

Universidad de las Ciencias Informáticas

Facultad 2



**Título: BluePub Sender: Sistema de Envío de
Publicidad vía Bluetooth.**

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

Autores: Graviel Ortega Aponte.

Yailemi Guerra Reyes.

Tutores: Ing. Joan Martínez Herrera.

Ing. Félix Maikel García Pérez.

La Habana

2012



“Todos y cada uno de nosotros paga puntualmente su cuota de sacrificio consciente de recibir el premio en la satisfacción del deber cumplido, conscientes de avanzar con todos hacia el Hombre Nuevo que se vislumbra en el horizonte.”

Ernesto “Che” Guevara

DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de este trabajo y autorizo al Centro de Telemática de la Facultad 2 de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

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

Autores:

Graviel Ortega Aponte

Yailemi Guerra Reyes

Tutores:

Ing. Joan Martínez Herrera

Ing. Félix Maikel García Pérez

DATOS DE CONTACTO

AGRADECIMIENTOS

Queremos agradecer de forma común a aquellas personas que formaron parte del desarrollo de este trabajo de diploma, personas que siempre estuvieron dispuestas a ayudarnos en el momento que lo necesitamos; aún con sus cargas de trabajo supieron sacar un tiempo para atender nuestra necesidad.

A Salvador González Gómez por su ayuda incondicional.

A Aimet María Bajuelo Tejido por dedicarnos tiempo para el diseño de los logotipos del sistema y sus sugerencias que siempre fueron muy bienvenidas.

A todas las personas que incondicionalmente nos facilitaron sus teléfonos celulares para realizar las pruebas del sistema.

Al tribunal y al oponente por ser tan exigentes en el desarrollo de este trabajo de diploma.

A los tutores Joan y Félix, por hacernos sentir parte de la familia, por enseñarnos lo que sabemos.

Felipao, aunque no te dejábamos descansar y al llamado del almuerzo no te permitíamos asistir a tiempo, gracias por los momentos de alegría que nos hacías pasar.

Joan, gracias por la dedicación y entrega a la hora de explicar cualquier duda, y gestionar los dispositivos que necesitamos.

Graviel

Quiero agradecer a Aquel que me ha dado la vida y las fuerzas para llegar hasta este momento, a quién en momentos de tristeza y soledad me dio consuelo y fuerzas para continuar. A mi Dios que sin Él no hubiese sido posible este trabajo.

A mis padres Beny y Galy por haber sido los instrumentos que Dios usó para educarme y formarme, por darme lo que necesitaba siempre además de todo su amor, cariño y comprensión.

A mi abuelo Hermide, no soy médico, pero ya soy ingeniero, gracias por quererme tanto.

A mis abuelas Nelda y Edalia por todo el amor y cariño que han sabido darme.

A mis primos y tíos por formar una gran familia.

A mi novia por su amor, apoyo y comprensión en los momentos de tensión y estrés.

A mis pastores por la guianza espiritual: Obispo Ricardo Pereira, Rev. Roberto Díaz, Rev. Jorge Ortega y sus familias.

A mis hermanos de la UCI, todos los que conocí y conozco porque aportaron su granito de arena en mi crecimiento y formación, por ser mi familia estos cinco años.

A José Alejandro (el Calde), Driggs, Roxana, Niorges, Leandro, Ángel Delvis, Reiniel y Yarima, Ronni, Alfredo, Alberto, Leidis, Yisel, Yordanka, Yoelvis, Leordanis, Bairam y Juan Fuentes.

A mis compañeros de aula desde el primer año hasta ahora.

A los muchachos del apartamento: Darlyn, Eduardo, Manuel Gutierrez, Manuel Lazo y Reinel.

A todos muchas gracias por ser parte de mí.

Yaileni

A mi mamita Meyvis y a mi papito Carlitos por siempre estar a mi lado, por su preocupación, entrega, amor, cariño y atención. Por ser los mejores padres del mundo.

A mi hermanita Yailen por darme su amor incondicional.

A toda la familia por siempre brindarme apoyo en todos los momentos de mi vida.

A Luna que ha sido mi amiga en la Universidad donde hemos compartido cinco años de estudio, risas, cueros y llantos.

A Nelsito por brindándome su atención y cariño desde el primer día y por siempre estar aconsejándome.

A mi compañero de tesis Graviel "el gordo", con él compartí los tres primeros y el último año de la universidad, a pesar de que discutimos un poco siempre nos llevamos bien y aunque no me lo diga yo sé que le gustó hacer la tesis conmigo.

A Dios y a mi Virgen de Regla.

Graviel

Dedico esta tesis a mis padres, por haberme formado como un hombre de bien, y he aquí el resultado de sus 23 años de esfuerzo y dedicación.

A mi novia Dayana por haber llegado a mi vida en el momento indicado.

A mi familia por el apoyo y la confianza que tuvieron siempre en mí.

A mis padres y mi hermanita por estar siempre conmigo, brindarme amor, comprensión y ayuda.

A toda la familia por siempre confiar en mí.

A quien ya no está físicamente pero si en mi corazón, por haber estado siempre pendiente de mis cosas desde que nací.

Yailemi

RESUMEN

La publicidad es un área muy importante para muchas empresas, centros comerciales y entidades de venta. Los métodos utilizados para promover sus productos muchas veces no llegan a la mayoría de los usuarios potenciales. En el proceso de gestión de la publicidad existen problemas que afectan su correcto funcionamiento. En ocasiones no se tiene el control de qué publicidades se dieron, en qué momento y a quién se le remitió. La publicidad por proximidad haciendo uso de la tecnología Bluetooth abre múltiples opciones para empresas, tiendas y establecimientos comerciales, permitiendo tener una comunicación directa con los clientes, informándole de sus ofertas. El presente trabajo pretende desarrollar un sistema que gestione el envío de publicidad a dispositivos móviles mediante tecnología Bluetooth. Para ello se describen los procesos de envío de publicidad con el fin de mejorar el trabajo de los publicistas y propiciar el ahorro de recursos en las campañas publicitarias. Se obtuvo como resultado un sistema con un grupo de funcionalidades que facilitarán a los agentes publicitarios realizar su trabajo de forma más eficiente, permitiendo generar reportes estadísticos a través de gráficas que muestran el comportamiento de los envíos publicitarios, dando solución a una serie de problemas que existen en la realización del proceso publicitario.

PALABRAS CLAVE

Bluetooth, Publicidad, Aplicaciones Distribuidas, ServicioOBEX (*Object Exchange*).

TABLA DE CONTENIDOS

INTRODUCCIÓN5

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA.....9

1.1 Introducción 9

1.2 Estado del Arte 9

1.3 Tecnologías a utilizar 10

1.3.1 Bluetooth 11

1.3.2 Lenguaje de Programación 12

1.3.3 Lenguaje de Modelado. UML (*Unified Modeling Language*)..... 13

1.3.4 Frameworks a utilizar 13

1.3.5 JDBC (*Java Database Connectivity*) 16

1.3.6 Socket 16

1.3.7 API para Bluetooth. BlueCove 2.1.0..... 18

1.4 Metodología de Desarrollo 18

1.4.1 FDD (*Feature Driven Development*)..... 18

1.5 Herramientas 19

1.5.1 IDE (*Entorno Integrado de Desarrollo*). Eclipse 3.5.0..... 19

1.5.2 Servidor Web. Apache Tomcat 6.0.26 19

1.5.3 Sistema Gestor de Base de Datos. PostgreSQL 8.4.0 20

1.5.4 Herramienta de Modelado. Visual Paradigm 8.0 20

1.5.5 JfreeChart..... 21

1.6 Conclusiones Parciales..... 21

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA22

2.1 Introducción 22

2.2 Problemática..... 22

2.3 Objeto de automatización 22

2.4 Propuesta del sistema 22

2.5 Modelo de Dominio 23

2.6 Relación de los requerimientos 24

2.6.1 Requerimientos funcionales..... 24

2.6.2 Requerimientos no funcionales..... 26

2.7 Definición de los casos de uso..... 27

TABLA DE CONTENIDOS

2.7.1	Definición de los actores del sistema	27
2.7.2	Diagrama de casos de uso	29
2.7.3	Descripción de casos de uso	29
2.8	Conclusiones Parciales.....	34
CAPÍTULO 3: DISEÑO DEL SISTEMA		35
3.1	Introducción	35
3.2	Arquitectura	35
3.2.1	Cliente-Servidor	35
3.2.2	Patrón Arquitectónico N Capas.....	36
3.2.3	Patrón Arquitectónico MVC (<i>Modelo Vista Controlador</i>).....	37
3.3	Patrones de Diseño	38
3.3.1	Patrones GRASP (<i>General Responsibility Assignment Software Patterns</i>).....	38
3.3.2	Patrón Inversión de Control.	40
3.3.3	Patrón DAO (<i>Data Access Object</i>).....	41
3.4	Diseño	42
3.5	Diagrama de Paquetes	42
3.6	Diagrama de Clases del Diseño.....	42
3.6.1	Diagrama de Clases del Caso de Uso Gestionar Publicidad.....	43
3.6.2	Diagrama de Clases del Caso de Uso Enviar Publicidad.	44
3.7	Diagramas de interacción. Diagrama de secuencia	44
3.7.1	Diagrama de Secuencia del Caso de Uso Enviar Publicidad.	45
3.8	Diseño de la base de datos.....	46
3.8.1	Modelo de Datos.....	46
3.9	Conclusiones Parciales.....	47
CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBA		48
4.1	Introducción	48
4.2	Implementación	48
4.2.1	Diagrama de despliegue	48
4.2.2	Diagrama de componentes.....	49
4.3	Pruebas de software	50
4.3.1	Pruebas Unitarias	51
4.3.2	Modelo de prueba. Pruebas de Funcionalidad.....	51

