estudia a continuación)
 - “acerca de...”: Mostrar una ventana conteniendo información sobre el proyecto umaguma.
 - (instancia de la clase AcercaDe)
 - “Cerrar Programa”: Salir del software uServer.
3. **Gráficas:** En esta zona de la ventana principal se encuentran 2 gráficas que muestran en tiempo real el tráfico de bytes, por el uServer, debido a mensajes de texto –gráfica superior- y a mensajes multimedia –gráfica inferior-.

4. **Consola:** La zona restante es un cuadro de texto (que no puede ser editable) en el cual se muestran distintos eventos por los que pasa el uServer, como la conexión o desconexión de Terminales o uAP, enrutamiento de mensajes de texto, enrutamiento de mensajes multimedia, etc.

➤ **Menú de Opciones:**

Como se mencionó en la descripción de la ventana principal, el botón “Opciones >>>” despliega un menú con diferentes opciones aplicables al elemento que se selecciono del árbol de elementos conectados.

Como se muestra a continuación, el menú a desplegar varía dependiendo del tipo de elemento seleccionado, ya que por ejemplo, las opciones sobre un Terminal o sobre un uAP serán diferentes.

	uServer	Zona	uAP	Terminal
Opciones >>>	Enviar Mensaje	Enviar Mensaje	Enviar Mensaje	Enviar Mensaje
acerca de...	Enviar Imagen	Enviar Imagen	Enviar Imagen	Enviar Imagen
Cerrar Programa	INFO	INFO	INFO	INFO
	Desconectar	Desconectar	Desconectar	Desconectar
	Limpiar Zonas	Eliminar Zona	Visibilidad	

Las distintas opciones que aparecen en el menú son:

- **Enviar Mensaje:** Permite enviar un mensaje de texto al elemento seleccionado. (se describe más adelante)
- **Enviar Imagen:** Permite enviar una imagen al elemento seleccionado. (se describe más adelante)
- **INFO:** Muestra una ventana con información pertinente del elemento seleccionado. (se describe más adelante)
- **Desconectar:** Permite desconectar cualquier elemento seleccionado, haciendo la siguiente distinción según el tipo de elemento seleccionado:
 - **uServer:** desconecta a todos los uAP del Servidor.

- **Zona:** desconecta del uServer a todos los Puntos de Acceso de dicha Zona
- **uAP:** desconecta al AP en cuestión del uServer.
- **Terminal:** desconecta al Terminal de la RED (por medio del uAP al que se encuentra conectado dicho Terminal).
- **Limpiar Zonas (solo uServer):** Elimina todas las Zonas en las que no se encuentra ningún uAP conectado.
- **Eliminar Zona (solo Zona):** Elimina la Zona seleccionada, siempre que no haya ningún uAP conectado a dicha Zona.
- **Visibilidad (solo uAP):** Permite configurar las políticas de Visibilidad del uAP seleccionado. (se describe más adelante)

➤ **Ventana Nuevo AP:**

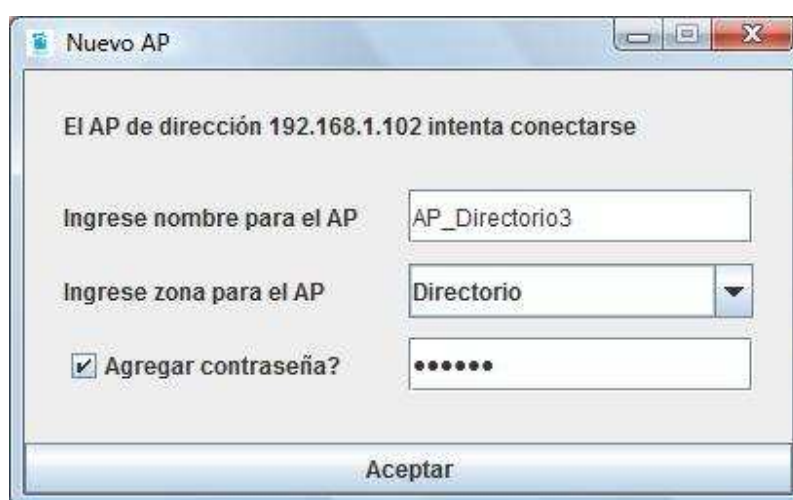


Figura 8.16 – Ventana de Nuevo AP

Es la ventana que se muestra cada vez un uAP intenta conectarse al uServer y permite configurar ciertos parámetros del nuevo Punto de Acceso.

Como se observa en la figura, cuenta con un encabezado que despliega la dirección IP del AP que intenta conectarse al Servidor y algunos campos para que el administrador ingrese los siguientes datos:

- **Nombre del AP:** Permite al administrador asignarle un nombre al nuevo Punto de Acceso.
- **Zona para el AP:** Campo para asignar el AP a una Zona. El administrador cuenta con la opción de elegir una de las Zonas ya existentes desplegando el ComboBox o ingresar una nueva Zona simplemente digitando su nombre.
- **Agregar contraseña?:** Da la opción de agregar o no contraseña al nuevo AP. Como se mencionó en la descripción de la ventana de inicio, este CheckBox solo estará habilitado si se selecciono el CheckBox “Contraseñas en los AP?” en dicha ventana.

Si se selecciona el CheckBox, aparecerá a su derecha –como en el ejemplo– un campo de texto para ingresar la contraseña. En caso de que no se seleccione el CheckBox, dicho campo de texto no aparecerá.

➤ **Ventana Enviar mensaje:**



Figura 8.17 – Ventana Enviar mensaje

Esta ventana aparece cada vez que se selecciona la opción “Enviar Mensaje” en la interfaz principal.

Como se observa, cuenta con 4 componentes, un cuadro de texto en el cual el administrador ingresa el mensaje a enviar, 2 botones –uno para enviar el mensaje y otro para cancelar la acción- y un encabezado, que indica a quién va dirigido el mensaje y dependerá del nodo del árbol que se haya seleccionado. El mensaje puede estar dirigido a toda la RED (como en el ejemplo), a una Zona, a un AP o a un Terminal.

➤ **Ventana Enviar Imagen:**



Figura 8.18 – Ventana Enviar Imagen

Similar a la ventana para enviar mensajes, esta ventana aparece cada vez que se selecciona “Enviar Imagen” desde la ventana principal.

También cuenta con un encabezado, que indica el destino de la imagen y varía de acuerdo al tipo de elemento seleccionado en la interfaz principal, un botón para enviar la imagen y otro para cancelar la acción, y en el centro de la ventana cuenta con un botón que permite explorar en busca de la imagen a enviar –a la derecha- y una etiqueta que muestra el nombre y la ruta de la imagen seleccionada.

➤ **Ventana Visibilidad:**



Figura 8.17 – Ventana Visibilidad

Al igual que las anteriores, esta ventana aparece cuando se selecciona la opción “Visibilidad” en el menú principal, pero con la diferencia de que dicha opción estará habilitada solamente al seleccionar un uAP.

Esta interfaz permite configurar las políticas de Visibilidad, mencionada en el capítulo anterior, de cada Punto de Acceso.

Para ello, cuenta con los siguientes componentes (de arriba hacia abajo):

Encabezado: Muestra la dirección y el nombre del uAP (entre paréntesis) que está siendo modificado.

TextField “de filtrado”: Permite filtrar las direcciones de los Puntos de Acceso que serán mostrados.

En el ejemplo, se ingresó “000”, de manera que solo se mostrarán los AP que comiencen con dicha combinación de números.

Visibilidad del resto de los AP: Aparece en el centro de la pantalla, y consta de iconos indicando con un tick o una cruz la visibilidad del uAP que aparece a la derecha de cada icono.

En el ejemplo, se está configurando la Visibilidad del uAP AP_Directorio1, y se observa que puede comunicarse con el AP AP_Directorio2, pero no puede hacerlo con los Puntos de Acceso AP_Cantina1 ni AP_Cantina2.

Para cambiar la Visibilidad de un uAP basta con clickear arriba del icono o del nombre de dicho AP.

Botones “Modificar” y “Cancelar”, utilizados para aplicar los cambios o para cancelarlos, respectivamente.

➤ **Ventana INFO:**

Las ventanas que se ilustran a continuación, son creadas cada vez que se selecciona la opción "INFO" en el menú de opciones.

Como se puede observar, el fin de cada una de las ventanas es mostrar al administrador información relevante sobre cada uno de los elementos.

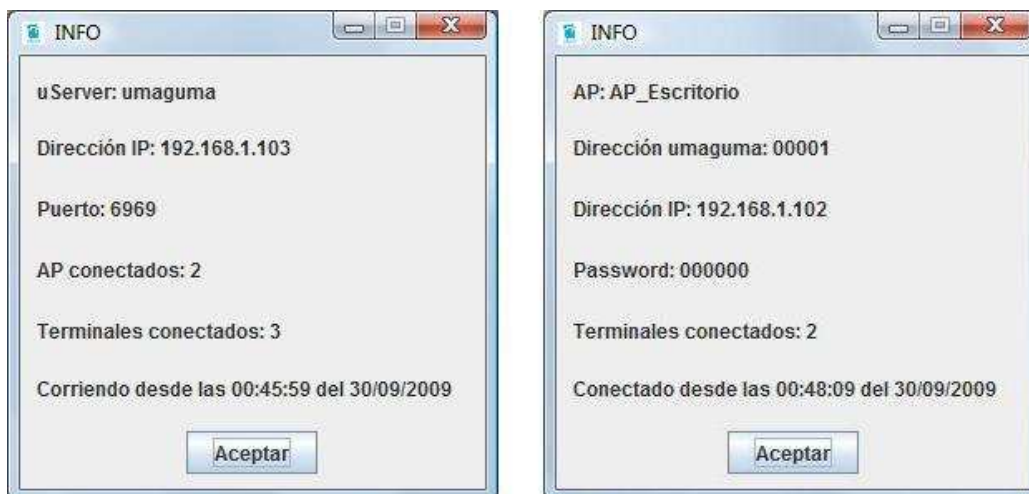


Figura 8.18 – Ventanas de información de uServer y uAP



Figura 8.19 – Ventanas de información de una zona y un Terminal

8.3.4 Interfaz gráfica del Gestor de Estadísticas

Al ejecutar el software Gestor de estadísticas, en una primera etapa se debería ingresar los datos de la base de datos que almacena la información que los nodos de la red reportan cada 15 minutos.

Los datos a ingresar son IP y puerto de la base de datos, junto con su respectiva contraseña.



Imagen 8.20 – Ventana inicial del Gestor de Estadísticas

En esta misma pantalla debe seleccionarse el tipo de nodo (Access Point o Servidor) al que se desea analizar estadísticamente, pulsando en el botón correspondiente.

En función de la opción seleccionada, surgirá una nueva pantalla. En el caso de haber sido seleccionado “Ver estadísticas del Servidor”, se solicitará en la nueva pantalla que se ingrese la fecha y hora, tanto de inicio como de fin del período en el cual se analizarán las estadísticas del nodo.



Imagen 8.21 – Ventana de selección de estadísticas de Servidor

Seleccionado el período de análisis estadístico, se pulsa el botón “Ver tabla de estadísticas” lo que hará que se presenten al usuario las correspondientes tablas con los datos almacenados en la base de datos, previamente filtrados por el criterio que él mismo estableció.

Esta tabla presenta los campos que fueron explicados en el capítulo 6, con sus respectivos datos obtenidos en cada reporte que se almacenó cada 15 minutos.

Fecha/Hora	MSG Txt	MSG Img	MSG Voz	MSG E S&F	MSG R S&F	MSG A S&F	Bytes Txt	Bytes Img	Bytes Voz	Bytes E S&F	Bytes R S&F	Bytes A S&F	Tot. bytes R	Tot. bytes E
20090930 00:48...	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20090930 00:49...	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20090930 00:50...	0	1	4	0	0	0	0	3448	22588	0	0	0	26036	26036
20090930 00:51...	21	1	2	0	0	0	354	2579	12574	0	0	0	15507	15507
20090930 00:52...	59	0	2	0	0	0	716	0	7198	0	0	0	7914	7914
20090930 00:53...	4	0	0	0	0	0	52	0	0	0	0	0	52	52
20090930 00:54...	36	3	2	0	0	0	387	11031	37534	0	0	0	48952	48952
20090930 00:55...	6	0	0	0	0	0	100	0	0	0	0	0	100	100
20090930 00:56...	4	5	2	0	0	0	45	16938	44638	0	0	0	61621	61621
20090930 00:57...	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20090930 00:58...	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20090930 00:59...	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20090930 01:00...	0	1	0	0	0	0	0	3168	0	0	0	0	3168	3168
20090930 01:01...	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20090930 01:02...	0	0	0	2	0	2	0	0	0	67	0	31	0	67
20090930 01:03...	0	2	0	0	2	0	0	6382	0	0	121	0	6503	6503
20090930 01:04...	0	0	2	0	0	0	0	0	45342	0	0	0	45342	45342
20090930 01:05...	16	3	0	0	0	0	144	8748	0	0	0	0	8892	8892
20090930 01:06...	24	0	2	0	0	0	284	0	8990	0	0	0	9254	9254
20090930 01:07...	48	0	2	1	0	1	792	0	18	30	0	12	810	840
20090930 01:08...	204	0	3	0	0	1	3672	0	25517	0	0	12	29189	29189
20090930 01:09...	180	3	0	0	0	1	3240	11133	0	0	0	12	14373	14373
20090930 01:10...	0	0	1	0	0	1	0	0	11663	0	0	12	11663	11663
20090930 01:11...	0	0	1	0	1	0	0	0	19727	0	56	0	19783	19783
20090930 01:12...	0	2	1	0	0	0	0	5166	5391	0	0	0	10557	10557
20090930 01:13...	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Referencias: MSG = Mensajes Txt = Texto Img = Imagenes
E = Enviados R = Recibidos A = Acumulados
S&F = Store and Forward Tot. = Total

Imagen 8.22 – Ventana con tabla de estadísticas de Servidor

En la parte inferior de la tabla, se puede observar el botón “Ver Graficas”. Pulsando dicho botón se desplegará en una nueva ventana la cual contiene todas las gráficas que también fueron detallados en el capítulo 6.