4.3.3	No conformidades detectadas en el sistema.....	57
4.4	Conclusiones Parciales.....	58
CONCLUSIONES		59
RECOMENDACIONES		60
REFERENCIAS BIBLIOGRÁFICAS		61
BIBLIOGRAFÍA.....		63
ANEXOS.....		¡ERROR! MARCADOR NO DEFINIDO.
Anexo 1: Entrevista sobre la Gestión de la Publicidad.		¡Error! Marcador no definido.
Anexo 2: Descripción de los Casos de Uso del Sistema		¡Error! Marcador no definido.
Tabla 1: CU Autenticar Usuario.		¡Error! Marcador no definido.
Tabla 2: CU Controlar Envío de Publicidad.		¡Error! Marcador no definido.
Tabla 3: CU Comprobar Servidores.		¡Error! Marcador no definido.
Tabla 4: CU Gestionar Usuarios.....		¡Error! Marcador no definido.
Tabla 5: CU Gestionar Servidores.....		¡Error! Marcador no definido.
Tabla 6: CU Generar Reportes.....		¡Error! Marcador no definido.
Anexo 3: Diagramas de Clases del Diseño		¡Error! Marcador no definido.
Figura 1: Diagrama de Clases del Caso de Uso Autenticar Usuario.....		¡Error! Marcador no definido.
Figura 2: Diagrama de Clases del Caso de Uso Gestionar Usuario.		¡Error! Marcador no definido.
Figura 3: Diagrama de Clases del Caso de Uso Gestionar Servidores.....		¡Error! Marcador no definido.
Figura 4: Diagrama de Clases del Caso de Uso Controlar Envío de Publicidad.¡		¡Error! Marcador no definido.
Figura 5: Diagrama de Clases del Caso de Uso Comprobar Servidores.		¡Error! Marcador no definido.
Figura 6: Diagrama de Clases del Caso de Uso Generar Reportes.....		¡Error! Marcador no definido.
Anexo 4: Diagramas de Secuencia		¡Error! Marcador no definido.
Figura 1: Diagrama de Secuencia del Escenario Añadir Publicidad, CU Gestionar Publicidad.¡		¡Error! Marcador no definido.
Figura 2: Diagrama de Secuencia del Escenario Consultar Publicidad, CU Gestionar Publicidad.¡		¡Error! Marcador no definido.
Figura 3: Diagrama de Secuencia del Escenario Eliminar Publicidad, CU Gestionar Publicidad.¡		¡Error! Marcador no definido.
Figura 4: Diagrama de Secuencia del Caso de Uso Autenticar Usuario.		¡Error! Marcador no definido.
Figura 5: Diagrama de Secuencia del Escenario Añadir Usuario, CU Gestionar Usuario.¡		¡Error! Marcador no definido.
Figura 6: Diagrama de Secuencia del Escenario Modificar Usuario, CU Gestionar Usuario.¡		¡Error! Marcador no definido.
Figura 7: Diagrama de Secuencia del Escenario Eliminar Usuario, CU Gestionar Usuario.¡		¡Error! Marcador no definido.
Figura 8: Diagrama de Secuencia del Escenario Añadir Servidor, CU Gestionar Servidor.¡		¡Error! Marcador no definido.

TABLA DE CONTENIDOS

Figura 9: Diagrama de Secuencia del Escenario Modificar Servidor, CU Gestionar Servidor.	¡Error! Marcador no definido.
Figura 10: Diagrama de Secuencia del Escenario Eliminar Servidor, CU Gestionar Servidor.	¡Error! Marcador no definido.
Figura 11: Diagrama de Secuencia del Escenario Ejecutar Aplicación, CU Controlar Envío de Publicidad.	¡Error! Marcador no definido.
Figura 12: Diagrama de Secuencia del Escenario Detener Aplicación, CU Controlar Envío de Publicidad.	¡Error! Marcador no definido.
Figura 13: Diagrama de Secuencia del Caso de Uso Generar Reportes.	¡Error! Marcador no definido.
Anexo 5: Diagramas de Componentes.	¡Error! Marcador no definido.
Figura 1: Diagrama de Componentes de Usuarios.	¡Error! Marcador no definido.
Figura 2: Diagrama de Componentes de Servidores.	¡Error! Marcador no definido.
Figura 3: Diagrama de Componentes de Servidor.	¡Error! Marcador no definido.
Figura 4: Diagrama de Componentes de Reportes.	¡Error! Marcador no definido.
Anexo 6: Casos de Prueba	¡Error! Marcador no definido.
GLOSARIO	64

INTRODUCCIÓN

La publicidad es una herramienta básica en la comunicación entre personas, es parte del entorno social, cultural y comercial, tan así que origina cambios en la conducta y el comportamiento del público que la consume. Desde tiempos antiguos ha sido una necesidad para las diferentes empresas y negocios dar a conocer a sus clientes los productos que ofertan, esto en la mayoría de los casos es una tarea compleja, ya que consume una gran cantidad de recursos. A través de los años se han utilizado varias formas y técnicas de difundir publicidad como son la expresión oral, anuncios de estilo grafiti, utilización de cuernos, tambores y campanas para atraer la atención del público; el surgimiento de la imprenta permitió la difusión más extensa de los mensajes publicitarios con la impresión de periódicos y almanaques.

En los tiempos actuales la publicidad ha evolucionado en gran medida con respecto a los métodos utilizados en tiempos remotos, el surgimiento de la radio y la televisión, los anuncios en exteriores como vallas, marquesinas, transportes públicos, letreros luminosos; anuncios cerrados, que se exhiben en videojuegos, películas y series; anuncios en puntos de venta que se sitúan en el lugar que se realizará la venta, han sido factores claves para este desarrollo.

La telefonía celular surge en la década de los 70 del siglo XIX y crece a pasos agigantados en el mundo. Estos aparatos cada vez más pequeños representan, mejor que ningún otro medio en la actualidad, un punto de convergencia tecnológica digno de consideración: además de transmitir conversaciones, los teléfonos celulares incursionan en el envío de mensajes cortos, de imágenes en movimiento y de algunos servicios de Internet. Además, conlleva una carga simbólica interesante en cuanto a su uso: sobre la mesa, en el auto, en el cine, en la calle. [1]

Hay diversas maneras de conectar dispositivos electrónicos, mediante cables, señales de radio y rayos de luz infrarrojos, una variedad incluso mayor de conectores, enchufes y protocolos, por lo que el arte de conectarlos entre sí es cada día más complejo; con el surgimiento de Internet y las tecnologías inalámbricas como teléfonos móviles, mandos a distancia, ordenadores portátiles, surge la necesidad de intercambiar información entre ellos y enviar archivos; una de las tecnologías creadas para dar solución a esta necesidad es la tecnología Bluetooth (tecnología de ondas de radio de corto alcance), gracias a esta tecnología los dispositivos que la implementan pueden comunicarse entre ellos cuando se encuentran dentro de su área de cobertura. La comunicación puede establecerse a través de objetos creando, una Red de Área Personal (PAN) entre un dispositivo y otro.

La publicidad por proximidad haciendo uso de la tecnología Bluetooth abre un sin número de opciones para empresas, tiendas y establecimientos comerciales, permitiendo tener una comunicación directa con los clientes, informándole de nuevos productos, rebajas de precios, entre otros servicios.