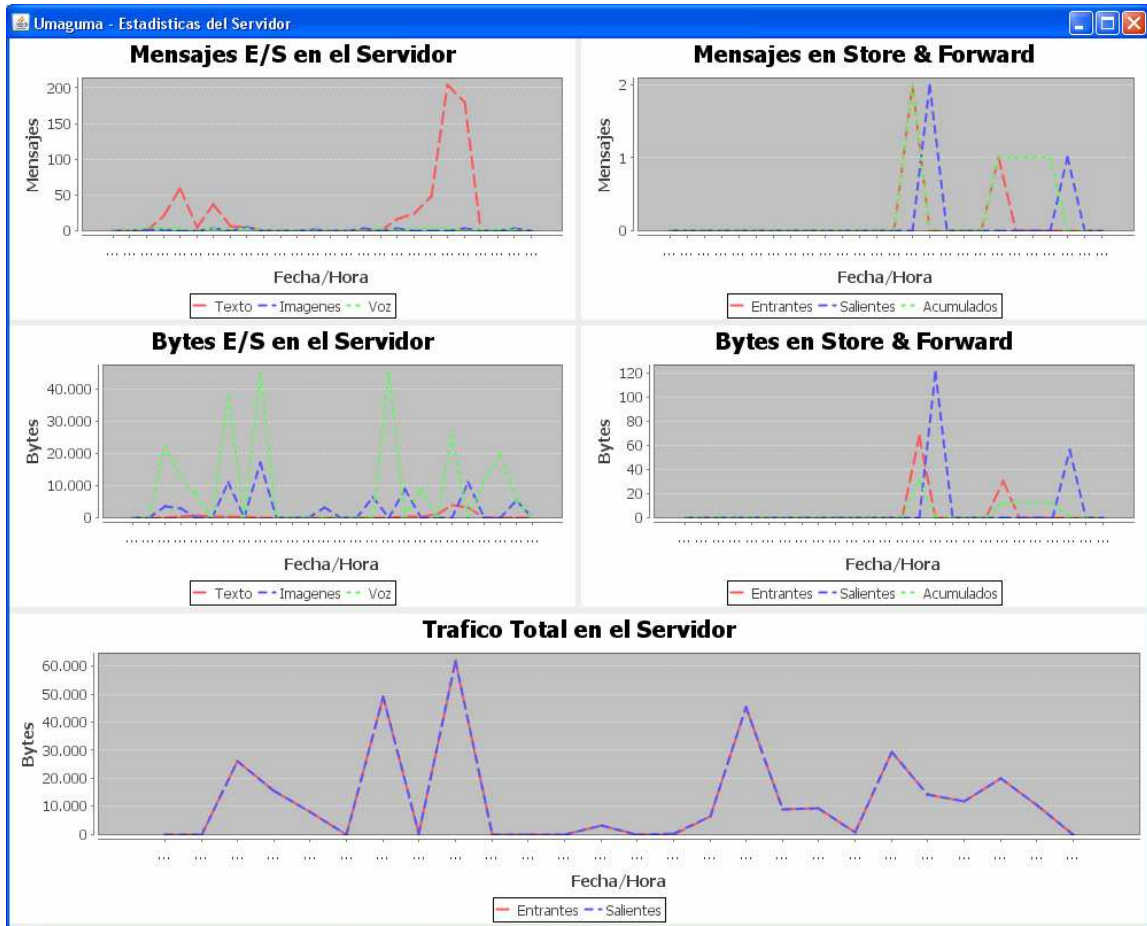


Imagen 8.23 – Ventana de gráficas de estadísticas de Servidor

En analogía con lo presentando anteriormente, si en la pantalla inicial se opta por pulsar el botón “Ver Estadísticas de los AP”, aparecerá una nueva pantalla muy similar a la desplegada si se oprime el otro botón, pero con la particularidad de que al existir más de un Access Point se da la posibilidad de establecer mas restricciones.



Imagen 8.24 – Ventana de selección de estadísticas de los Access Points

Esta restricción adicional supone la posibilidad de analizar estadísticamente a todos los Access Points dentro del período establecido o de solo estudiar uno en particular.

Si se desea analizar las estadísticas de un AP en particular, debe indicarse la dirección de este en el campo junto a “Dirección de AP”.

Por el contrario, si lo requerido es realizar un análisis de todos los Access Points, se debe tildar la casilla que se encuentra junto al texto “Todos los APs”. Esta acción hará que instantáneamente se anule el campo de “Dirección de AP” ya que pasaría a perder sentido el llenado de este campo.

Establecidos los criterios, a consecuentemente se debe pulsar el botón “Ver tablas de estadísticas” para observar los datos recopilados.

Estadísticas por uAP - umaguma												
Fecha/Hora	Dir. AP	MSG E/S Txt	MSG E/S Img	MSG E/S Voz	MSG Totales	Bytes E/S Txt	Bytes E/S Img	Bytes E/S Voz	Bytes TX local	Bytes TX serv.	Tot. Bytes TX	
20090930 00:49:00	00000	0	0	0	0	0	0	0	0	0	0	
20090930 00:50:00	00000	0	1	0	1	0	2710	0	2710	0	2710	
20090930 00:51:00	00000	2	2	4	8	20	6896	22588	3448	15254	18702	
20090930 00:52:00	00000	20	1	2	23	344	2579	12574	10	2913	2923	
20090930 00:53:00	00000	74	0	2	76	866	0	7198	150	716	866	
20090930 00:54:00	00000	4	0	0	4	52	0	0	0	0	0	
20090930 00:55:00	00000	37	4	2	43	402	14708	37534	3692	11076	14768	
20090930 00:56:00	00000	7	4	2	13	116	11896	44638	2990	8970	11960	
20090930 00:57:00	00000	5	0	0	5	56	0	0	11	33	44	
20090930 00:58:00	00000	0	1	0	1	0	2959	0	2959	0	2959	
20090930 00:59:00	00000	0	0	0	0	0	0	0	0	0	0	
20090930 01:00:00	00000	0	1	0	1	0	5228	0	5228	0	5228	
20090930 01:01:00	00000	0	2	0	2	0	6336	0	3168	3168	6336	
20090930 01:02:00	00000	0	0	0	0	0	0	0	0	0	0	
20090930 01:03:00	00000	1	0	0	1	30	0	0	0	30	30	
20090930 01:04:00	00000	0	3	0	3	0	9573	0	3191	6382	9573	
20090930 01:05:00	00000	0	0	2	2	0	0	45342	0	0	0	
20090930 01:06:00	00000	16	4	0	20	144	11664	0	2916	8748	11664	
20090930 01:07:00	00000	36	0	2	38	396	0	8990	132	264	396	
20090930 01:08:00	00000	150	0	2	152	2640	0	18	846	1740	2586	
20090930 01:09:00	00000	225	0	4	229	4050	0	34236	10069	20138	30207	
20090930 01:10:00	00000	334	5	0	339	6012	18289	0	9928	14373	24301	
20090930 01:11:00	00000	0	0	2	2	0	0	23326	11663	11663	23326	
20090930 01:12:00	00000	1	0	1	2	56	0	19727	0	19727	19727	
20090930 01:13:00	00000	0	0	1	1	0	0	5391	0	5391	5391	
20090930 01:14:00	00000	0	0	1	1	0	0	7183	0	7183	7183	

Referencias: MSG = Mensajes Txt = Texto Img = Imágenes
Dir. AP = Dirección del AP E/S = Entrada/Salida Tot. = Total
TX local = Transmítidos localmente TX serv. = Transmítidos al Servidor

[Ver Gráficas](#)

Imagen 8.25 – Ventana con tabla de estadísticas de Access Point

Al igual que con la tabla presentada para el caso de las estadísticas del Servidor, se presenta en la parte inferior de la tabla la opción de visualizar los diferentes gráficos que se explicaron en el capítulo 6.

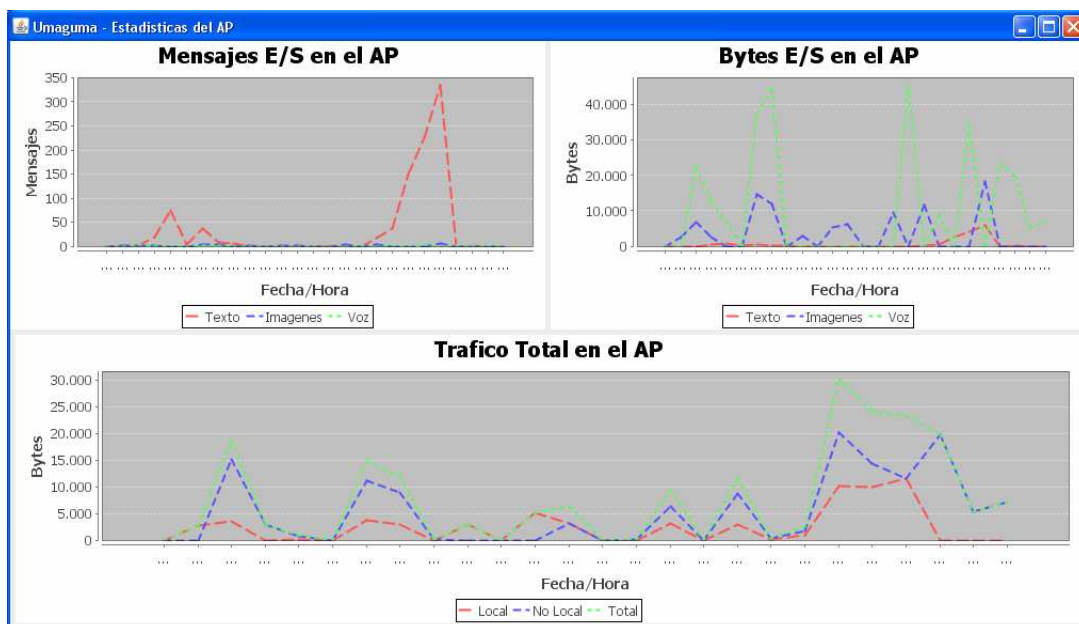


Imagen 8.26 – Ventana de gráficos de estadísticas de Access Point

Sólo restaría hacer notar que en el caso de que se opte por la visualización de las tablas de estadísticas de todos los Access Points en simultáneo dentro del período seleccionado, no se presentará la opción de observar gráficos (no aparecerá dicho botón en la parte inferior de la tabla) ya que haría poco clara la visión de varios gráficos en simultáneo. Considerando además que al hacer una distinción de tipo de mensajes en cada gráfica (mediante la representación en diversos trazos), el tráfico de cada Access Point es representado en un mismo gráfico por tres trazos diferentes, por no sería una buena opción representar a más de un AP en una misma gráfica, ya que sería dificultosa la visualización de los detalles.

Capítulo 9: Resultados

9.1. Introducción

Habiendo concluido las etapas de desarrollo y pruebas de nuestro sistema, estamos en condiciones de poner sobre la mesa los resultados recogidos y hacer un análisis objetivo al respecto. Debemos entonces hacer un recuento de las características de la implementación final del proyecto y compararlas con lo que fue nuestra idea original, evaluando si se alcanzaron los objetivos planteados y en que medida se logro esto.

Posteriormente analizaremos el desempeño que tuvo nuestro sistema frente a pruebas en las que se intentó evaluar todas sus funcionalidades para distintos escenarios, llevándolo incluso a límites de carga en los cuales el sistema podría fallar o eventualmente mostrar debilidades.

9.2. Objetivos alcanzados

En esta sección analizaremos los resultados funcionales obtenidos en nuestro sistema, comparándolo con lo planificado inicialmente, particularmente prestando atención al alcance definido para el proyecto y evaluando el éxito de la implementación final.

Se logró implementar exitosamente una red capaz de interconectar terminales móviles utilizando Bluetooth como red de acceso y TCP/IP como red de núcleo. Para esto se desarrolló todo el software necesario, llevando a la práctica los diferentes nodos de la red. Se diseñó y desarrolló un protocolo de comunicación que maneje todo el control del sistema a lo largo de todas las etapas. Se diseñaron interfaces gráficas de usuario para cada uno de los nodos, a través de los cuales el administrador maneja la gestión del sistema y el cliente hace uso de él. También logramos llevar a la práctica servicios que se ejecutan sobre la red implementada, como son el intercambio de mensajería instantánea con sesión de usuario, la transmisión de imágenes, grabación y envío de mensajes de audio y el envío de mensajes Store and Forward. Con todo esto estaríamos alcanzando exitosamente los requisitos planteados inicialmente como alcance del proyecto, con lo que estaríamos en condiciones de considerarlo exitoso.

A continuación haremos un resumen de los detalles de la implementación final que no se encontraban establecidos de manera definitiva en el comienzo. Como primera característica a destacar podemos mencionar que el mecanismo de inicio de sesión se

implementó utilizando una lista fija de puntos de acceso cargados en el software de antemano, no permitiendo la búsqueda de dispositivos. Quedó implementado también el mecanismo de autenticación mediante contraseñas. Cada vez que un nuevo punto de acceso ingresa a la red, el uServer le indica si debe o no solicitar contraseña y -en caso afirmativo- cual debe ser. Cuando un terminal desee conectarse a este punto de acceso se le solicitara el ingreso de contraseña, otorgándole tres posibilidades para hacerlo, de no ingresarla correctamente la conexión se dará por terminada.

Se llevó a cabo el sistema de mensajería mediante listas de destinos que se actualizan dinámicamente a medida que los usuarios entran y salen de la red. En cualquier momento un usuario puede acceder a la lista y seleccionar uno o varios destinos a los cuales desea que se le envíe el mensaje escrito, para esto debe utilizar la opción “Enviar”. Cada vez que un mensaje se recibe es desplegado junto con el remitente correspondiente y la numeración que indica el orden de llegada. Si se desea enviar el mensaje en modalidad Store and Forward se debe ingresar la dirección MAC del destino. El destinatario recibe -al conectarse- los mensajes Store and Forward almacenados en la base de datos junto con información del remitente, fecha y hora de envío.