En Cuba, el envío de publicidad haciendo uso de las nuevas tecnologías de la información es un campo casi inexplorado, los centros comerciales y entidades de venta, en su mayoría no cuentan con un sistema informático que les permita dar publicidad a los productos que ofertan y que esta publicidad llegue de forma directa a los usuarios. Los métodos usados son muy rudimentarios, entre ellos se encuentran los afiches en el interior de los salones comerciales; los seminarios y cursos que se les brindan a los trabajadores encargados de la venta de los productos y de relacionarse directamente con el cliente. Es necesario tener conocimiento sobre la gestión de la publicidad, qué información se transmitió a los usuarios y quién lo hizo. Es importante conocer el nivel de aceptación que tuvo determinada publicidad entre los usuarios, por lo que se hace necesario utilizar nuevas técnicas automatizadas que faciliten el trabajo de los publicistas y el ahorro de recursos. Para la automatización del envío de la publicidad es necesario un sistema que gestione el proceso de envío vía Bluetooth hacia los dispositivos móviles en varias localidades de un mismo establecimiento.

Por lo expuesto anteriormente surge el siguiente **problema a resolver**: ¿Cómo enviar publicidad de manera frecuente, organizada y gestionada hacia dispositivos móviles?

A partir del problema a resolver se puede definir como **objeto de estudio**: el proceso de envío de publicidad, teniendo como **campo de acción**: el envío de publicidad a dispositivos móviles que utilicen tecnología Bluetooth en Cuba.

El **objetivo general** es desarrollar un sistema que gestione el envío de publicidad a dispositivos móviles mediante tecnología Bluetooth.

Los **objetivos específicos** para profundizar el objetivo general son:

- Desarrollar una aplicación cliente que permita la gestión de la publicidad.
- Desarrollar una aplicación servidora que realice el envío de publicidad hacia los dispositivos móviles a través de la tecnología Bluetooth.
- Desarrollar un mecanismo de interconexión entre la aplicación cliente y la aplicación servidora para el envío de publicidad.

Para dar cumplimiento a estos objetivos se definieron las siguientes **tareas de la investigación**:

- Realizar un estudio de las aplicaciones de envío de publicidad vía Bluetooth en el mercado nacional e internacional.

- Realizar un estudio sobre el funcionamiento de Bluetooth.
- Seleccionar la metodología, tecnología, herramientas y lenguaje de programación a utilizar para desarrollar el sistema.
- Definir los frameworks a utilizar.
- Seleccionar librerías para el trabajo con Bluetooth.
- Definir cómo realizar el descubrimiento de dispositivos móviles con tecnología Bluetooth en el área de cobertura.
- Definir el gestor de bases de datos a utilizar.
- Definir herramienta para generar gráficas para monitorizar el envío de publicidad.
- Diseñar un mecanismo para mantener una cola de publicidad.
- Realizar un mecanismo de encuesta a los servidores para comprobar su correcto funcionamiento.
- Realizar un estudio del funcionamiento de un hilo de envío de publicidad.
- Definir qué patrones de diseño y arquitectura se ajustan al desarrollo de la solución informática.

Para desarrollar el sistema surgen las siguientes **preguntas científicas**:

- ¿Cómo funciona el proceso de envío de datos a un dispositivo inalámbrico a través de la tecnología Bluetooth?
- ¿Qué mecanismo utilizar para notificar al administrador del sistema que ocurrió un error en el proceso de envío de publicidad?
- ¿Cómo administrar un cliente remoto?
- ¿Existen APIs (*Application Programming Interface*) de envío de información vía Bluetooth para realizar transferencias de archivos?
- ¿Qué API utilizar para el envío de información vía Bluetooth?
- ¿Cómo realizar el descubrimiento de dispositivos móviles en el área de cobertura?
- ¿Qué mecanismo utilizar para comprobar el correcto funcionamiento de los servidores de envío de publicidad?

Para la realización de las tareas de la investigación se utilizaron los siguientes métodos de investigación:

Métodos Teóricos:

Análítico – Sintético: se aplica para analizar la documentación referente al tema de investigación, permitiendo la extracción de los elementos más importantes que se relacionan con el objeto de estudio

para lograr una mejor comprensión del mismo. La aplicación del método se evidencia en el capítulo uno en los epígrafes Tecnologías a Utilizar, Metodología de Desarrollo y Herramientas.

Histórico – Lógico: se utiliza para realizar un estudio del estado del arte de los sistemas de envío de publicidad en el mundo y en Cuba; además se estudiaron diferentes tecnologías y herramientas dando la posibilidad de analizar la trayectoria histórica de los mismos, así como su evolución y desarrollo.

Métodos Empíricos:

Entrevista: se utilizó con el fin de investigar las vías de publicidad existentes en Cuba para promocionar los productos que se ofertan en tiendas y centros comerciales a disposición de los clientes.

El presente trabajo de diploma está compuesto por cuatro capítulos, de los cuales se muestra un resumen a continuación:

Capítulo I “Fundamentación Teórica”: se exponen los aspectos teóricos relacionados con el desarrollo de este trabajo. Se hace un estudio de los antecedentes del sistema; de las tecnologías que se usan en la solución y se les da fundamento a las metodologías, lenguajes, conceptos y herramientas utilizadas.

Capítulo II “Características del Sistema”: se da una propuesta del sistema, se describe el modelo de dominio, se especifican los requisitos que debe cumplir el sistema y se definen los casos de uso.

Capítulo III “Diseño del Sistema”: se elaboran los diagramas del diseño que propone la metodología empleada. Muestra la descripción de la arquitectura y patrones de diseño empleados. Incluye el diseño de la base de datos.

Capítulo IV “Implementación y Prueba”: contiene los diagramas de componentes y el diagrama de despliegue, a través del cual se puede ver la distribución del sistema de forma física. Contiene además la realización y descripción de los casos de prueba.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

En este capítulo se hace un estudio de las aplicaciones informáticas que gestionan el envío de publicidades utilizando la tecnología Bluetooth. Se definen las herramientas, lenguajes y metodologías que serán utilizadas para el desarrollo de la aplicación.

1.2 Estado del Arte

Para conocer cómo se comporta en el mundo el uso de las tecnologías de la información para gestionar el proceso de envío de publicidad se realizó un estudio del estado del arte. A continuación se muestran ejemplos de aplicaciones de envío de publicidad que utilizan la tecnología Bluetooth:

SEM-BT (*Sistema de Envío Masivo vía Bluetooth*) es un sistema de transmisión masiva de información y publicidad, el cual ha sido desarrollado por IngSoft I+D+I¹. El sistema está basado en la tecnología Bluetooth que permite enviar información multimedia (Publicidad, Catálogos, Juegos, etc.) a terminales móviles tales como teléfonos móviles y PDAs (*Personal Digital Assistant*) encontradas en su radio de acción. Este sistema permite hasta 7 conexiones simultáneas; cuenta con un registro de datos donde almacena información sobre los envíos aceptados, rechazados e ignorados. El SEM-BT soporta todo tipo de archivos, imágenes, videos, audio, juegos, documentos y programas; teniendo un radio de acción de cien metros en espacios abiertos. Con este sistema se reduce el gasto innecesario de papel disminuyendo la publicidad escrita por su correspondiente formato digital. [2]

BluetoothSENDING proporciona herramientas avanzadas para crear, gestionar y controlar campañas de Marketing Móvil de forma sencilla y eficaz. Permite adaptar de forma automática contenido multimedia como video, audio, imágenes, animaciones, cupones, tarjetas personales electrónicas, textos y concursos al tamaño de pantalla y requisitos técnicos de cada teléfono móvil. El sistema permite el envío de contenido digital vía Bluetooth de forma estratégica, genera gráficas y estadísticas en tiempo real, diseña y adapta la publicidad de acuerdo a la necesidad del cliente. [3]

BlueMessenger es un sistema diseñado por Ditecom² para el envío de publicidad a móviles que tengan activado el Bluetooth. Permite crear, modificar y ver las estadísticas de cualquier campaña; enviar cualquier tipo de archivo publicitario (imágenes, videos, música, juegos, programas java) en un radio de acción de hasta

¹ Empresa especializada en el desarrollo e innovación de software para su integración en dispositivos tecnológicos.