Se consiguió implementar un sistema de políticas mediante el cual el administrador de la red es capaz de establecer la visibilidad de los usuarios desde un punto de acceso hacia otro, aislando o no determinadas zonas según lo requiera la aplicación. Esto actualiza dinámicamente las listas de destinos de cada uno de los nodos de la red. También se llevaron a cabo exitosamente mecanismos para enviar a través de la red mensajes con grabaciones de voz y fotografías tomadas desde los terminales. El sistema dispone además de un sistema de reportes estadísticos que funciona tanto dinámicamente (en cada nodo) como en modalidad de resumen a nivel de toda la red.

El sistema se implementó manejando la posibilidad de funcionar con la red completa (todos los nodos) o utilizando solo un punto de acceso en modalidad Standalone, que se comporta como conmutador local sin la necesidad de interactuar con el servidor y base de datos. Esto también permite que el sistema sea capaz de soportar fallas en la conectividad IP, conmutando desde la situación normal hacia la modalidad Standalone.

Con estos resultados a la vista, estamos en condiciones de afirmar que los objetivos planteados para el proyecto han sido alcanzados exitosamente, llegándose a realizar agregados al sistema que originalmente no estaban planeados como son el manejo de estadísticas, el envío de mensajes de audio y el manejo de políticas de visibilidad.

9.3. Pruebas de desempeño

Durante todo el transcurso de la etapa de desarrollo se realizaron diversas pruebas sobre el sistema para evaluar su desempeño, buscando poner a prueba los pequeños avances que el proyecto alcanzaba día a día. Estas pruebas fueron de gran utilidad para poner en evidencia las distintas problemáticas que surgían y realizar de esta manera un proceso de debugging del proyecto hasta lograr que cada funcionalidad anduviera correctamente. Esto tenía sus instancias más relevantes en las reuniones periódicas que mantuvo el grupo completo, en las cuales se buscaba integrar todo lo desarrollado durante los días previos y fijando los lineamientos para los próximos avances.

Una vez finalizado este proceso de refinamiento del software se procedió a realizar las pruebas finales sobre el sistema -aparentemente- terminado, intentando evaluar los puntos clave y verificando que todo funcionase correctamente aún en condiciones extremas de tráfico. Para realizar los experimentos se desplegó una LAN compuesta por 4 computadoras conectadas a un Switch Ethernet mediante cables Patchcord. Una de las computadoras hacía las veces de servidor mientras que en otra se corría la base de datos y el software de reporte para informes estadísticos. Las dos computadoras restantes se utilizaron como puntos de acceso. Se utilizaron también en las pruebas un total de 7 teléfonos celulares que tenían instalado el software del uCel.

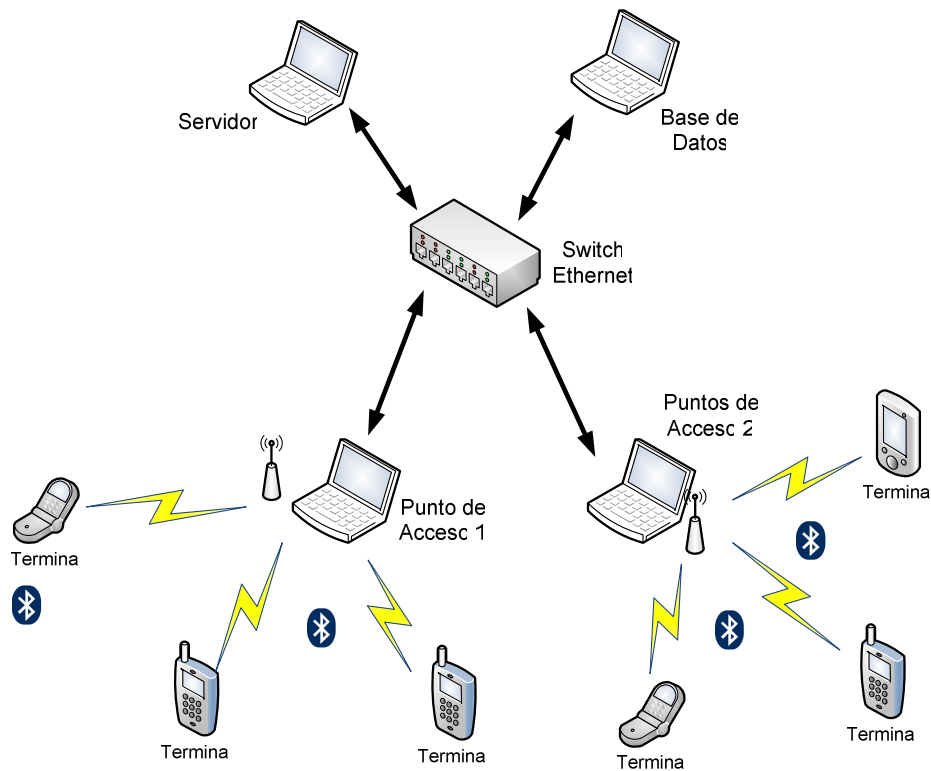


Figura 9.1 – Diagrama de la red de pruebas

En primer lugar se probó el funcionamiento del inicio de sesión, realizando para esto conexiones desde distintos terminales hacia los dos puntos de acceso disponibles, intentando cubrir todos los casos de uso posibles. En particular nos interesaba observar el funcionamiento de la red ante solicitudes de conexión simultáneas. Para todos los casos observamos que la conexión se establecía correctamente, actualizando al instante la lista de destinos alcanzables de todos los celulares conectados. También verificamos en esta instancia el correcto funcionamiento de la desconexión de terminales, tanto viniendo desde el usuario como desde el sistema, retirando instantáneamente al usuario saliente de las listas de todos los usuarios conectados.

Se probó luego el funcionamiento del mecanismo de inicio de sesión mediante ingreso de contraseñas, obteniendo resultados exitosos y funcionando todo de manera fluida, comprobamos que tras el tercer desacierto en la contraseña la sesión expira y el terminal es automáticamente desconectado. Verificamos también que si durante el transcurso de una sesión común se pierde la comunicación con el uAP (por ejemplo por una falla en la computadora que lo ejecuta), el terminal es capaz de detectar la falta de conectividad (al cabo de pocos segundos) dando -él mismo- por finalizada la comunicación y avisándole al usuario sobre la caída.

Como siguiente prueba realizamos el siguiente experimento: conectamos varios terminales a cada uno de los puntos de acceso y de manera abrupta, cortamos la comunicación de uno de los uAP con el servidor. Como resultado obtuvimos que el punto de acceso que quedó aislado del resto cambió automáticamente su modalidad, pasando a funcionar en Standalone como conmutador local (sin uServer), con lo cual los usuarios que se encontraban conectados a él no perdieron su sesión, simplemente dejaron de visualizar en su lista de destinos a los usuarios del otro uAP, manteniendo a los que comparten la zona con él. Para el punto de acceso que continuó conectado a la red la situación no se vio modificada, salvo por el hecho de que perdió de vista a los destinos que se encontraban en la otra zona de la red.

Luego le llegó el turno a las pruebas sobre políticas de visibilidad, para esto conectamos nuevamente varios terminales a cada uno de los dos puntos de acceso disponibles, y luego -desde la interfaz del uServer- quitamos la visibilidad mutua a los uAP. Observamos inmediatamente que las listas de destinos de todos los celulares se actualizaron mostrando únicamente a los usuarios visibles según las políticas. Luego restituimos la situación original de visibilidad y las listas de destinos se actualizaron correctamente, con lo que pudimos verificar el buen funcionamiento de esta aplicación. También probamos el envío de mensajes del sistema a los celulares, tanto desde el uAP como del uServer, obteniéndose buenos resultados, los terminales recibieron un mensaje en forma de alerta (pop-up) con el contenido correcto, y en el título se indicaba quien enviaba el mensaje.

Una vez realizados todos estos estudios nos disponemos a probar el desempeño del sistema frente a pruebas de carga, para esto hicimos circular tráfico de la manera más intensa posible a través de los nodos. La idea concretamente fue conectar a la red tantos celulares como fuera posible (7 en total), distribuyéndolos entre los dos puntos de acceso disponibles, para luego comenzar a enviar repetidamente desde cada Terminal mensajes hacia todo el resto de los usuarios. Esto se realizó durante un período de aproximadamente un minuto, observándose luego la respuesta del sistema. En un principio, los mensajes arribaban a sus respectivos destinos de manera inmediata, sin perderse ninguno en el camino, lo cual mostraba un muy buen funcionamiento. Con el pasar de los segundos (y de los mensajes) se comenzó a notar que los mensajes llegaban a sus destinos con un pequeño retraso. Luego, al terminar el minuto de envío se pudo observar que continuaron llegando mensajes a todos los celulares por aproximadamente medio minuto más, observándose también que se habían recibido el total de mensajes enviados. Esto nos permitió concluir que -al menos a priori- la red se comporta de manera robusta ante grandes cargas de tráfico, a menos de un pequeño retraso en la recepción para casos extremos, y que no se producen pérdidas en los mensajes. Mediante una prueba similar pero cambiando el contenido del mensaje transmitido pudimos constatar que la red también mantiene el orden de transmisión de los mensajes.

Continuando con las pruebas de desempeño, nos abocamos a probar el envío de mensajes multimedia a través de la red, comenzando con la captura de imágenes. Hicimos pruebas con todos los celulares disponibles, consiguiendo satisfactoriamente tomar fotos y enviarlas a todos los destinos que se encontraban disponibles en la red, los cuales fueron capaces de visualizar sin problemas las imágenes recibidas. La transmisión de la foto se dio con una demora del orden de los 2 segundos y en todos los casos la recepción se efectuó correctamente. Para el caso de las grabaciones de voz no hubo problemas de compatibilidad en la grabación del mensaje, no así en la reproducción del mensaje recibido. Si bien todos los terminales utilizados para la prueba fueron capaces de capturar, codificar, transmitir y recibir los mensajes de audio, hubo dos modelos en los cuales no se consiguió efectuar la reproducción del mensaje recibido. Lo que se observó fue que al presionar el comando "Reproducir" no se logró escuchar audio alguno, ni tampoco fue desplegado ningún mensaje de error por parte del dispositivo. Los teléfonos en los que ocurrió esto fueron el Sony Ericsson W810 y el Nokia E51. La transmisión -también en este caso- presenta una demora del orden de los 2 segundos, dependiendo del largo del mensaje grabado. Se realizaron pruebas con capturas de hasta 20 segundos sin presentarse problemas en la transmisión, tiempo más que suficiente para enviar un mensaje de voz concreto.

Ya acercándonos al final de esta etapa de pruebas le toca el turno al envío de mensajes Store and Forward, para probar esto escribimos desde varios teléfonos distintos mensajes en diferentes momentos, para luego enviarlos -en modalidad Store and Forward- hacia un mismo destino que en aquel momento se encontraba desconectado de la red, para esto hubo que introducir su dirección MAC. Cuando este Terminal inicio sesión

instantáneamente recibió la totalidad de mensajes enviados anteriormente, con su correspondiente remitente, fecha y hora correctamente desplegados. Los mensajes son mostrados secuencialmente. Con esto verificamos el correcto funcionamiento de este servicio.

Por último debemos mencionar que durante todo el proceso de pruebas descrito los indicadores de tráfico presentes en los uAP y uServer se mantuvieron trabajando correctamente, reflejando de manera dinámica el estado del nodo para cada instante de tiempo. También fueron probados para este proceso los reportes estadísticos desarrollados para el sistema. La base de datos se mantuvo capturando información de los restantes nodos y luego de las pruebas pudimos visualizar correctamente (al menos en apariencia) los gráficos que mostraban el tráfico circulante por toda la red.

Capítulo 10: Conclusiones

Llegado este punto, hemos superado las etapas de planificación, diseño, desarrollo y pruebas de nuestro sistema, por lo que estamos en condiciones de evaluar en su totalidad el proyecto y realzar las conclusiones correspondientes sobre los resultados obtenidos y sobre el desarrollo general del proyecto.

Como principal conclusión podemos destacar que el proyecto fue totalmente exitoso en sus resultados, desde el momento que logró cumplir en tiempo y forma todos los objetivos planteados inicialmente, objetivos que fueron definidos como parte del plan de trabajo y que se incluyen en este documento en la sección "Alcance del proyecto". Se logro diseñar exitosamente una red celular multiservicio que cumple con los requisitos de funcionamiento planteados. Desarrollamos un protocolo de comunicación capaz de interconectar diferentes nodos y brindar los servicios que originalmente habíamos planeado. Logramos desarrollar e implementar un prototipo de la red que funcionó a la perfección en todos los requerimientos planteados, incluso superando lo planificado inicialmente. Se desarrollaron interfaces gráficas de usuario que permiten un manejo intuitivo de las aplicaciones.

En cada una de las etapas del proyecto se logró cumplir con el cronograma de trabajo establecido inicialmente, respetando los objetivos propuestos para cada uno de los entregables en tiempo y forma. Se realizaron 2 presentaciones intermedias mostrando ante otros grupos y tutores los logros parciales del proyecto. La relación con nuestro tutor (el Ing. Javier Pereira) se mantuvo fluida a lo largo de todo el proceso, realizando reuniones periódicamente en las que el grupo exponía los últimos avances del proyecto y se fijaban objetivos a corto plazo, así como lineamientos globales.

En una primera instancia (antes de comenzar con las etapas de diseño), el grupo debió embarcarse en una investigación acerca de los distintos temas que formarían parte del proyecto, esto abarcó principalmente temas referentes a la tecnología Bluetooth, sus fundamento teóricos, protocolos y su funcionamiento practico, cuestiones sobre las cuales ninguno de los integrantes del grupo tenía conocimiento alguno. También se debieron abordar temas adicionales como son el desarrollo de software en Java (tanto para computadoras como para celulares), las comunicaciones mediante TCP/IP, bases de datos y SQL, la gestión de proyectos y hasta el uso de herramientas para dar formato a documentos (Latex). Todo esto constituyó sin duda un aporte enorme en el desarrollo académico y profesional del grupo a lo largo de este año y medio que duró el transcurso del proyecto.

Con respecto al desempeño del grupo podemos afirmar que a lo largo del tiempo los avances se mantuvieron incesantes, en parte gracias a la constancia, compromiso y dedicación con que fue encarado el proyecto. Se logró una correcta división de tareas en la cual la dedicación quedó repartida de manera equitativa y que, gracias a la fluida comunicación del grupo, desembocó en un muy buen ritmo de trabajo. Pese a lo anterior, hay que destacar que el proyecto atravesó por varias etapas de estancamiento en las cuales aparecieron inconvenientes que no resultaban sencillos de sortear. Afortunadamente, en esos casos, el grupo supo continuar los avances paralelamente por otros carriles, manteniendo la atención en el problema planteado hasta que finalmente éste era resuelto. Muchas veces se aprovechaban este tipo de situaciones para dedicarse al estudio teórico de temas que requerían más profundización.

Finalmente nos resta comentar que si bien hemos avanzado mucho a lo largo de todo este tiempo, queda planteado sinnúmero de posibles extensiones para el proyecto, así como distintas aplicaciones que utilicen parcialmente algunos de los resultados obtenidos. Es por esto que dejamos documentado lo más fielmente posible nuestro proyecto, intentando describir todo el camino recorrido por el grupo, de la manera más detallada posible, esperando que en un futuro sirva como guía para posibles investigaciones o proyectos de otros estudiantes, a continuación listamos algunas sugerencias para posibles extensiones del proyecto:

- Realizar mejoras en la interfaz grafica e incluir más facilidades para el usuario en la comunicación.
- Optimizar la eficiencia del protocolo de comunicación.
- Permitir al usuario enviar archivos guardados en memoria así como guardar los archivos recibidos.
- Mejorar el procedimiento de envío de mensajes Store and Forward, por ejemplo utilizando el numero MSISDN del terminal en lugar de su dirección MAC.
- Mejorar la compatibilidad de los dispositivos multimedia (audio y cámara).
- Enviar grabaciones de video capturadas desde la cámara.
- Manejar la posibilidad de establecer comunicaciones de voz en tiempo real.
- Permitir al usuario enviar mensajes que ingresen en la red celular publica como SMS o MMS y viceversa.

Anexo 1: Bluetooth

A1.1 Introducción

Bluetooth es una especificación que define redes de área personal inalámbricas (wireless personal area network, WPAN). Está basada en un enlace de radio de bajo coste y corto alcance, proporcionando conexiones instantáneas (ad hoc) para entornos de comunicaciones tanto móviles como estáticos. La especificación es desarrollada por Bluetooth SIG y, a partir de su versión 1.1, sus niveles más bajos (en concreto, el nivel físico y el control de acceso al medio) se formalizan también en el estándar IEEE 802.15.1. [3]

El principal objetivo de esta tecnología, es la posibilidad de reemplazar los muchos cables propietarios que conectan unos dispositivos con otros por medio de un enlace radio universal de corto alcance. Por ejemplo, la tecnología de radio Bluetooth implementada en el teléfono celular y en el ordenador portátil reemplazaría el molesto cable utilizado hoy en día para conectar ambos aparatos. Las impresoras, las agendas electrónicas, los PDA, los faxes, los teclados, los joysticks y prácticamente cualquier otro dispositivo digital son susceptibles de formar parte de un sistema Bluetooth.

Pero más allá de reemplazar los incómodos cables, la tecnología Bluetooth ofrece un puente a las redes de datos existentes, una interfaz con el exterior y un mecanismo para formar en el momento, pequeños grupos de dispositivos conectados entre sí de forma privada fuera de cualquier estructura fija de red.

El nombre procede del rey danés y noruego Harald Blåtand cuya traducción al inglés sería Harold Bluetooth, conocido por buen comunicador y por unificar las tribus noruegas, suecas y danesas. [2]

De la misma manera, Bluetooth intenta unir diferentes tecnologías como las de las computadoras, los teléfonos móviles y el resto de periféricos. [2]

A1.2 Como funciona

Bluetooth opera en la banda de 2.4 GHz libre para ISM (banda de frecuencia industrial, científica y médica, 2.402 - 2.480 GHz). El sistema emplea un transmisor de salto de frecuencia para contrarrestar las interferencias y la pérdida de intensidad, y cuenta con gran número de portadoras de espectro ensanchado por salto de frecuencia

(FHSS). Para minimizar la complejidad del transmisor, se utiliza una modulación de frecuencia binaria. La tasa de transferencia de símbolos es de 1 MS/s (megasímbolos por segundo), que admite una velocidad de transmisión de 1 Megabit por segundo (Mbps) en el modo de transferencia básica y una velocidad de transmisión aérea total de 2 a 3 Mbps en el modo de transferencia de datos mejorada. [1]

Salto de frecuencia y salto adaptable de frecuencia (AFH)

Como se mencionó, Bluetooth utiliza la técnica FHSS, que consiste en dividir la banda de frecuencia de 2.402 - 2.480 GHz en 79 canales (denominados saltos) de 1 MHz de ancho cada uno para luego, transmitir la señal utilizando una secuencia de canales que sea conocida tanto para el dispositivo emisor como para el receptor. Esta secuencia puede adaptarse para excluir la sección de frecuencias utilizadas por los dispositivos que están causando interferencias. La técnica de salto adaptable mejora la coexistencia de la tecnología Bluetooth con los sistemas ISM estáticos (es decir, sin salto) cuando éstos se encuentran localizados. [1]

Ranuras de tiempo y paquetes – Transmisión bidireccional

El canal físico se subdivide en unidades de tiempo denominadas ranuras. Los datos intercambiados entre los dispositivos Bluetooth se transmiten en forma de paquetes que se emplazan en estas ranuras. Cuando la situación lo permite, se pueden asignar varias ranuras consecutivas a un único paquete. El salto de frecuencia se produce durante la transmisión o recepción de los paquetes. La tecnología Bluetooth consigue la transmisión bidireccional mediante la técnica de acceso múltiple o dúplex por división de tiempo (TDD), dividiendo el canal en intervalos de 625 us. [1]

A1.3 Topología de red

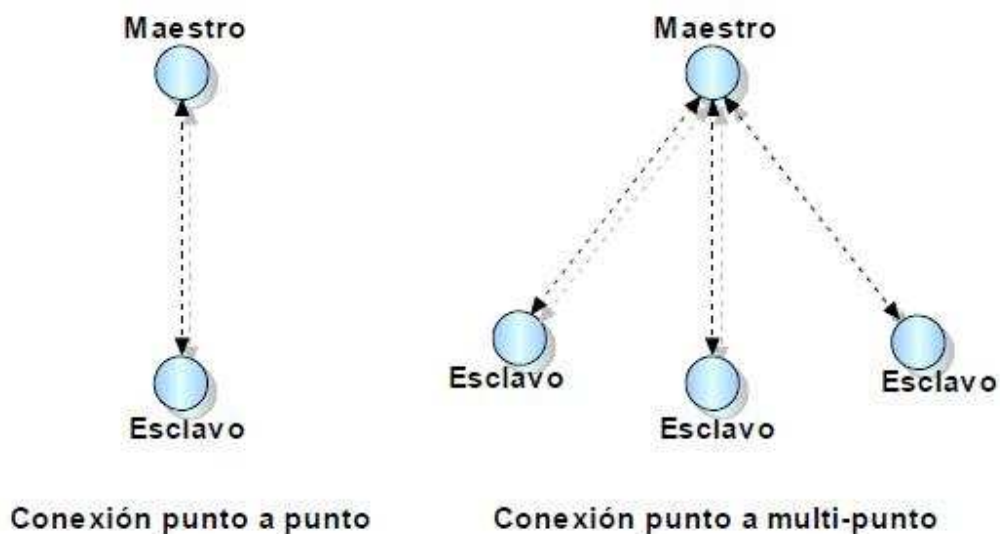
El estándar Bluetooth se basa en el modo de operación maestro/esclavo. Presenta dos tipos de configuraciones posibles, las cuales se pueden expandir a un número considerable de elementos para conformar así las redes y sub-redes. La estructura que maneja esta tecnología está compuesta, en su forma más básica, por lo que se denomina una Piconet y en una estructura un poco más compleja a la que se denomina una Scatternet.

A1.3.1 Piconet

Dos o más dispositivos Bluetooth que comparten un mismo canal forman una piconet. Para regular el tráfico en el canal, uno de los dispositivos participantes se convertirá en maestro, pero por definición, la unidad que establece la piconet asume éste papel y todos los demás serán esclavos. Los participantes podrían intercambiar los papeles si una unidad esclava quisiera asumir el papel de maestra. Sin embargo sólo puede haber un maestro en la piconet al mismo tiempo. Los dispositivos en una piconet poseen una dirección lógica de 3 bits, para un máximo de 8 dispositivos. Algunos esclavos pueden ser removidos de la piconet pero aún estar utilizando el mismo patrón de salto de frecuencia y escuchar por transmisiones del maestro. A un dispositivo en estas condiciones se le denomina esclavo estacionado (dispositivo en modo de espera), de los cuales una piconet puede tener hasta 255.

Cada unidad de la piconet utiliza su identidad maestra y reloj nativo para seguir en el canal de salto. Cuando se establece la conexión, se añade un ajuste de reloj a la propia frecuencia de reloj nativa de la unidad esclava para poder sincronizarse con el reloj nativo del maestro. El reloj nativo mantiene siempre constante su frecuencia, sin embargo los ajustes producidos por las unidades esclavas para sincronizarse con el maestro, sólo son válidos mientras dura la conexión.

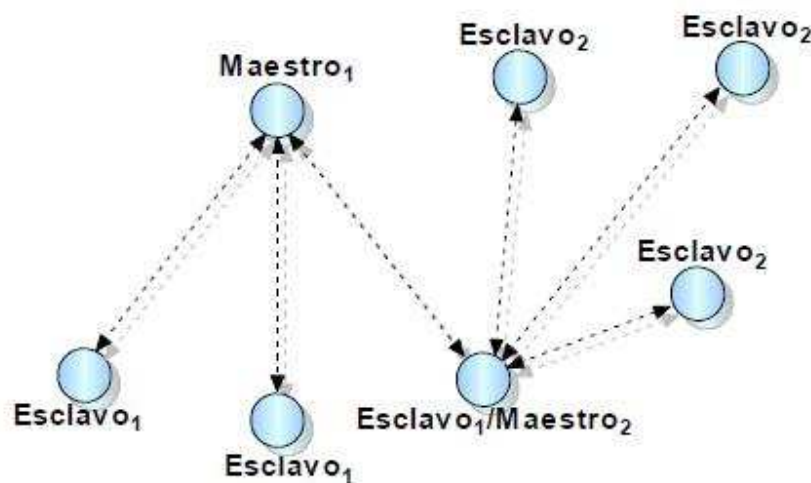
Bluetooth se ha diseñado para operar en un ambiente multi-usuario. Los dispositivos pueden habilitarse para comunicarse entre sí e intercambiar datos de una forma transparente al usuario. Hasta ocho usuarios o dispositivos pueden formar una piconet y hasta 10 Piconets pueden coexistir en una misma área de cobertura. Dado que cada enlace es codificado y protegido contra interferencia y pérdida de enlace, Bluetooth puede considerarse una red inalámbrica de corto alcance y muy segura.



A1.3.2 Scatternet

Bluetooth permite que dos o más piconets puedan conectarse entre sí para formar una red más amplia, denominada "scatternet", al utilizar ciertos dispositivos que actúan como puente entre las dos piconets.

Cada piconet en una scatternet tiene su propio patrón de salto de frecuencia e igual que en una piconet, los dispositivos pueden entrar y salir de ellas de forma dinámica (redes adhoc). [4]

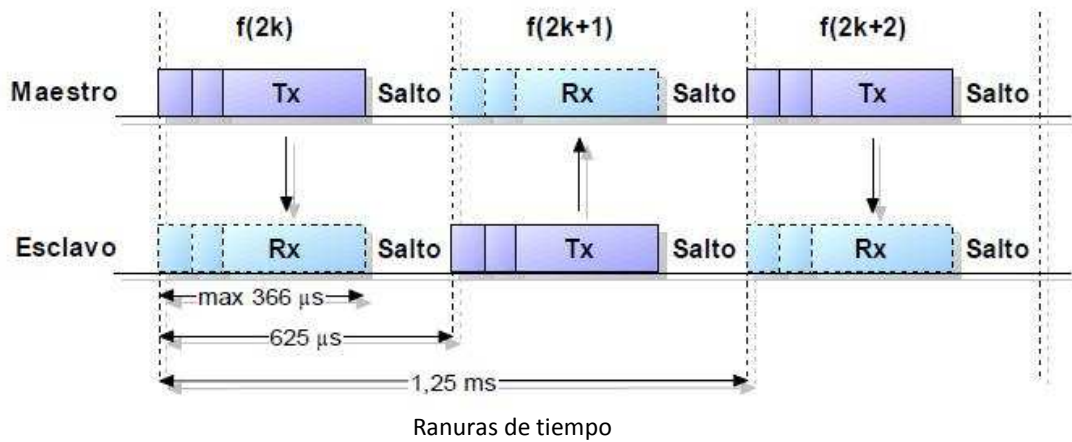


A1.4 Esquema de comunicación

La tecnología emplea un esquema de multiplexación por división de tiempo (TDM) el cual permite dividir un único canal en ranuras de tiempo. Lo cual significa que los dispositivos no pueden transmitir y recibir datos simultáneamente, similar a una comunicación half-duplex en un medio alambrado. [4]

Cada ranura de tiempo es numerada con enteros consecutivos y tiene una longitud de 625 μ s, así, el maestro transmitirá en ranuras de tiempo par y el esclavo en ranuras de tiempo impar. De esta forma, si el maestro transmite, el esclavo recibe, y viceversa. [4]

La ranura está dividida en dos partes: la primera consta de una estructura de datos denominada paquete y la segunda es el lapso de tiempo utilizado por los radios para sintonizarse en la siguiente frecuencia de salto. La siguiente figura muestra este comportamiento. [4]