² Empresa dedicada al diseño y la distribución de instrumentación electrónica, especialistas en instrumentación basada en PC.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

cien metros alrededor del servidor de publicidad. Los servidores de publicidad BlueMessenger reconocen gran parte de los móviles del mercado, adaptando automáticamente el tamaño de las imágenes para que se ajusten al tamaño de la pantalla de cada móvil. [4]

En Cuba, el envío de publicidad utilizando las tecnologías de la información es un área que no se ha explotado al máximo, existe un sistema llamado BlueSpace que es un software de marketing de proximidad mediante antena Bluetooth, que se basa en el envío de contenidos a los teléfonos móviles que se encuentran en el área de influencia del dispositivo emisor. El objetivo fundamental es llegar a las personas, con informaciones valiosas, en el lugar y momento adecuados a través de sus dispositivos móviles. Muy recomendado para eventos, ferias, discotecas y recepciones de las instalaciones. Este sistema es capaz de enviar tipos de contenidos como textos, imágenes, multimedia y tonos para móviles. [5]

Algunas de las ventajas que tiene este sistema son: [5]

- Interfaz simple y cómoda de usar.
- Permite detección y envío simultáneo de contenido a todos los dispositivos móviles ubicados en el área de cobertura Bluetooth.
- Realiza control de la cantidad de dispositivos encontrados, las peticiones aceptadas y rechazadas por los clientes y el número de repeticiones, identificando si el terminal ya ha recibido el contenido.
- Una vez aceptado el contenido no es vuelto a enviar.
- Es muy liviano y consume muy pocos recursos.

Luego de analizar los sistemas existentes, se llega a la conclusión de que no se puede utilizar ninguno de ellos por las características que presentan, estando algunos bajo licencias privativas y otros son sistemas que se encuentran ubicados en una misma computadora por lo que no cumplen con las condiciones que requiere BluePub Sender: Sistema de Envío de Publicidad de poder establecer varios servidores de envío en diferentes departamentos de una misma entidad de venta.

1.3 Tecnologías a utilizar

El avance de la ciencia y la técnica ha facilitado el trabajo a sus usuarios en todo el mundo. Las tecnologías actuales cada vez evolucionan más, por lo que es necesario mantener una constante actualización sobre el desarrollo de las mismas. Para la realización de este sistema es necesario contar con tecnologías que permitan llegar a una solución del problema a resolver.

1.3.1 Bluetooth

“Bluetooth es una tecnología de radio de corto alcance, que permite conectividad inalámbrica entre dispositivos remotos. Se diseñó pensando básicamente en tres objetivos: pequeño tamaño, mínimo consumo y bajo precio.” [6]

Bluetooth opera en una banda de radio a 2.4 GHz. “Su máxima velocidad de transmisión de datos es de 1 Mbps. El rango de alcance Bluetooth depende de la potencia empleada en la transmisión. La mayor parte de los dispositivos que usan Bluetooth transmiten con una potencia nominal de salida de 0 dBm, lo que permite un alcance de unos 10 metros en un ambiente libre de obstáculos.” [6]

La información que se intercambia entre dos unidades Bluetooth se realiza mediante un conjunto de ranuras que forman un paquete de datos. Cada paquete comienza con un código de acceso de 72 bits, que se deriva de la identidad maestra, seguido de un paquete de datos de cabecera de 54bits. Éste contiene importante información de control, como tres bits de acceso de dirección, tipo de paquete, bits de control de flujo, bits para la retransmisión automática de la pregunta, y chequeo de errores de campos de cabecera. La dirección del dispositivo es en forma hexadecimal. Finalmente, el paquete que contiene la información, que puede seguir al de la cabecera, tiene una longitud de 0 a 2745 bits. [6]

Si un equipo se encuentra dentro del radio de cobertura de otro, éstos pueden establecer conexión entre ellos. Cada dispositivo tiene una dirección única de 48 bits, basada en el estándar IEEE (*Institute of Electrical and Electronics Engineers*) 802.11 para WLAN (*Wireless Local Area Network*). En principio sólo son necesarias un par de unidades con las mismas características de hardware para establecer un enlace. Dos o más unidades Bluetooth que comparten un mismo canal forman una piconet. [6]

Para una mejor comprensión de lo antes explicado se muestra a continuación la siguiente imagen: [6]

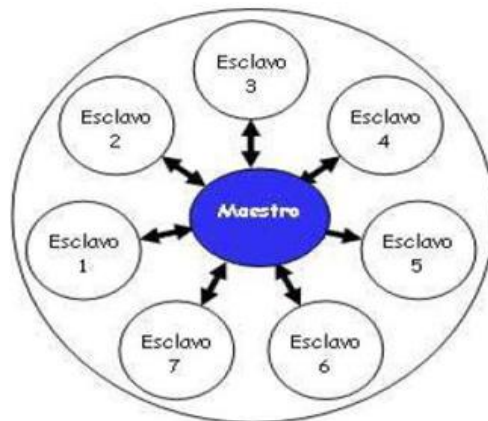


Figura 1: Piconet.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

Para regular el tráfico en el canal, una de las unidades participantes se convertirá en maestra, 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. Hasta ocho usuarios o dispositivos pueden formar una piconet y hasta diez piconets pueden coexistir en una misma área de cobertura. [6]

Los equipos que comparten un mismo canal sólo pueden utilizar una parte de su capacidad. Aunque los canales tienen un ancho de banda de un 1Mbit, cuantos más usuarios se incorporan a la piconet, disminuye la capacidad hasta unos 10 Kbit/s aproximadamente. Para poder solucionar este problema se adoptó una solución de la que nace el concepto de scatternet. Las unidades que se encuentran en el mismo radio de cobertura pueden establecer potencialmente comunicaciones entre ellas. Sin embargo, sólo aquellas unidades que realmente quieran intercambiar información comparten un mismo canal creando la piconet. Este hecho permite que se creen varias piconets en áreas de cobertura superpuestas. [6]

Para comprender lo explicado anteriormente se muestra a continuación la siguiente imagen: [6]

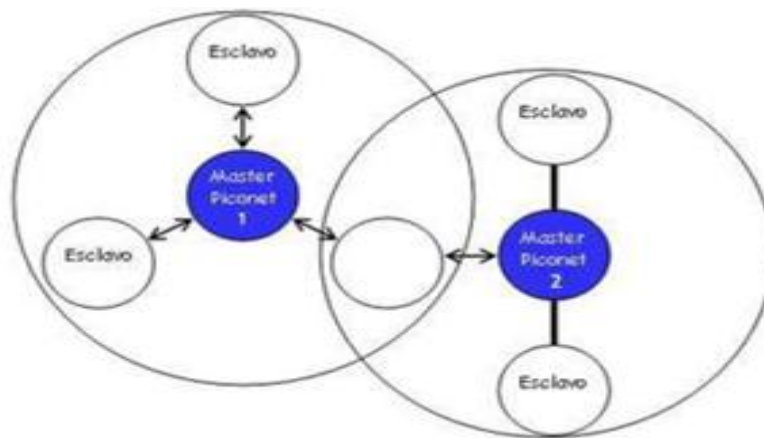


Figura 2: Scatternet.

1.3.2 Lenguaje de Programación

“Un lenguaje de programación es un conjunto de símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones, y es utilizado para controlar el comportamiento físico y lógico de una máquina.” [7]

1.3.2.1 Java

Dentro de los lenguajes de programación se encuentra Java que es desarrollado por Sun Microsystems a principio de los años noventa. Algunas de las características de Java son: [8]

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

- Orientado a objetos: diseñado como un lenguaje orientado a objetos. Los objetos agrupan en estructuras encapsuladas tanto sus datos como los métodos que manipulan esos datos. Apunta hacia la programación orientada a objetos, especialmente en entornos cada vez más complejos y basados en red.
- Distribuido: proporciona una colección de clases para su uso en aplicaciones de red, que permiten abrir sockets, establecer y aceptar conexiones con servidores o clientes remotos, facilitando así la creación de aplicaciones distribuidas.
- Robusto: diseñado para crear software altamente fiable. Proporciona numerosas comprobaciones en compilación y en tiempo de ejecución. Sus características de memoria liberan a los programadores de una familia entera de errores, ya que se ha prescindido por completo de los punteros, y la recolección de basura elimina la necesidad de liberación explícita de memoria.
- Seguro: dada la naturaleza distribuida de Java, la seguridad se impuso como una necesidad de vital importancia. Se implementaron barreras de seguridad en el lenguaje y en el sistema de ejecución en tiempo real.

Se seleccionó Java como lenguaje de programación por la experiencia del equipo de desarrollo en el trabajo con el lenguaje; además por ser multiplataforma, de código libre. Es independiente de la plataforma puesto que existe una máquina virtual de Java que facilita la comunicación entre el sistema operativo y el programa.