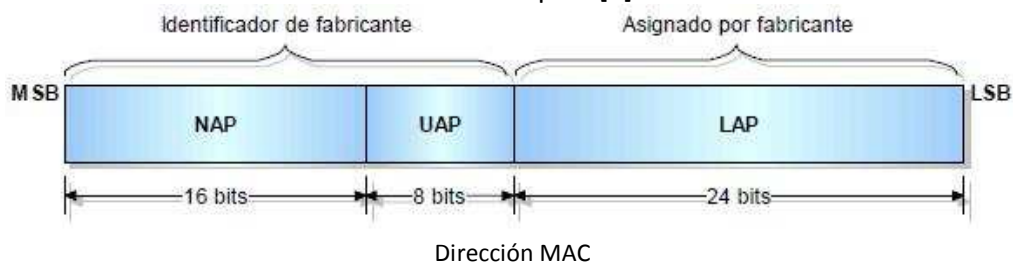


Existen paquetes multi-ranura los cuales pueden ocupar 1, 3 o hasta 5 ranuras, extendiendo la cantidad de información que pueda tener un solo paquete. [4]

A1.5 Direccionamiento Bluetooth

Las direcciones Bluetooth son números expresados en forma hexadecimal, los cuales pueden ser propios de cada dispositivo o asignados durante la operación de la piconet. Dentro de la especificación central se definen las siguientes direcciones:

- **Dirección de dispositivo Bluetooth BD_ADDR:** es un identificador IEEE 802 único de 48 bits gravado electrónicamente para cada dispositivo Bluetooth. Es la dirección utilizada para controlar el acceso al medio, y por lo tanto se usa llamarla dirección MAC. Está dividida en tres campos: [4]



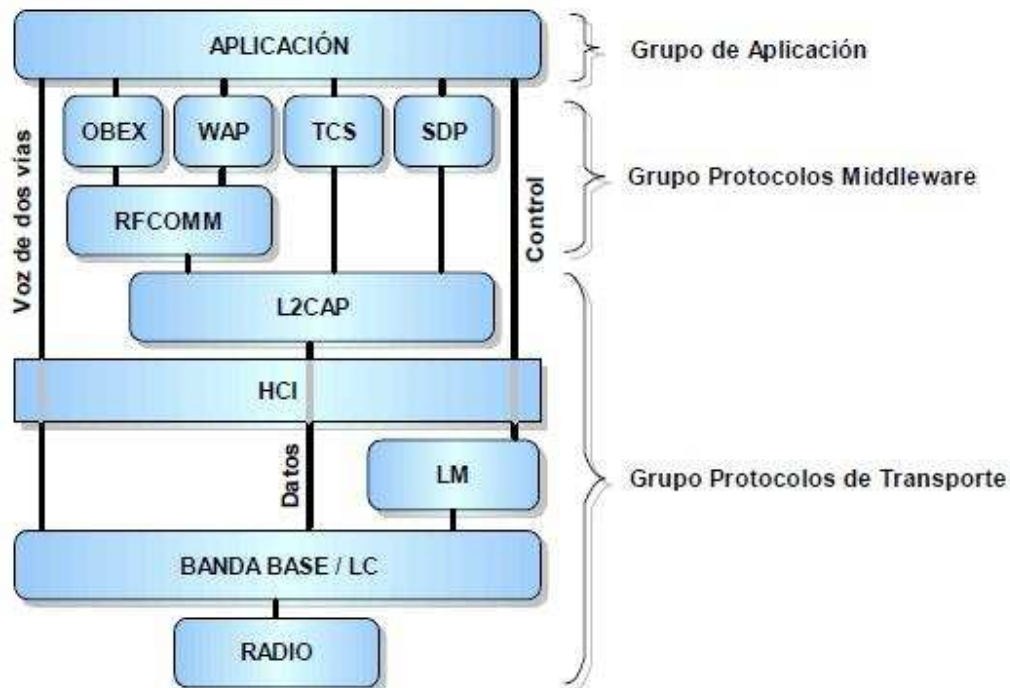
El NAP es la parte no significativa de la dirección y es utilizada para procedimientos de seguridad. Las partes alta (UAP) y baja (LAP) de la dirección son utilizadas para corrección de error y patrones de salto de frecuencia. La MAC y el reloj conforman

los elementos fundamentales utilizados para determinar el patrón de salto de frecuencia, tanto por el maestro como el esclavo. [4]

- **Dirección de miembro activo AM_ADDR:** es una dirección de 3 bits asignada por el maestro para identificar cada uno de los esclavos activos. Las direcciones desde el b'001' al b'111' se asignan a cada uno de los siete posibles esclavos de la piconet. La dirección b'000' es utilizada por el maestro para transmitir paquetes a todos los esclavos simultáneamente (broadcast). [4]
- **Dirección de miembro estacionado PM_ADDR:** es una dirección de 8 bits asignada por el maestro para reintegrar esclavos estacionados a la piconet. Cuando un miembro de la piconet se vuelve esclavo estacionado, renuncia a su AM_ADDR y el maestro le asigna una PM_ADDR. Las direcciones desde 0x01 hasta 0xFF se asignan a cada uno de los 255 posibles esclavos estacionados. Cuando PM_ADDR = 0x00, el esclavo solamente puede ser reintegrado a la piconet mediante su BD_ADDR. [4]
- **Dirección de solicitud de acceso AR_ADDR:** es otra dirección de 8 bits de longitud, entregada por el maestro al estacionar un esclavo y determina una ventana de tiempo en la cual el esclavo estacionado puede solicitarle al maestro ser un miembro activo de nuevo. [4]

A1.6 Pila de Protocolo Bluetooth

La pila Bluetooth es el centro de la tecnología ya que describe el lenguaje común que deben utilizar los dispositivos para comunicarse entre sí. Está constituida de varias capas entre las cuales, a excepción de las capas más bajas, existen protocolos que permiten su interacción. [4]



El grupo de transporte está integrado por un conjunto de protocolos que permiten la búsqueda de dispositivos Bluetooth, el manejo y la configuración de enlaces lógicos y físicos que conforman una “tubería virtual” por la cual fluyen los datos desde y hacia las capas más altas. Como se puede notar en la pila Bluetooth, los protocolos de transporte son indispensables para cualquier aplicación y en su mayoría pueden ser implementados tanto en hardware como firmware. Las capas que conforman este grupo son: radio, banda base y controlador de enlace (LC), manejador de enlace (LM), interfaz controlador de host (HCI) y protocolo de adaptación y control de enlace lógico (L2CAP). [4]

Los protocolos de capa media (middleware), presentan interfaces estándar para el flujo de datos desde el grupo de transporte hacia la capa de aplicación. En términos simples, funcionan como protocolos intermediarios. Estos incluyen: RFCOMM, OBEX, WAP (al igual que TCP, IP y PPP), TCS y SDP. [4]

El grupo de aplicación se refiere a software y es suministrado por el fabricante del aparato o vendido por desarrolladores independientes. Tales programas se acompañan con la pila del protocolo Bluetooth para obtener beneficios del aparato. En caso de aplicaciones existentes, o legado, que no traen soporte Bluetooth, posiblemente requieran de software adicional de adaptación. [4]

Para una aplicación determinada, no es necesario utilizar todas las capas de la pila. La especificación de perfiles indica que capas en particular se deben implementar y de qué manera según la aplicación. Los programadores pueden iniciar con los perfiles que son bases para dichas aplicaciones, pero estas son virtualmente ilimitadas, así que la especificación sólo menciona algunas. [4]

A1.6.1 Comparación con el modelo OSI



Como se aprecia en la figura, la pila de protocolo Bluetooth no sigue exactamente el modelo OSI, aunque si se hace un seguimiento gradual de las capas se pueden encontrar transiciones en la implementación de cada una de ellas desde hardware, pasando por firmware, y finalmente llegando a software. [4]

La capa física define las características eléctricas del medio de transporte para los datos, por lo que el radio y parte de la banda base cubren esta capa. La capa de enlace de datos es responsable de la transmisión, marco y control de error sobre un particular enlace, por lo que traslapa la tarea del controlador de enlace y parte final de la banda base, incluyendo chequeo y corrección de error. [4]

Una de las razones por la cual la pila Bluetooth no es igual al modelo OSI es la interoperabilidad. Dado que las capas altas del modelo OSI dan libertad a los fabricantes de implementarlas como desearán, dispositivos de diferentes fabricantes posiblemente no se comunicaban adecuadamente. Anticipando el problema, se tuvo en cuenta esta

situación, así que la especificación junto con otros documentos relacionados, fue construida de tal forma que se requería la interoperabilidad no importando quien diseñaba y construía el dispositivo. [4]

A1.6.2 Capa de Radio

La capa de Radio es la encargada de la conexión física (2,4 GHz, 79 canales de 1MHz, cambia de canal hasta 1.600 veces por segundo (multiplexado), alcance desde 10cm a 10m aunque puede extenderse hasta 100 metros incrementando el consumo). [3]

A1.6.3 Capa Banda Base y Controlador de Enlace (LC)

La banda base es la responsable de la codificación y decodificación del canal al igual que el manejo y control de tiempos a bajo nivel del enlace físico. El controlador de enlace se encarga de indicarle a la banda base que operaciones a nivel de enlace debe realizar para los diferentes paquetes de datos. En unión trabajan para:

- Ensamblar paquetes desde capas altas de protocolo y enviarlos al radio. [4]
- Recibir bits del radio y ensamblarlos en paquetes para las capas altas. [4]
- Generar el patrón de salto de frecuencia. [4]
- Controlar el error, asegurando la integridad de los datos. [4]
- Realizar operaciones básicas de seguridad, entre otras. [4]

Paquetes de Banda Base

El concepto de paquete es muy utilizado en todas las capas de la pila de protocolo Bluetooth y de forma general es llamado unidad de dato de protocolo (PDU) por lo que los paquetes de banda base se denominan BB_PDUs. Estos PDUs se caracterizan por tener un conjunto de campos bien definidos como se muestra en la figura:



Paquete de Banda Base

Tipos de Paquetes

Los diferentes tipos de paquetes son relacionados a dos enlaces físicos conocidos como ACL y SCO. Los enlaces ACL ofrecen alta integridad de la información y latencia, por lo que son básicamente utilizados para transferencia de datos. A diferencia de los enlaces SCO que son empleados para transmitir información de audio, donde no es tan importante la integridad de la información pero si una baja latencia. Debido a su baja latencia los paquetes SCO no son retransmitidos. [4]

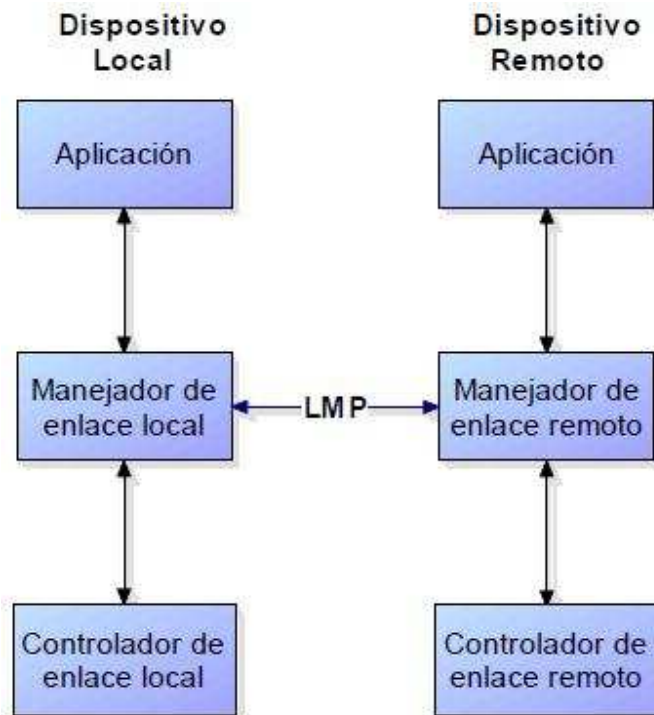
Para establecer un enlace SCO, inicialmente es necesario establecer un enlace ACL el cual permite llevar la información de control. [4]

Los tipos de paquetes se encuentran organizados en tres grupos: paquetes de control de enlace (ó comunes), ACL y SCO. Los de control de enlace permiten hacer reconocimientos de paquetes, corrección de error, procedimientos de pregunta y búsqueda y sincronización de dispositivos. [4]

A1.6.4 El Manejador de Enlace (LM)

La capa LM es tal vez la primera dentro de la pila de protocolo Bluetooth que comienza a implementarse como software. Por lo general, los fabricantes construyen la capa LM como firmware en una memoria independiente o en el mismo circuito integrado de banda base. [4]

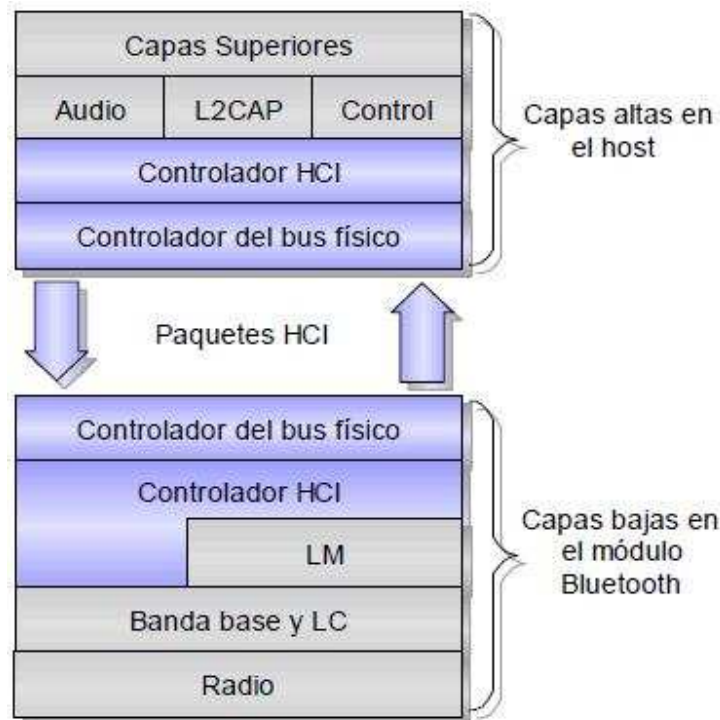
El manejador de enlace permite convertir comandos provenientes de la capa HCI en acciones a nivel de banda base, desempeñando las siguientes tareas: integrar o remover esclavos de la piconet, configurar y establecer enlaces, habilitar modos de ahorro de energía y controlar los modos de prueba. Adicionalmente un LM local se comunica con otro LM remoto a través del protocolo manejador de enlace (LMP), como muestra la siguiente figura. [4]



A1.6.5 Interfaz Controlador de Host (HCI)

La capa HCI presenta una interfaz estándar para sistemas, en donde las capas más bajas se encuentran separadas de las altas. En caso que toda la pila Bluetooth pueda ser contenida en un único dispositivo, esta capa no es necesaria y puede ser omitida (sistemas embebidos). [4]

Por lo general, los fabricantes implementan el radio, la banda base y el manejador de enlace en un único dispositivo denominado módulo Bluetooth. Así, las capas altas que requieren más capacidad de memoria y poder de procesamiento, pueden ser implementadas en un PC (host). [4]



Interacción módulo Bluetooth - host

La comunicación entre el módulo Bluetooth y el host se realiza mediante paquetes HCI, de los cuales existen 3 tipos: comandos, eventos y datos. Cada uno de estos paquetes fluye a través de una conexión física llamada capa de transporte HCI y puede ser implementada como UART, RS232, USB y PCMCIA, en caso que el dispositivo host sea un computador portátil. [4]

A1.6.6 Protocolo de Adaptación y Control de Enlace Lógico (L2CAP)

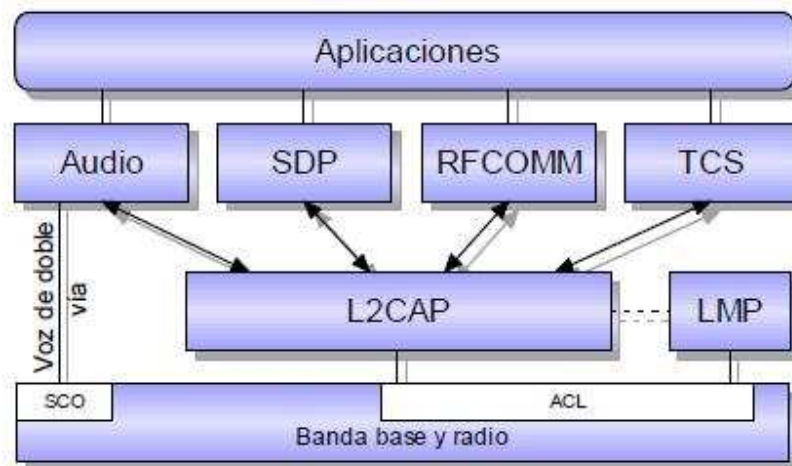
Esta capa normalmente es referida como L2CAP y cumple un papel importante para el intercambio de paquetes de datos, entre las capas bajas y altas incluyendo la aplicación. La capa L2CAP no soporta la transferencia de información de audio de dos vías en tiempo real; como fue mencionado antes, esta información viaja hacia y desde la banda base directamente. [4]

L2CAP asume que el enlace de banda base es confiable y no tiene que preocuparse por retransmitir paquetes en caso de error. Sin embargo, no garantiza confiabilidad para paquetes transmitidos a todos los esclavos (broadcast), debido a que los esclavos no tienen forma de confirmarle al maestro que la transmisión de estos fue exitosa. [4]

Las funciones del L2CAP pueden ser repartidas en 4 categorías:

- **Multiplexación de protocolos:** permite llevar paquetes provenientes de las capas bajas a alguno de los varios protocolos de las capas altas y a su vez puede tomar paquetes de las capas altas y llevarlos a las capas inferiores.

La siguiente figura muestra como la capa L2CAP distribuye paquetes. [4]



L2CAP como multiplexor de protocolos

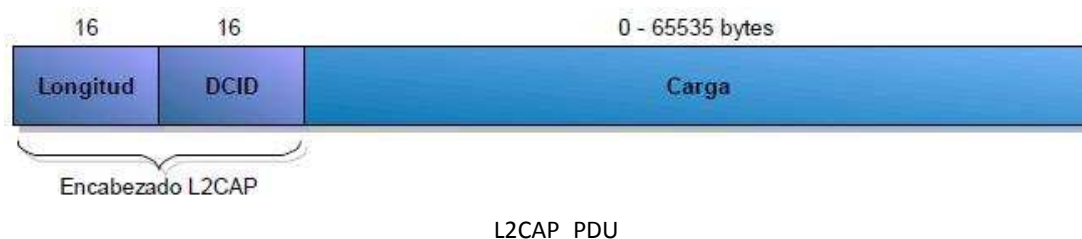
- **Segmentación y reensamble:** Por lo general, los paquetes provenientes de capas superiores son más grandes que los paquetes de banda base. Así, la capa L2CAP deberá dividir un paquete grande en paquetes más pequeños de tal forma que la banda base pueda transferirlos al radio. El proceso contrario al anterior es denominado reensamble y permite armar de nuevo un paquete que viene por partes desde la banda base. Este proceso es transparente, ya que las capas superiores siempre ven un único paquete. [4]
- **Calidad de servicio (QoS):** Le permite a las aplicaciones establecer requerimientos para el ancho de banda, máximo nivel de latencia, cuán rápido llegan los paquetes, entre otras. Cuando estos parámetros no son configurados, un servicio de mejor esfuerzo es asumido bajo las circunstancias del enlace actual. [4]
- **Manejo de grupo:** Los esclavos de la piconet pueden pertenecer a un grupo de protocolos aparte de la misma piconet, en los cuales se comparte información. [4]

Canales L2CAP

El concepto de canal permite establecer tuberías de datos entre un dispositivo maestro y uno esclavo. Cada extremo en una tubería se conoce como identificador de canal (CID), en particular, si el paquete L2CAP se origina en un dispositivo local, este extremo de la tubería se llama identificador de canal fuente (SCID). De lo contrario, si el paquete L2CAP es enviado a través del canal a un dispositivo remoto, este extremo del canal se le denomina identificador de canal destino (DCID). Existen tres tipos de canales:

- **De conexión orientada (CO):** Permite intercambiar datos entre una aplicación del maestro y otra localizada en un único esclavo. El rango de valores, tanto para el SCID como para el DCID es 0x0040 a 0xFFFF. [4]
- **Sin conexión (CL):** Es normalmente utilizado para enviar datos desde un maestro a un número de esclavos en forma simultánea. El SCID va desde 0x0040 hasta 0xFFFF, a diferencia del DCID que siempre es 0x0002. [4]
- **De señalamiento:** Este canal existe por defecto entre el maestro y cada uno de sus esclavos y permite establecer otros canales para el intercambio de datos de usuario. Tanto el SCID como el DCID es 0x0001. [4]

Los paquetes L2CAP viajan a través de los canales y tienen una estructura relativamente simple si se comparan con los paquetes de banda base y manejador de enlace. Constan de un encabezado que contiene un número de 16 bits para indicar la longitud de la carga (máximo 64 kB) y el identificador de canal CID de 16 bits igualmente. [4]



A1.6.7 Protocolos Intermediarios

Estos protocolos presentan interfaces estándar a la capa de aplicación. Su uso depende del tipo de aplicación y por lo tanto no todos son necesarios a la hora de implementarla. Algunos de estos protocolos son adoptados por la tecnología Bluetooth, tales como protocolos de Internet (TCP, IP y PPP) y protocolos de la asociación de datos infrarrojos, o IrDA por sus siglas en inglés (OBEX e IrMC). Los protocolos restantes son

específicos de la tecnología Bluetooth y son: RFCOMM, TCS y SDP. A continuación se describe brevemente dos de los más importantes:

- **RFCOMM:** Partiendo de la idea del reemplazo de cables, se consideró la implementación de un puerto serial emulado para muchas de las aplicaciones que requieran de este tipo de interfaz física. RFCOMM es en realidad una adaptación del estándar ETSI TS 07.101 para necesidades específicas de la pila de protocolo Bluetooth. Adicionalmente, esta capa permite emular los pines de una interfaz RS-232 como TX, RX, CTS, RTS, DSR etcétera y tener hasta 60 enlaces lógicos seriales multiplexados entre dos dispositivos. [4]
- **Protocolo de descubrimiento de servicio (SDP):** Es un protocolo que permite averiguar por servicios disponibles en un ambiente Bluetooth. El SDP se basa en una comunicación persona a persona (peer-to-peer) donde existe un dispositivo cliente que puede solicitar un servicio o averiguar que servicios ofrece un segundo dispositivo servidor (esquema cliente-servidor). Los servicios que puede ofrecer un dispositivo servidor se encuentran organizados por clase y atributo, en una base de datos denominada registro de servicios. [4]

A1.6.8 Perfiles

También llamados modelos de uso, los perfiles definen aplicaciones de la tecnología Bluetooth para el usuario final. Debido a que los perfiles establecen una base común para ciertas aplicaciones, la tecnología permite que las aplicaciones utilicen hardware Bluetooth sin importar el fabricante. Los perfiles están originalmente enfocados hacia las siguientes categorías: teléfono tres en uno, el último headset, puente Internet, punto de acceso de datos, empuje de objeto, transferencia de archivo, sincronización automática, entre otras aplicaciones que surgirán con el avance de la tecnología. [4]

A continuación se mencionan algunos de los perfiles más utilizados.

Perfiles genéricos

Este grupo de perfiles es la base de muchos otros perfiles y esta compuesto por:

- **Perfil de acceso genérico (GAP):** Es la base común de todos los demás perfiles ya que permite la interoperabilidad de los protocolos del grupo de transporte e incluye operaciones como procedimientos de pregunta y establecimiento de

conexión. El GAP establece políticas para establecer comunicación entre dos dispositivos y define modos de descubrimiento, conexión y apareamiento. El modo de descubrimiento, por ejemplo, dice como un dispositivo Bluetooth descubre a otros, es decir, decide si para procesos de pregunta utiliza el GIAC, a lo que se denomina modo de descubrimiento general o utiliza el LIAC a lo que se llama modo de descubrimiento limitado. [4]

- **Perfil de aplicación de descubrimiento de servicio (SDAP):** Este describe las características funcionales de una aplicación de descubrimiento de servicio. Aun más, e igualmente importante, el SDAP describe la forma que la capa SDP deberá usar el grupo de protocolos de transporte Bluetooth para llevar a cabo las transacciones de descubrimiento de servicio. Este aspecto del SDAP también forma base para ejecutar el descubrimiento de servicio dentro de otros perfiles. [4]

Perfiles de puerto serial e intercambio de objeto

El perfil de puerto serial utiliza directamente la capa RFCOMM para realizar el reemplazo de cable en algunos escenarios de uso. Los perfiles de intercambio de objeto se utilizan en dispositivos de telefonía y computación apoyándose directamente en los protocolos adoptados de IrDA. Los perfiles que incluyen este grupo son:

- **Perfil de puerto serial (SPP):** Define los requerimientos para configurar las conexiones de un cable serial emulado. El SPP se apoya en el protocolo RFCOMM que es transparente a la capa de aplicación, es decir, programas que utilicen un puerto serial deberán pensar que existe un puerto serial alambrado. [4]
- **Perfil de empuje de objeto (OPP):** Se utiliza para empujar (enviar) un objeto a otro dispositivo. Contrario a su nombre, este perfil también permite halar (recuperar) de otro dispositivo que soporte esta operación. El OPP se utiliza básicamente para el intercambio de tarjetas de presentación electrónicas (vCard), calendarios (vCalendar), mensajes (vMessage) y notas (vNote) si los dispositivos soportan este tipo de formatos. [4]
- **Perfil de transferencia de archivo (FP):** Este perfil permite el intercambio de archivos entre dos dispositivos. Si alguno de los archivos esta autorizado, este podrá buscar archivos en el sistema, abrir y ver tanto archivos como directorios, transferirlos de uno a otro y realizar operaciones de operación tales como crear, mover o borrar carpetas en otro dispositivo. [4]
- **Perfil de sincronización (SP):** Define la capacidad de dos dispositivos conectarse y actualizar sus archivos como libretas telefónicas, correos electrónicos, calendarios

entre otros, a la última versión existente. El proceso de sincronización puede realizarse manual o automáticamente. [4]

Otros perfiles importantes son los de telefonía (Perfil de telefonía inalámbrica, Perfil de intercomunicador y Perfil de auricular) y los de red (Perfil de redes de marcado, Perfil de acceso a redes de área local y Perfil de fax). [4]

A1.7 Modos y Procedimientos de operación

Normalmente, un dispositivo Bluetooth se conecta a otro dispositivo Bluetooth compatible en una piconet para intercambiar datos. En la tecnología inalámbrica Bluetooth, los equipos se comunican directamente (comunicación ad-hoc) y existen, además, procedimientos de operación que facilitan la creación de piconets para que se establezcan comunicaciones adicionales. Los procedimientos y modos se aplican en capas diferentes de la arquitectura, por lo que un dispositivo podrá participar simultáneamente en varios de estos procedimientos y modos. [1]

A1.7.1 Procedimiento de pregunta y búsqueda (inquiry)

Estos procedimientos son fundamentales para constituir una piconet, debido a que el potencial maestro requiere conocer que dispositivos existen en la proximidad y con cual o cuales en particular se desea conectar. [4]



Diagrama de estados para establecer una piconet

El mecanismo de pregunta es utilizado cuando se requiere conocer que dispositivos están disponibles para la conexión. En caso contrario, se reduce el tiempo de conexión si se conocen los dispositivos utilizando directamente el mecanismo de búsqueda para conectarse con un dispositivo en particular. Una vez lograda la conexión, los esclavos pueden participar activamente de la piconet o decidir entrar en alguno de los modos de ahorro de energía. [4]

Tanto el dispositivo receptor como el emisor de la solicitud de búsqueda, podrán estar conectados a otro dispositivo con tecnología Bluetooth dentro de una piconet. El tiempo de búsqueda o de ocupación del canal físico correspondiente debe ser proporcional a la demanda de calidad de servicio (QoS) para las comunicaciones lógicas existentes. [1]

A1.7.2 Procedimiento de paginación o conexión (paging)

Las conexiones se establecen por un procedimiento asimétrico en el que un dispositivo Bluetooth realiza la paginación o conexión, mientras que otro dispositivo Bluetooth susceptible de conectarse realiza la búsqueda de la señal. Se trata de un procedimiento dirigido, es decir, sólo habrá respuesta del dispositivo Bluetooth hacia el que va destinado. [1]