1.3.3 Lenguaje de Modelado. UML (*Unified Modeling Language*).

Se centra en la representación gráfica de un sistema; indica cómo crear y leer los modelos. UML resuelve de forma bastante satisfactoria un viejo problema del desarrollo de software como es su modelado gráfico. [9]

Sus principales ventajas son: [10]

- Se puede usar para modelar distintos tipos de sistemas: sistemas de software, sistemas de hardware, y organizaciones del mundo real. UML ofrece nueve diagramas en los cuales modelar sistemas.
- Es una consolidación de muchas de las notaciones y conceptos más usadas.

1.3.4 Frameworks a utilizar.

1.3.4.1 Spring Framework.

Spring es un framework de código abierto, creado para hacer frente a la complejidad del desarrollo de aplicaciones empresariales. La utilidad de Spring no se limita al desarrollo del lado del servidor. Cualquier

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

Aplicación Java se puede beneficiar de Spring en términos de simplicidad, la capacidad de prueba, y acoplamiento. [11]

Algunas características de Spring:[11]

- Ligerio: es de peso ligero en términos de tamaño. La mayor parte de Spring Framework se puede distribuir en un único archivo .jar que pesa poco más de 2,5MB.
- Orientado a aspectos: ofrece soporte para programación AOP (*Orientada a Aspectos*) que permite el desarrollo cohesionado por la separación de la aplicación, la lógica de negocio de los servicios del sistema (tales como la auditoría y la gestión de transacciones).
- Configurable: permite configurar y componer complejas solicitudes de los componentes más simples. Los objetos de aplicación son compuestos de forma declarativa, por lo general en un archivo XML (*Extensible Markup Language*).
- Flexible: diseñado como una serie de módulos que pueden trabajar independientemente uno de otro. Además intenta mantener un mínimo acoplamiento entre la aplicación y el propio Framework de forma que podría ser desvinculada de él sin demasiada dificultad. [12]

Spring proporciona la posibilidad de integrarse con otras herramientas incluyendo otros frameworks. Cuenta con interfaces en su diseño promoviendo la reutilización del código. Da la posibilidad de desarrollar más rápido, de forma más organizada y más eficiente, es un Framework maduro con varios años en el mercado. Provee soluciones muy bien documentadas y fáciles de usar, por lo que se seleccionó como Framework a utilizar.

1.3.4.2 Spring Security

Spring Security es un Framework escrito en Java que provee autenticación avanzada, autorización y otros elementos de seguridad para aplicaciones empresariales usando Spring. [11]

Para asegurar aplicaciones web, Spring Security usa filtros de servlets, dichos filtros interceptan las peticiones hechas a los servlets para realizar la autenticación e implementar la seguridad. Spring Security también puede implementar seguridad a un nivel inferior mediante el aseguramiento de invocaciones de métodos. Para asegurar métodos Spring Security usa Spring AOP, el cual aplica aspectos que aseguran que el usuario tiene los permisos requeridos para invocar los métodos seguros. [11]

Spring Security está conformado por cinco componentes claves, los cuales se muestran en la siguiente imagen: [11]

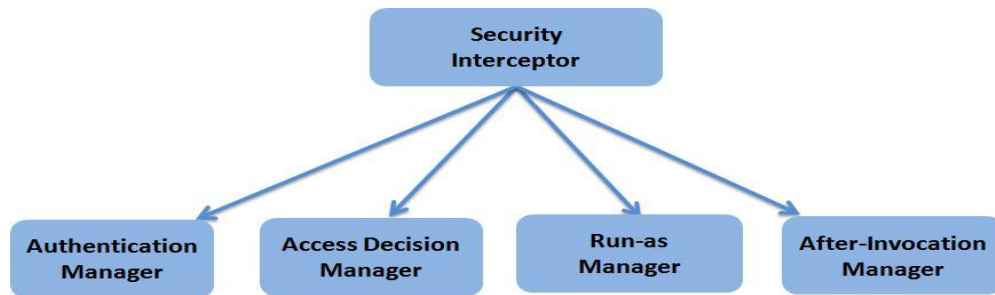


Figura 3: Elementos fundamentales de Spring Security.

- Security Interceptor realmente no aplica reglas de seguridad. Intercepta el acceso a los recursos para implementar la seguridad y delega la autoridad de administrarla a los demás managers.
- Authentication Managers responsable de determinar quién es quién. Realiza esto al considerar el principal (generalmente un usuario) y las credenciales (generalmente una contraseña).
- Una vez que se ha determinado quién es quién, entra en funcionamiento el Access Decision Manager. Este es el que realiza el proceso de autorización; decide si se puede acceder a un recurso, en dependencia de la información de la autenticación y de los atributos de seguridad asociados a cada recurso.
- Cuando hay recursos con diferentes requerimientos de seguridad y, una vez autenticado y garantizado el acceso, se usa Run-as Manager para reemplazar la autenticación existente por una que da completo acceso a estos recursos.
- After-Invocation Manager el componente que asegura de que el usuario que ha realizado la petición tenga permitido ver los datos que están siendo retornados de un recurso con seguridad.

1.3.4.3 Dojo Framework

Dojo es un framework de código abierto que facilita el desarrollo de aplicaciones Web. Resuelve asuntos de usabilidad comunes como pueden ser la navegación y detección del navegador, soportar cambios de URL(*Uniform Resource Locator*) en la barra de URLs para luego regresar a ellas, y la habilidad de degradar cuando Java Script no es completamente soportado en el cliente. [13]

La selección de Spring Framework, Spring Security y Dojo Framework estuvo condicionada principalmente por la experiencia con la que cuenta el equipo de desarrollo en el trabajo con estos frameworks.

1.3.5 JDBC (*Java Database Connectivity*)

JDBC es una API de Java para ejecutar sentencias SQL (*Structured Query Language*). Consta de un conjunto de clases e interfaces escrito en lenguaje de programación Java. Permite enviar sentencias SQL virtualmente a cualquier base de datos relacional. Se puede escribir un solo programa usando la API JDBC y será capaz de enviar sentencias SQL a la base de datos apropiada. JDBC extiende lo que puede hacerse con Java. Posibilita establecer una conexión con una base de datos; enviar sentencias SQL y procesar los resultados. [14]

Dentro de los objetivos de la filosofía JDBC se encuentran: [11]

- Proveer una interfaz homogénea al resto de APIs de Java.
- Ser un API simple, y desde ahí, ir creciendo.
- Mantener los casos comunes de acceso a Base de Datos lo más sencillo posible.
 - Conservar la sencillez en los casos más comunes (SELECT, INSERT, DELETE y UPDATE).
 - Hacer realizables los casos menos comunes: Invocación de procedimientos almacenados, entre otros.
- JDBC prefiere incluir gran cantidad de métodos, en lugar de hacer métodos complejos con gran cantidad de parámetros.

Teniendo en cuenta que el sistema se desarrollará en lenguaje Java y que la tecnología JDBC es el mecanismo por excelencia de acceso a datos dentro de este lenguaje se decide utilizar esta API para el acceso a los datos.

1.3.6 Socket

“Un socket (enchufe), es un método para la comunicación entre un programa cliente y un programa servidor en una red.” [15]

Un socket queda definido por un par de direcciones IP (*Internet Protocol*) local y remota, un protocolo de transporte y un par de números de puerto local y remoto. Para que dos programas puedan comunicarse entre sí es necesario que se cumplan ciertos requisitos: [15]

- Que un programa sea capaz de localizar al otro.
- Que ambos programas sean capaces de intercambiarse cualquier secuencia de octetos, es decir, datos relevantes a su finalidad.

Para ello son necesarios los tres recursos que originan el concepto de socket: [15]

- Un protocolo de comunicaciones, que permite el intercambio de octetos.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

- Un par de direcciones del Protocolo de Red (Dirección IP, si se utiliza el Protocolo TCP/IP (*Transmission Control Protocol/ Internet Protocol*)), que identifica la computadora de origen y la remota.
- Un par de números de puerto, que identifica a un programa dentro de cada computadora.

“Los sockets permiten implementar una arquitectura cliente-servidor. La comunicación debe ser iniciada por uno de los programas que se denomina programa cliente. El segundo programa espera a que otro inicie la comunicación, por motivo se denomina programa servidor.” [15]

Un socket es un proceso o hilo presente en la máquina cliente y en la máquina servidora, que sirve para que el programa servidor y el cliente lean y escriban la información. Esta información será la transmitida por las diferentes capas de red. [15]

“Cuando un cliente conecta con el servidor se crea un nuevo socket, de esta forma, el servidor puede seguir esperando conexiones en el socket principal y comunicarse con el cliente conectado, de igual manera se establece un socket en el cliente en un puerto local.” [15]

“Una aplicación servidor normalmente escucha por un puerto específico esperando una petición de conexión de un cliente, una vez que se recibe, el cliente y el servidor se conectan de forma que les sea posible comunicarse entre ambos. Durante el proceso, el cliente es asignado a un número de puerto, mediante el cual envía peticiones al servidor y recibe de este las respuestas correspondientes.” [15]

“Similarmente, el servidor obtiene un nuevo número de puerto local que le servirá para poder continuar escuchando cada petición de conexión del puerto original. De igual forma une un socket a este puerto local.

La principal ventaja de los sockets radica en que son muy eficientes a la hora de enviar un número elevado de mensajes y datos.” [15]

“Las propiedades de un socket dependen de las características del protocolo en el que se implementan. Generalmente la comunicación con sockets se realiza mediante un protocolo de la familia TCP/IP. Los dos más utilizados son: TCP y UDP (User Datagram Protocol).” [15]

Cuando se implementan con el protocolo TCP, los sockets tienen las siguientes propiedades: [15]

- Orientado a conexión.
- Se garantiza la transmisión de todos los octetos sin errores ni omisiones.
- Se garantiza que todo octeto llegará a su destino en el mismo orden en que se ha transmitido. Estas propiedades son muy importantes para garantizar la corrección de los programas que tratan la información.

El protocolo UDP es un protocolo no orientado a la conexión. Sólo se garantiza que si un mensaje llega, llegue bien. En ningún caso se garantiza que llegue o que lleguen todos los mensajes en el mismo orden que se mandaron. [15]

1.3.7 API para Bluetooth. BlueCove 2.1.0

BlueCove es una implementación JSR-82 (*Java Specification Request*) de J2SE (*Java Standard Edition*). Originalmente desarrollado por Intel Research. BlueCove se ejecuta en cualquier JVM (*Java Virtual Machine*) a partir de la versión 1.1 en los sistemas operativos Windows Mobile, Windows XP y Windows Vista, Mac OS. Desde la versión 2.1 BlueCove distribuye bajo la Apache Software License, versión 2.0. [16]

1.4 Metodología de Desarrollo

Las Metodologías de Desarrollo de Software surgen por la necesidad de utilizar una serie de procedimientos, técnicas, herramientas y soporte documental para desarrollar un producto de software. Dichas metodologías guían a los desarrolladores al crear un nuevo software, los requisitos de un software a otro son tan variados y cambiantes, que ha dado lugar a que exista una gran variedad de metodologías para la creación del software. [17]

1.4.1 FDD (*Feature Driven Development*)

Se considera que FDD se encuentra a medio camino entre las metodologías RUP (*Rational Unified Process*) y XP (*eXtreme Programming*). Está pensado para proyectos con tiempos de desarrollo menor de un año. Se basa en un proceso iterativo con iteraciones cortas aproximadamente de dos semanas que producen un software funcional que el cliente y la dirección de la empresa pueden ver y monitorizar. Las iteraciones se deciden en base a funcionalidades, que son pequeñas partes del software con significado para el cliente. [18]

Un proyecto que usa la metodología FDD se divide en cinco fases: [18]

- Desarrollo de un modelo general.
- Construcción de la lista de funcionalidades.
- Plan de versiones en base a las funcionalidades a implementar.
- Diseñar en base a las funcionalidades.
- Implementar en base a las funcionalidades.

Las primeras tres fases ocupan gran parte del tiempo en las primeras iteraciones, siendo las dos últimas las que absorben la mayor parte del tiempo según va avanzando el proyecto. [18]

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

El trabajo se realiza en grupo, aunque siempre habrá un responsable final con mayor experiencia que tendrá la última palabra en caso de no llegar a un acuerdo. Al hacerlo en grupo se consigue que todos formen parte del proyecto y que los menos expertos aprendan de las discusiones de los más experimentados. [18]

Se decide usar esta metodología por implementarse mejor para proyectos cortos y equipos pequeños además, el equipo de desarrollo cuenta con experiencia en el trabajo con la misma. Genera una cantidad de documentación aceptable, dando la posibilidad de realizar nuevas versiones del mismo. Le da libertad a los desarrolladores y no se basa en formalismos en la documentación, sino en controles propios y una comunicación fluida con el cliente.

1.5 Herramientas

Para lograr una calidad satisfactoria y un mejor aprovechamiento del tiempo empleado es necesario hacer una correcta selección de las herramientas a utilizar durante el proceso de desarrollo del software.

1.5.1 IDE (*Entorno Integrado de Desarrollo*). Eclipse 3.5.0

Es un entorno de desarrollo integrado, de código abierto y multiplataforma. Es una potente y completa plataforma de programación, desarrollo y compilación de elementos tan variados como sitios web. Dispone de un editor de texto con resaltado de sintaxis donde se puede ver el contenido del fichero en el que se está trabajando, la compilación es en tiempo real. [19]

En cuanto a la utilización de Eclipse para la creación de aplicaciones clientes se puede decir que provee al programador con frameworks muy ricos para el desarrollo de aplicaciones gráficas, definición y manipulación de modelos de software y aplicaciones web. Incluye las herramientas de desarrollo de Java, ofreciendo un IDE con un compilador de Java interno y un modelo completo de los archivos fuente de Java. Esto permite técnicas avanzadas de refactorización y análisis de código. Hace uso de un espacio de trabajo permitiendo modificaciones externas a los archivos en tanto se refresque el espacio de trabajo correspondiente. [19]

Se decide utilizar Eclipse como entorno de desarrollo por ser una potente herramienta de programación, de código abierto, multiplataforma, poco consumo de memoria y además soportar el lenguaje de programación Java. El equipo de desarrollo cuenta con experiencia de trabajo con Eclipse.

1.5.2 Servidor Web. Apache Tomcat 6.0.26

Apache Tomcat es una implementación de software de código abierto de la especificación Java Servlet y tecnologías Java Server Pages. Es desarrollado en un entorno abierto y participativo, publicado bajo la licencia Apache versión 2. [20]

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

Es un servidor web ya que gestiona solicitudes y respuestas HTTP (*Hyper Text Transfer Protocol*); además es servidor de aplicaciones o contenedor de Servlets/JSP (*Java Server Pages*). [21]

Se selecciona el servidor web Apache Tomcat por la necesidad de usar aplicaciones distribuidas en el sistema a realizar ya que permite el trabajo con Servlets, es fácil de configurar, fiable, puede funcionar como servidor web por sí mismo. Funciona en todos los sistemas operativos que dispongan de la máquina virtual de Java.

1.5.3 Sistema Gestor de Base de Datos. PostgreSQL 8.4.0

PostgreSQL es un sistema de gestión de bases de datos basado en código abierto. Es un sistema objeto-relacional, ya que incluye características de la orientación a objetos, como puede ser la herencia, tipos de datos, funciones, restricciones, disparadores, reglas e integridad transaccional. A pesar de esto, no es un sistema de gestión de bases de datos puramente orientado a objetos. Algunas de las características principales del gestor de bases de datos son: [22]

- Soporta distintos tipos de datos: además del soporte para los tipos base, también soporta datos de tipo fecha, monetarios, elementos gráficos, datos sobre redes, cadenas de bits, entre otros. También permite la creación de tipos propios.
- Permite la declaración de funciones propias, así como la definición de disparadores.
- Soporta el uso de índices, reglas y vistas.
- Incluye herencia entre tablas (aunque no entre objetos, ya que no existen), por lo que a este gestor de bases de datos se le incluye entre los gestores objeto-relacionales.
- Permite la gestión de diferentes usuarios, como también los permisos asignados a cada uno de ellos.

Entre las ventajas que posee PostgreSQL se encuentran las que siguen: [22]

- Posee una gran escalabilidad: es capaz de ajustarse al número de CPUs y a la cantidad de memoria que posee el sistema de forma óptima, haciéndole capaz de soportar una mayor cantidad de peticiones simultáneas de manera correcta.
- Implementa el uso de retrocesos, subconsultas y transacciones, haciendo su funcionamiento mucho más eficaz.
- Tiene la capacidad de comprobar la integridad referencial, así como también la de almacenar procedimientos en la propia base de datos.

1.5.4 Herramienta de Modelado. Visual Paradigm 8.0

Las Herramientas CASE (*Computer Aided Assisted Automated Software Systems Engineering*) se definen como un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

desarrolladores, durante el ciclo de vida del desarrollo de un software; fueron desarrolladas para automatizar los procesos y facilitar las tareas de coordinación de los eventos que necesitan ser mejorados. La principal ventaja de la utilización de una herramienta CASE, es la mejora de la calidad de los desarrollos realizados y, en segundo término, el aumento de la productividad. [23]

Visual Paradigm es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software; ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. [24]

Algunas de las características son: [25]

- Entorno de creación de diagramas para UML 2.1.
- Diseño centrado en casos de uso y enfocado al negocio que genera un software de mayor calidad.
- Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- Capacidades de ingeniería directa e inversa.
- Modelo y código que permanece sincronizado en todo el ciclo de desarrollo.
- Disponibilidad de integrarse en los principales IDEs.
- Disponibilidad en múltiples plataformas.