El dispositivo susceptible de conectarse utiliza un canal físico especial para recibir los paquetes con la solicitud de conexión que envía el dispositivo de paginación. Este canal físico tiene atributos específicos del dispositivo que se va a conectar, de ahí que sólo un dispositivo de paginación con datos del anterior pueda comunicarse en este canal. [1]

Los dispositivos involucrados en el procedimiento pueden estar conectados, a su vez, con otros aparatos con tecnología Bluetooth en una piconet distinta. El tiempo de paginación o de ocupación del canal físico correspondiente debe ser proporcional a la demanda de calidad de servicio (QoS) para las comunicaciones lógicas existentes. [1]

A1.7.3 Modo conectado

Cuando el procedimiento de conexión se realiza correctamente, los dispositivos quedan físicamente conectados entre sí en una piconet. Una vez que la conexión está establecida, es posible crear y activar otros enlaces lógicos, así como cambiar los modos de los enlaces lógicos y físicos mientras los dispositivos siguen conectados al canal físico de la piconet. Además, se podrán realizar procedimientos de búsqueda, paginación o exploración, o bien conectarse a otras piconets sin necesidad de abandonar la primera. [1]

Mientras un dispositivo esclavo está conectado a una piconet, se produce una comunicación lógica ACL predeterminada entre el dispositivo maestro y esclavo. Existen dos métodos para eliminar esta comunicación. El primero consiste en desconectar el dispositivo del canal físico de la piconet, lo que ocasionará que desaparezca la jerarquía de los canales L2CAP, los enlaces lógicos y las comunicaciones lógicas que vinculan los dispositivos. El segundo, en crear un enlace físico para que el dispositivo esclavo quede en modo park, es decir, a la espera; tras lo que se dará por concluida la comunicación lógica ACL predeterminada. Esto sólo es posible si se han dado por terminadas todas las comunicaciones lógicas, excepto la de difusión del dispositivo esclavo activo que no puede crearse ni eliminarse directamente. [1]

A1.7.4 Modos de ahorro de energía

Modo hold o de sostenimiento

El modo hold no suele ser habitual, aunque se aplica a las ranuras sin reservar del enlace físico. En este modo, el enlace físico sólo está activo cuando hay ranuras reservadas para el funcionamiento de los enlaces síncronos SCO y eSCO (enhanced SCO). Los enlaces asíncronos estarán inactivos. Los modos de retención funcionan una vez para cada llamada; terminada la llamada, se vuelve al modo anterior. [1]

Modo sniff o de reastro

En este modo el esclavo conservando su dirección de miembro activo, reduce su actividad en la piconet apagando su receptor durante un tiempo prefijado, para luego encenderse una cantidad de ranuras de tiempo y esperar por paquetes del maestro. Una vez finalizadas las transmisiones del maestro, el esclavo espera una cantidad de ranuras de tiempo adicional antes de volver a desactivar su receptor. Este proceso es periódico hasta que se decida regresar a la actividad normal de piconet (modo activo). [1]

Modo park o de espera

Un dispositivo esclavo puede permanecer conectado a una piconet aun cuando su enlace físico se encuentre en estado de espera. En este modo el esclavo renuncia a su dirección de miembro activo, y a cambio, el maestro le asigna una dirección de miembro estacionado y una dirección de solicitud de acceso. Un esclavo en este modo no participa activamente de la piconet, pero se mantiene en sincronismo con el maestro en caso de

retornar a esta. Tanto el esclavo como el maestro pueden iniciar procedimientos para regresar a la actividad normal de la piconet. [1]

A1.7.5 Procedimiento de intercambio de funciones

El procedimiento de intercambio de funciones permite intercambiar las funciones de dos dispositivos conectados a una piconet. Para ello, es necesario pasar del canal físico definido por el dispositivo maestro original al canal físico que definirá el nuevo maestro. En este proceso de salto al nuevo canal, la jerarquía de enlaces físicos y comunicaciones lógicas se elimina para, a continuación, volverse a establecer. Tras el intercambio de funciones, el canal físico original de la piconet podrá dejar de existir o seguir activo, si hay dispositivos esclavos que siguen conectados al canal del maestro original. [1]

Los valores de calidad de servicio (QoS) o los modos establecidos (como el de escucha reducida) en las comunicaciones lógicas originales se pierden tras el intercambio de funciones y tendrán que volver a negociarse una vez completada la operación. [1]

Anexo 2: TCP/IP

TCP/IP es un conjunto de protocolos. Por un lado, TCP (Transmission-Control-Protocol, en español Protocolo de Control de Transmisión) es un protocolo de comunicación orientado a conexión y fiable a nivel de transporte. Por otra parte, IP (de sus siglas en inglés Internet Protocol), es un protocolo no orientado a conexión usado tanto por el origen como por el destino para la comunicación de datos a través de una red de paquetes conmutados. [5]

En algunos aspectos, TCP/IP representa todas las reglas de comunicación para Internet y se basa en la noción de dirección IP, es decir, en la idea de brindar una dirección IP a cada equipo de la red para poder enrutar paquetes de datos. Debido a que el conjunto de protocolos TCP/IP originalmente se creó con fines militares, está diseñado para cumplir con una cierta cantidad de criterios, entre ellos:

- Dividir mensajes en paquetes
- Usar un sistema de direcciones
- Enrutar datos por la red
- Detectar errores en las transmisiones de datos

En la pila de protocolos TCP/IP, TCP es la capa intermedia entre el protocolo de Internet (IP) y la aplicación. Habitualmente, las aplicaciones necesitan que la comunicación sea fiable y, dado que la capa IP aporta un servicio de datagramas no fiable (sin confirmación), TCP añade las funciones necesarias para prestar un servicio que permita que la comunicación entre dos sistemas se efectúe libre de errores, sin pérdidas y con seguridad. [5]

En la siguiente tabla se muestra la composición de la pila de protocolos (a la izquierda) y algunos ejemplos de protocolos que la pueden componer (en la derecha).

Aplicación	http, ftp, DNS, ...
Transporte	TCP , UDP, ...
Red	IP
Enlace	Ethernet, WiFi, Token Ring, ...
Física	Cable UTP, Interfaz de radio, ...

[5]

Los servicios provistos por TCP corren en el anfitrión (host) de cualquiera de los extremos de una conexión, no en la red. Por lo tanto, TCP es un protocolo para manejar conexiones de extremo a extremo. Tales conexiones pueden existir a través de una serie de conexiones punto a punto, por lo que estas conexiones extremo-extremo son llamadas circuitos virtuales. Las características del TCP son:

- **Orientado a conexión:** dos computadoras establecen una conexión para intercambiar datos. Los sistemas de los extremos se sincronizan con el otro para manejar el flujo de paquetes y adaptarse a la congestión de la red. [5]
- **Operación Full-Duplex:** una conexión TCP es un par de circuitos virtuales, cada uno en una dirección. Sólo los dos sistemas finales sincronizados pueden usar la conexión. [5]
- **Error Ckecking:** una técnica de chacksum (suma de comprobación) es usada para verificar que los paquetes no estén corrompidos. [5]
- **Acknowledgements:** sobre recibo de uno o más paquetes, el receptor regresa un acknowledgement (reconocimiento) al transmisor indicando que recibió los paquetes. Si los paquetes no son notificados, el transmisor puede reenviar los paquetes o terminar la conexión si el transmisor cree que el receptor no está más en la conexión. [5]
- **Flow Control:** si el transmisor está desbordando el buffer del receptor por transmitir demasiado rápido, el receptor descarta paquetes. Los acknowledgement fallidos alertan al receptor para bajar la tasa de transferencia o dejar de transmitir. [5]
- **Servicio de recuperación de Paquetes:** el receptor puede pedir la retransmisión de un paquete. Si el paquete no es notificado como recibido (ACK), el transmisor envía de nuevo el paquete. [5]

Los servicios confiables de entrega de datos son críticos para aplicaciones tales como transferencias de archivos (FTP por ejemplo), servicios de bases de datos, proceso de

transacciones y otras aplicaciones de misión crítica en las cuales la entrega de cada paquete debe ser garantizada. [5]

El siguiente esquema muestra la estructura de la trama TCP:

+	Bits 0 - 3	4 - 7	8 - 15	16 - 31
0	Puerto Origen		Puerto Destino	
32	Número de Secuencia			
64	Número de Acuse de Recibo (ACK)			
96	longitud cabecera TCP	Reservado	Flags	Ventana
128	Suma de Verificación (Checksum)		Puntero Urgente	
160	Opciones + Relleno (opcional)			
224	Datos			

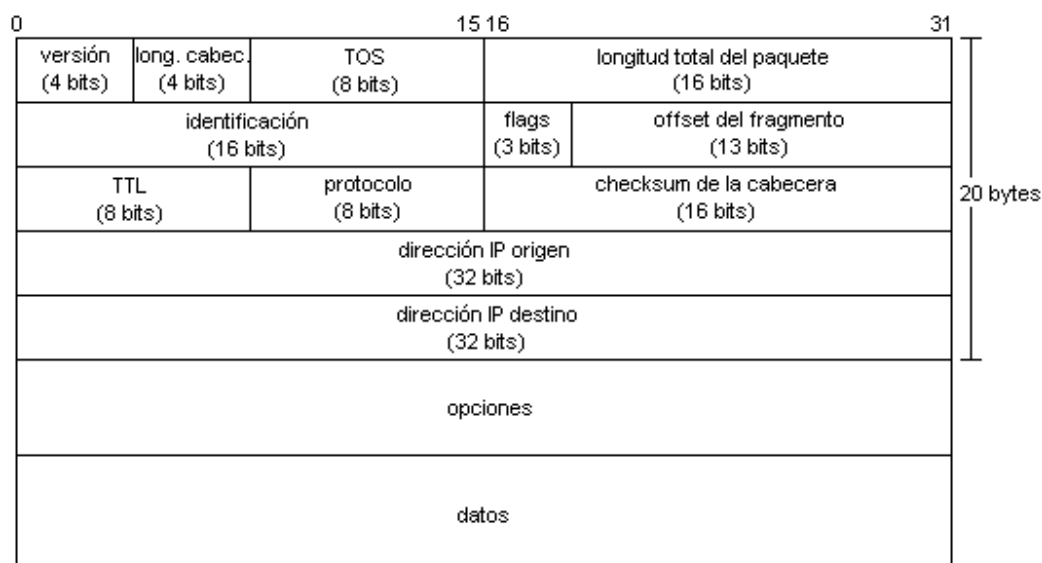
[5]

En lo que respecta a IP, los datos en una red basada en IP son enviados en bloques conocidos como paquetes o datagramas (en el protocolo IP estos términos se suelen usar indistintamente). En particular, en IP no se necesita ninguna configuración antes de que un equipo intente enviar paquetes a otro con el que no se había comunicado antes.

El Protocolo de Internet provee un servicio de datagramas no fiable (también llamado del mejor esfuerzo (best effort), lo hará lo mejor posible pero garantizando poco). IP no provee ningún mecanismo para determinar si un paquete alcanza o no su destino y únicamente proporciona seguridad (mediante checksums o sumas de comprobación) de sus cabeceras y no de los datos transmitidos. Por ejemplo, al no garantizar nada sobre la recepción del paquete, éste podría llegar dañado, en otro orden con respecto a otros paquetes, duplicado o simplemente no llegar. Si se necesita fiabilidad, ésta es proporcionada por los protocolos de la capa de transporte, como TCP. [6]

El actual y más popular protocolo de red es IPv4 (IP versión 4, utilizado en este proyecto). IPv6 (IP versión 6) es el sucesor propuesto de IPv4; poco a poco Internet está agotando las direcciones disponibles por lo que IPv6 utiliza direcciones de fuente y destino de 128 bits, muchas más direcciones que las que provee IPv4 con 32 bits. [6]

Un paquete IP se compone de la siguiente manera:



- **Versión (4 bits):** Define el número de versión de cabecera utilizada.
- **Tamaño de Cabecera (4 bits):** Longitud en palabras de 32bits .
- **Tipo de Servicio (8 bits):** Indica una serie de parámetros sobre la calidad de servicio.
- **Longitud Total (16 bits):** Tamaño total en octetos del datagrama (cabecera más datos). En caso de fragmentación, contendrá el tamaño de la parte, no del datagrama original.
- **Identificador (16 bits):** Identificador único del datagrama, utilizado en caso de que el mismo deba ser fragmentado para poder distinguir las partes de un datagrama de los de otro.
- **Indicadores (3 bits):** Utilizado para especificar valores relativos a la fragmentación de paquetes.
- **Posición del fragmento (13 bits):** Indica la posición en unidades de 64 bits que ocupa el paquete actual dentro del datagrama original.

- **Tiempo de vida (8 bits):** Número máximo de enrutadores que un paquete puede atravesar.
- **Protocolo (8 bits):** Indica el protocolo utilizado en el siguiente nivel para la parte de datos del datagrama.
- **Checksum de la cabecera (16 bits).**
- **Dirección IP de Origen (32 bits).**
- **Dirección IP de Destino (32 bits).**
- **Opciones (variable):** Aunque no es obligatoria, cualquier ruteador tiene que poseer la capacidad de interpretación del campo.

Anexo 3: SQL

SQL cuya sigla significa Structured Query Language (en español, Lenguaje de Consulta Estructurado) es un lenguaje declarativo de acceso a bases de datos que permite especificar diversos tipos de operaciones en éstas.

Es conocido como el lenguaje universal de comunicación con bases de datos. El hecho de ser universal no significa que sea idéntico su uso en cada sistema base de datos, ya que cada uno además de poseer varias funciones en común con otros, también posee funciones específicas de la misma.