El equipo de desarrollo cuenta con experiencia en el trabajo con esta herramienta, por lo que se seleccionó como herramienta de modelado.

1.5.5 JfreeChart

JFreeChart es una librería de código abierto 100% para la generación de gráficos escrita en Java que hace que sea fácil para los desarrolladores mostrar gráficos de calidad profesional en sus aplicaciones. [26]

Esta herramienta será utilizada para diseñar visualmente los reportes que deberá generar el sistema. Además vinculados con la librería de Java JasperReport proporciona un potente entorno para desarrollar el sistema.

1.6 Conclusiones Parciales

En este capítulo se realizó un estudio de las aplicaciones de envío de publicidad existentes en el mundo. Se definieron las tecnologías, herramientas y metodología a utilizar durante el proceso de desarrollo del software.

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

2.1 Introducción

Para el desarrollo del capítulo se realizará una propuesta del sistema además de especificar los requisitos funcionales y los requisitos no funcionales que debe cumplir el sistema. Se muestra el modelo de dominio y se definen los casos de uso definiendo los actores y se muestra el diagrama de casos de uso del sistema.

2.2 Problemática

La publicidad es un área muy importante para muchas empresas, centros comerciales y entidades de venta. En Cuba, los métodos utilizados para promover sus productos no llegan a la mayoría de los usuarios potenciales. Dado el avance de las tecnologías de la información y las telecomunicaciones, el uso de dispositivos móviles se ha incrementado en gran medida, por lo que se puede hacer llegar estos anuncios a una mayor cantidad de clientes. En el proceso de gestión de la publicidad existen problemas que afectan el correcto funcionamiento de este. No existe un sistema que haga llegar a los clientes las nuevas ofertas, por lo que estos en pocas ocasiones tienen conocimiento de las ofertas que en las entidades de venta se realizan. La mayor parte de los usuarios potenciales no tiene acceso a las tecnologías debido a la infraestructura con que cuenta el país, la cual es muy atrasada e insuficiente y se utiliza para los servicios públicos como la salud y la educación. Se desconoce el nivel de impacto de determinadas publicidades en las entidades de venta por lo que no se tiene conocimiento de que aceptación tuvo un determinado anuncio entre los usuarios de estas.

Por todo lo antes expuesto se hace necesario implementar nuevas técnicas automatizadas que gestionen el proceso de envío de publicidad utilizando la tecnología Bluetooth hacia los dispositivos móviles en varias localidades de un mismo establecimiento.

2.3 Objeto de automatización

Se desea automatizar el envío de publicidad a los usuarios de centros comerciales y entidades de venta. Además serán objeto de automatización los procesos de crear, modificar y eliminar publicidad. El sistema brindará la posibilidad de generar reportes que mostrarán el comportamiento de la publicidad que se acepta o se rechaza.

2.4 Propuesta del sistema

Se propone diseñar un sistema que permita interactuar con un servicio de envío de publicidad. El sistema debe estar instalado en servidores que cuenten con tecnología Bluetooth donde se ejecutará el servicio de envío de

publicidad, ubicadas en las diferentes áreas donde se desee ofrecer publicidad. Estos equipos enviarán información todo el tiempo siendo detenidas solo por el administrador del sistema. Los servidores se actualizarán a través de una aplicación cliente donde los agentes publicitarios podrán añadir, registrar y cargar los ficheros que se desean enviar. Deberá quedar registrado en una base de datos que usuario añadió la publicidad y en qué momento.

El administrador del sistema será el encargado de ejecutar, reiniciar, actualizar y detener los servidores en un momento determinado. El envío desde un servidor puede ser detenido independientemente sin afectar al resto de los equipos. Podrá el administrador generar gráficas estadísticas que muestren el comportamiento de los envíos publicitarios por cada departamento, por productos y por publicidades aceptadas por los clientes.

En la siguiente figura se muestra una descripción visual de la propuesta del sistema:

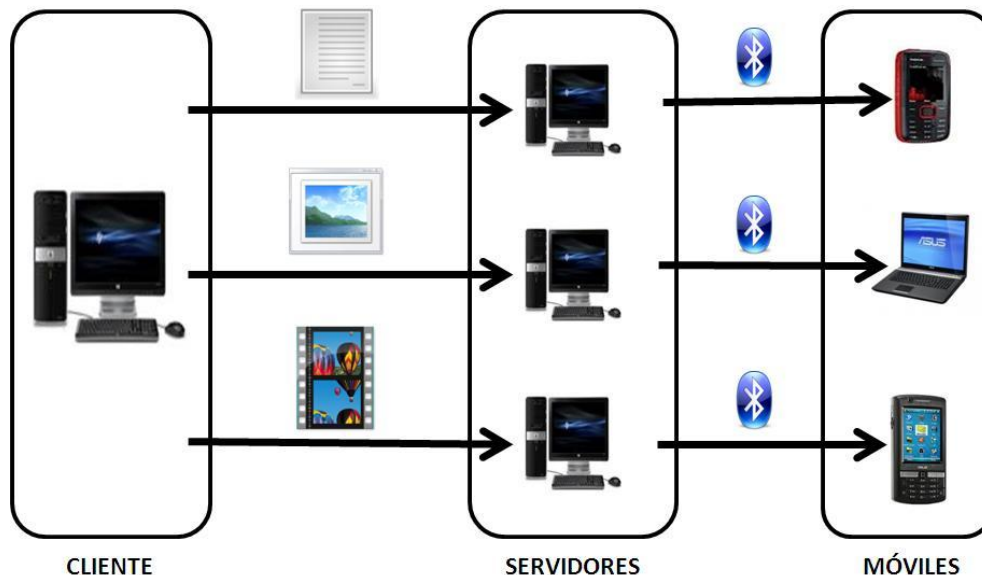


Figura 4: Propuesta del Sistema.

2.5 Modelo de Dominio

Para desarrollar cualquier sistema, independientemente de su complejidad, es necesario dividirlo en secciones, estas se pueden representar mediante modelos que permitan abstraer sus características esenciales. Como paso previo al diseño del sistema es preciso realizar un Modelo de Negocio o Modelo de Dominio para capturar y enunciar la visión del proceso, encaminado a satisfacer las necesidades durante estas fases.

Para la construcción del sistema no se realizó un modelo del negocio ya que no son claramente visibles los procesos del negocio, por tanto se realizó un modelo de dominio, el cual se muestra a continuación:

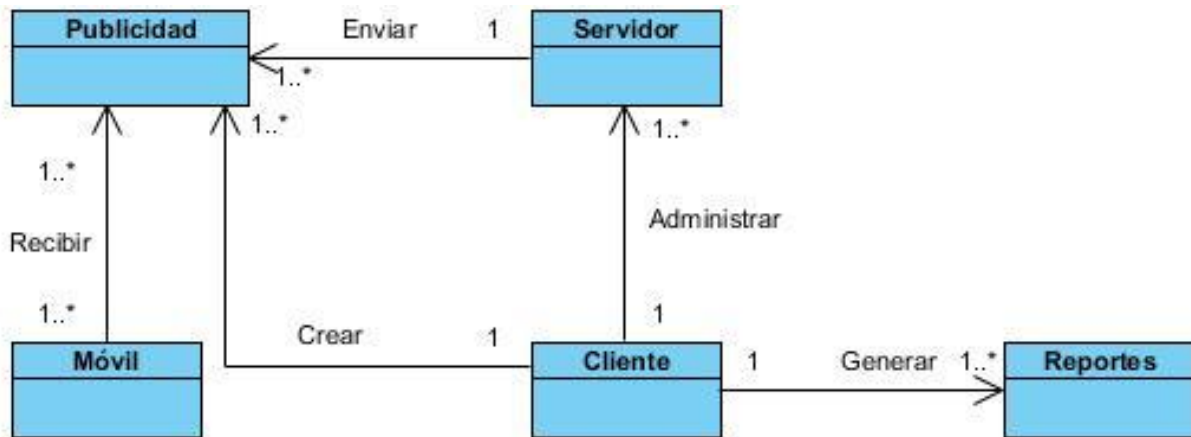


Figura 5: Modelo de Dominio.