Es un lenguaje declarativo considerado de “alto nivel”, que gracias a su fuerte base teórica y su orientación al manejo de conjuntos de registros, permite una alta productividad. [7]

SQL fue estandarizado por primera vez en 1986 por el ANSI, American National Standards Institute (en español, Instituto Nacional Estadounidense de Estándares), dando lugar a la primer versión estándar del lenguaje, llamado “SQL-86” o “SQL1”. Al año siguiente este estándar también fue adoptado por la ISO. [7]

Al no cubrir este estándar todas las necesidades de los desarrolladores, y al ser encontrarse por parte de estos algunas funciones suprimibles, en 1992 se lanza un nuevo estándar ampliado y revisado del SQL llamado “SQL-92” o “SQL2”. En realidad, entre estas dos revisiones se realizó una intermedia en 1989, pero sin contener cambios muy significativos la cual fue llamada “SQL-89”. [7]

Posteriormente, en 1999, se estableció el estándar “SQL:99” en el cual se agregaron expresiones regulares, consultas recursivas (para relaciones jerárquicas), triggers y algunas características orientadas a objetos. [7]

En 2003, se pasa al estándar “SQL:2003” el cual introduce algunas características de XML, cambios en las funciones, estandarización del objeto sequence y de las columnas auto numéricas. [7]

Finalmente en 2006, se define el último estándar hasta la actualidad, el cual define las maneras en las cuales el SQL se puede utilizar conjuntamente con XML. Además, establece maneras importar y guardar datos XML en una base de datos SQL, ofreciendo la posibilidad de manipularlos dentro de esta. [7]

Algunas de las bases de datos que utilizan SQL son:

- **MySQL:** es un gestor de bases de datos gratuito, muy potente, utilizado en muchos casos para servidores web. Consiste en un gestor relacional, multihilo y multiusuario.
- **PostgreSQL:** gestor de bases de datos relacional y orientado a objetos, de gran estabilidad y que admite SQL-92 y SQL-99.
- **Berkeley DB:** es un motor de base de datos de alto rendimiento y muy escalable que puede incluirse en cualquier aplicación.
- **Oracle:** es quizás el gestor de base de datos mas prestigioso. Se destaca entre sus potencialidades el soporte de transacciones, la estabilidad, escalabilidad y soporte multiplataforma.
- **Firebird:** es una base de datos relacional que ofrece muchas características del estándar SQL-92. Es de código abierto y en un principio fue desarrollado en lenguaje C para posteriormente ser reescrito en C++.
- **Microsoft SQL Server:** es un sistema de gestión de bases de datos relacionales desarrollado por Microsoft. Tiene la capacidad de poner a disposición de muchos usuarios grandes cantidades de datos de manera simultanea.

Algunos de los comandos básicos de SQL se ejemplifican a continuación:

- **Propiedad SELECT:** Hace la selección en una tabla de la base de datos.

```
SELECT * FROM datos
```

Esta sentencia seleccionaría todos, absolutamente todos los registros dentro de la tabla datos.

```
SELECT * FROM datos WHERE usuario='juan'
```

Seleccionaríamos todos los registros dentro de la tabla datos que tengan como usuario a "Juan".

- Propiedad INSERT INTO: Agrega un nuevo registro a la tabla elegida

```
INSERT INTO datos (usuario) VALUES ('nuevo registro')
```

Insertamos en la tabla datos en la columna “usuario”, un registro nuevo.

```
INSERT INTO datos (usuario, edad) VALUES ('juan',20)
```

Aquí insertamos 2 registros al mismo tiempo. Juan en la columna “usuario” y 20 en la columna “edad”.

- Propiedad DELETE: Borra registros de nuestra tabla

```
DELETE FROM datos WHERE usuario = 'juan'
```

Borramos los registros donde el usuario sea "juan".

```
DELETE FROM datos WHERE usuario = 'Mario' AND edad = 16
```

Borramos **solo** los usuarios de nombre Mario que tenían 16 años.

- Propiedad UPDATE: Actualiza registros, modificando datos ya existentes.

```
UPDATE datos SET usuario = 'juan'
```

Esta modificación renombrará todos los usuarios a "juan".

```
UPDATE datos SET usuario = 'Mario' WHERE edad = 16
```

Modificamos **solo** los registros que tenían 16 años. Ahora todos los usuarios de 16 años se llaman "Mario".

Anexo 4: J2SE

Luego de haber redondeado la idea del proyecto, se tuvo que realizar una solución lógica de la misma, de manera de tener un punto de partida para comenzar con el desarrollo del software que fue descrito en los capítulos anteriores. [8]

Una de las primeras decisiones que se tuvieron que tomar fue el lenguaje de programación a utilizar. Lo primero que se tuvo en cuenta fueron los conocimientos actuales de programación que tenían los integrantes del grupo y cuál era el alcance de cada uno de estos lenguajes. [8]

El lenguaje de programación Java fue el elegido, debido a que este era el de mayor conocimiento por parte de los integrantes del proyecto y cuenta con una gran variedad de funcionalidades de utilidad, por ejemplo el mismo es capaz de manejar protocolos como TCP/IP y Bluetooth, lo cual fue determinante para el desarrollo del proyecto. [8]

Al haber elegido java, tuvimos que familiarizarnos con las APIs de uso común del lenguaje, las cuales son implementadas por la colección J2SE. [8]

J2SE conocida como plataforma Java 2, Standard Edition (J2SE), actualmente es denominada Java Platform, Standard Edition (Java SE), consiste en una maquina virtual la cual contiene una colección de Apis del lenguaje de programación Java, de gran utilidad para la mayoría del software desarrollado bajo este lenguaje. [8]

La colección de paquetes provistos por Java SE se puede dividir en dos grandes grupos, un primer grupo que abarca los paquetes de propósito general y un segundo grupo que abarca los paquetes de propósito especial. [8]

A continuación se pasaran a describir los paquetes que componen cada uno de los grupos mencionados. [8]

Paquetes de Propósito General

➤ **Java.lang**

El paquete java.lang contiene clases fundamentales e interfaces fuertemente relacionadas con el lenguaje. Esto incluye las clases raíz que forman la jerarquía de clases, tipos relacionados con la definición del lenguaje, excepciones básicas, funciones matemáticas, Hilos y funciones de seguridad.

Las principales clases de este paquete son:

- **Object:** clase raíz de todas las jerarquías de clases.
- **Class:** clase raíz del sistema de reflexión java.
- **Throwable:** clase base de la jerarquía de clases de excepciones.
- **Thread:** clase que permite operaciones con hilos.
- **String:** clase que permite el manejo de de cadenas String. [9]

➤ **Java.io**

Este paquete contiene clases que soportan entrada/salida. Las clases del paquete son principalmente streams, sin embargo se incluye una clase para ficheros de acceso aleatorio. Las clases principales del paquete son `InputStream` y `OutputStream` las cuales son clases abstractas base para leer de y escribir a streams de bytes respectivamente. Las clases relacionadas `Reader` y `Writer` son clases abstractas base para leer de y escribir a streams de caracteres, respectivamente. [9]

➤ **Java.math**

Este paquete soporta aritmética multiprecision y suministra generadores de números primos multiprecision usados para la generación de claves criptográficas. [9]

➤ **Java.net**

Este paquete suministra rutinas especiales para IO de redes, permitiendo peticiones HTTP, así como también otras transacciones comunes. Como se pudo ver en capítulos anteriores este paquete fue de gran utilidad para la creación de conexiones entre los nodos de la red, mediante la manipulación de Sockets. [9]

➤ **Java.util**

Las estructuras de datos que agregan objetos son el foco del paquete `java.util`. En el paquete está incluida la API `Collectiones`, una jerarquía organizada de estructura de datos influenciada fuertemente por consideraciones de patrones de diseño. [9]

Paquetes de propósito Especial

➤ **Java.applet**

Creado para soportar la creación de applet java, el paquete java.applet permite a las aplicaciones ser descargadas sobre una red y ejecutarse dentro de una sandbox. Las restricciones de seguridad son impuestas fácilmente en la sandbox. [9]

➤ **Java.awt**

La Abstract Window Toolkit contiene rutinas para operaciones básicas GUI y utiliza ventanas básicas desde el sistema nativo subyacente. Muchas implementaciones independientes de la API java implementan todo excepto AWT, el cual no es usado por la mayoría de las aplicaciones del lado servidor. Este paquete también contiene la API de gráficos Java 2D. [9]

➤ **Java.sql**

Una implementación de la API JDBC (usada para acceder a bases de datos SQL) se agrupa en el paquete. [9]

➤ **Javax.Swing**

Swing es una colección de rutinas que se construyen sobre java.awt para suministrar un toolkit de widgets independiente de plataforma. Swing usa las rutinas de dibujado 2D para renderizar los componentes de interfaz de usuario en lugar de confiar en el soporte GUI nativo subyacente del Sistema operativo. [9]

Swing es un sistema muy rico por sí mismo, soportando pluggable looks and feels (PLAFs) para que los controles (widgets) en la GUI puedan imitar a aquellos del sistema nativo subyacente. Los patrones de diseño impregnan el sistema, especialmente una modificación del patrón modelo-vista-controlador, el cual afloja el acoplamiento entre función y apariencia. Una inconsistencia es que (para J2SE 1.3) las fuentes son dibujadas por el sistema nativo subyacente, limitando la portabilidad de texto. Mejoras, tales como usar fuentes de mapas de bits, existen. En general, las layouts(disposiciones de elementos) se usan y mantienen los elementos dentro de una GUI consistente a través de distintas plataformas. [9]

Es una plataforma independiente, Model-View-Controller Gui framework para Java. Sigue un simple modelo de programación por hilos, y posee las siguientes características principales:

- **Independencia de la plataforma**
- **Extensibilidad:** es una arquitectura altamente particionada: los usuarios pueden proveer sus propias implementaciones modificadas para sobrescribir las implementaciones por defecto. Se puede extender clases existentes proveyendo alternativas de implementación para elementos esenciales.
- **Customizable:** dado el modelo de representación programático del framework de swing, el control permite representar diferentes 'look and feel' (desde MacOS look and feel hasta Windows XP look and feel). Además, los usuarios pueden proveer su propia implementación look and feel, que permitirá cambios uniformes en el look and feel existente en las aplicaciones Swing sin efectuar ningún cambio al código de aplicación. [9]

Anexo 5: Java 2 Micro Edition (J2ME)

Dada la naturaleza de nuestro proyecto se nos presentaba inevitablemente la necesidad de encarar el desarrollo de aplicaciones que se ejecuten sobre un teléfono celular ordinario, esto nos condujo, casi obligadamente, a introducirnos en la programación mediante Java 2 Micro Edition. Podemos describir brevemente a J2ME como una especificación de un subconjunto de la plataforma Java orientada a proveer una colección certificada de APIs de desarrollo de software para dispositivos con recursos restringidos. Está orientado básicamente a productos de consumo como PDAs, teléfonos móviles o electrodomésticos. [10]

J2ME está enfocada a la aplicación de la tecnología Java en dispositivos electrónicos con capacidades computacionales y gráficas muy reducidas, tales como teléfonos móviles (nuestro caso), PDAs o electrodomésticos inteligentes. Esta edición en particular tiene componentes básicos que la diferencian de las demás versiones, como el uso de una máquina virtual denominada KVM (Kilo Virtual Machine, debido a que requiere sólo unos pocos Kilobytes de memoria para funcionar) en vez del uso de la JVM clásica, la inclusión de un pequeño y rápido recolector de basura entre otras diferencias. [11]

Las diferentes funcionalidades y alcances de J2ME se clasifican en lo que se conoce como Configuraciones y Perfiles, que responden a los distintos tipos de dispositivos sobre los cuales se ejecutará la aplicación desarrollada. Entendemos por Configuración a un conjunto de clases básicas orientadas a conformar el corazón de las implementaciones para dispositivos de características específicas. Existen 2 configuraciones definidas en J2ME: Connected Limited Device Configuration (CLDC) enfocada a dispositivos con restricciones de procesamiento y memoria, y Connected Device Configuration (CDC) enfocada a dispositivos con más recursos. En nuestro caso utilizaremos la Configuración CLDC 1.1 (JSR 139). Un Perfil en cambio comprende bibliotecas Java de clases específicas orientadas a implementar funcionalidades de más alto nivel para familias específicas de dispositivos, en el caso de Umaguma el Perfil utilizado es MIDP 2.0 (JSR 118). [11]

Todas las aplicaciones y librerías necesarias para llevar a cabo un desarrollo completo en J2ME las obtuvimos del paquete (desarrollado por Sun Microsystems) conocido como Wireless Toolkit 2.5.2 for CLDC, este incluye entre otras cosas las clases y API's básicas, el pre-verificador, el compilador y los simuladores de dispositivos celulares. Como entorno de desarrollo (IDE) utilizamos el programa (open source) Eclipse, desarrollado por Eclipse Foundation. [12] [13]

Referencias

- [1] Página oficial de la tecnología Bluetooth
<http://spanish.bluetooth.com>
- [2] Wikipedia, la enciclopedia libre - Bluetooth
<http://es.wikipedia.org/wiki/Bluetooth>
- [3] Wikipedia, la enciclopedia libre – Bluetooth (especificación)
[http://es.wikipedia.org/wiki/Bluetooth_\(especificaci%C3%B3n\)](http://es.wikipedia.org/wiki/Bluetooth_(especificaci%C3%B3n))
- [4] Proyecto Bluetooth FIE UTP – Documentación, Capitulo 1
<http://ohm.utp.edu.co/bluetooth/docu/cap1.pdf>
- [5] Wikipedia, la enciclopedia libre – Transmission Control Protocol (TCP)
http://es.wikipedia.org/wiki/Transmission_Control_Protocol
- [6] Wikipedia, la enciclopedia libre – Protocolo de Internet (IP)
http://es.wikipedia.org/wiki/Protocolo_de_Internet
- [7] Wikipedia, la enciclopedia libre – SQL
<http://es.wikipedia.org/wiki/SQL>
- [8] Wikipedia, la enciclopedia libre – Java SE (J2SE)
http://es.wikipedia.org/wiki/Java_SE
- [9] Wikipedia, la enciclopedia libre – Swing (biblioteca gráfica)
http://es.wikipedia.org/wiki/Swing_%28biblioteca_gr%C3%A1fica%29
- [10] Wikipedia, la enciclopedia libre – Java Micro Edition (J2ME)
http://es.wikipedia.org/wiki/Java_Micro_Edition
- [11] Java a tope: J2ME
Sergio Gálvez Rojas y Lucas Ortega Díaz
- [12] Java ME Technology APIs & Docs
<http://java.sun.com/javame/reference/apis.jsp>
- [13] Eclipse.org Home
<http://www.eclipse.org/>