Descripción de los objetos del dominio:

- **Servidor:** Sistema que se encarga de enviar la publicidad a los dispositivos móviles.
- **Cliente:** Aplicación que se encarga de generar los reportes, crear la publicidad y cargarla en los servidores.
- **Publicidad:** representa la publicidad.
- **Móvil:** recibe las publicidades enviadas por los servidores.
- **Reportes:** representa los datos de los reportes generados por el cliente.

2.6 Relación de los requerimientos

Los requerimientos para un sistema de software determinan lo que hará el sistema y definen las restricciones de su operación e implementación.

2.6.1 Requerimientos funcionales

Los requerimientos funcionales de un sistema son las capacidades o condiciones con que debe cumplir el mismo.

RF 1: Autenticar Usuario.

- Usuario
- Contraseña

RF 2: Gestionar Publicidad.

2.1 Añadir publicidad.

- Nombre
- Descripción

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

- Fichero

2.2 Eliminar publicidad.

- Nombre
- Fecha

2.3 Consultar horario de la publicidad enviada.

- Fecha Inicio
- Fecha Fin

RF 3: Enviar Publicidad.

RF 4: Controlar Envío de Publicidad.

4.1 Ejecutar aplicación.

4.2 Detener aplicación.

RF 5: Comprobar Servidores

RF 6: Gestionar Usuarios.

6.1 Añadir usuario.

- Nombre
- Primer Apellido
- Segundo Apellido
- Rol
- Nombre de Usuario
- Contraseña

6.2 Modificar usuario.

Para realizar la búsqueda del usuario:

- Nombre
- Rol
- Nombre de usuario

Para modificar el usuario:

- Nombre
- Primer Apellido
- Segundo Apellido
- Rol
- Nombre de Usuario

- Contraseña

6.3 Eliminar usuario.

- Nombre
- Rol
- Nombre de usuario

RF 7: Gestionar Servidores.

7.1 Añadir Servidor.

- Dirección IP
- Departamento
- Puerto

7.2 Modificar Servidor.

Para buscar el servidor:

- Dirección IP
- Departamento

Para modificar el servidor:

- Dirección IP
- Departamento
- Puerto

7.3 Eliminar Servidor.

- Dirección IP
- Departamento

RF 8: Generar Reportes.

- Fecha Inicio
- Fecha Fin

2.6.2 Requerimientos no funcionales

Los requerimientos no funcionales de un sistema son las cualidades que debe tener el producto para que sea más atractivo al cliente, rápido, usable y confiable, lo que puede hacer diferencia entre otros productos del mismo tipo.

Los requerimientos no funcionales del sistema son:

RNF 1: Portabilidad.

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

El sistema deberá ser multiplataforma.

RNF 2: Software.

- Cliente: requiere del servidor Apache Tomcat instalado. Es necesaria la instalación y configuración del servidor de base de datos PostgreSQL.
- Servidores: deben contar con la Máquina Virtual de Java versión 1.6 instalada.
- Sistema Operativo: podrá instalarse en sistemas operativos tales como Windows XP, Windows 7, Ubuntu.

RNF 3: Usabilidad.

- La solución informática a desarrollar deberá ser fácil de usar por los usuarios del sistema, lo que quiere decir que no es necesario para el usuario tener conocimientos informáticos avanzados para operar el sistema.
- Las imágenes y videos que podrá gestionar el sistema de envío de publicidad deben tener un tamaño menor a 1 Mb.

RNF 4: Seguridad.

El sistema debe proteger la integridad de la información y los contenidos. El acceso a cualquier acción del sistema, debe estar sometido a un proceso de autenticación del usuario donde serán especificadas las funcionalidades a las cuales tiene acceso, usuario y contraseña. A los usuarios autorizados se les garantizará el acceso a la información y los dispositivos o mecanismos utilizados para lograr la seguridad no ocultarán o retrasarán a los usuarios para obtener los datos deseados en un momento dado.

RNF 5: Hardware.

- El sistema podrá intercambiar archivos con dispositivos móviles compatibles con el servicio OBEX (*Object Exchange*) de transferencia de archivos vía Bluetooth.
- Las PC Servidores deben contar con un dispositivo Bluetooth incorporado para realizar los envíos de publicidad.
- Debe existir una conexión de red basada en TCP/IP para establecer la conexión entre los clientes y los servidores.

2.7 Definición de los casos de uso

2.7.1 Definición de los actores del sistema

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

Actores	Justificación
Usuario del Sistema	Es el encargado de añadir, registrar y cargar los ficheros que se desean enviar. Además podrá consultar el horario de la publicidad enviada.
Administrador del Sistema	El administrador del sistema será el encargado de controlar el envío de publicidad, deteniendo o ejecutando el mismo. Podrá generar gráficas estadísticas que muestren el comportamiento de los envíos publicitarios por departamentos, por productos y por publicidades aceptadas por los clientes. Tendrá a su cargo la gestión de los usuarios y podrá consultar el horario de la publicidad enviada. Además de poder gestionar la publicidad.
Reloj	Será el encargado de realizar los envíos de la publicidad además de comprobar el correcto funcionamiento de los servidores.

Tabla 1: Definición de los actores.

2.7.2 Diagrama de casos de uso

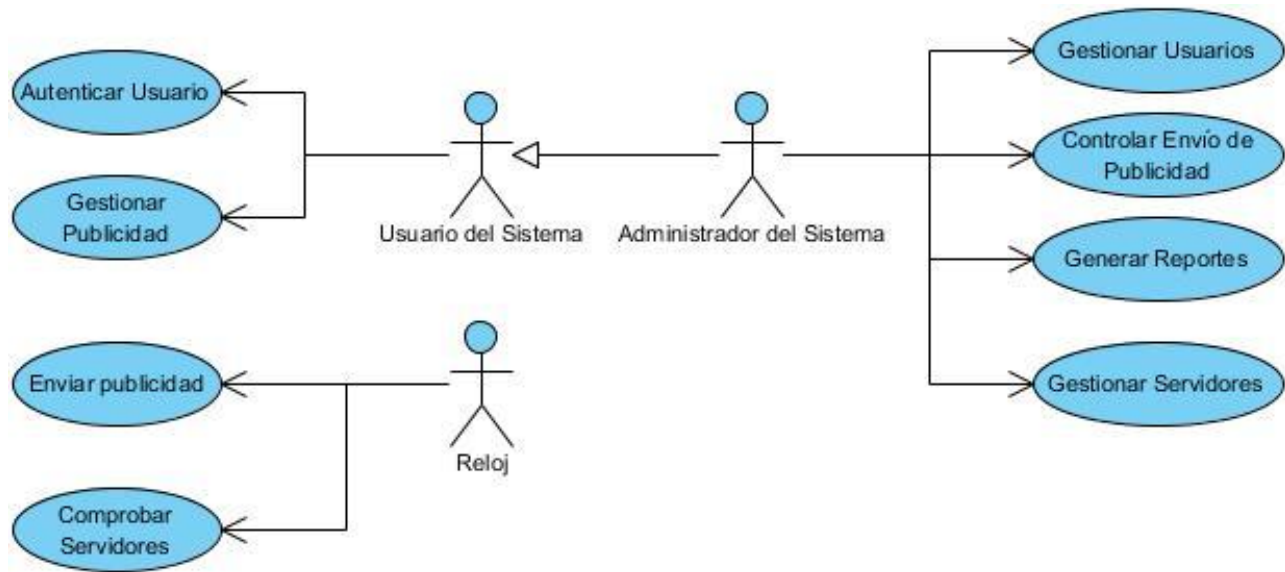


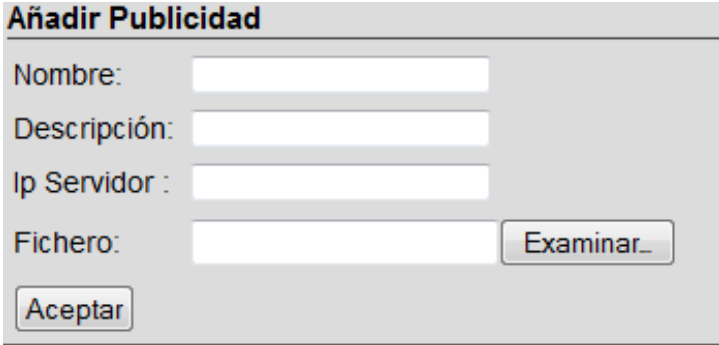
Figura 6: Diagrama de Casos de Uso del Sistema.

2.7.3 Descripción de casos de uso

Para especificar cada una de las funcionalidades que deben ser implementadas se describen los casos de uso del sistema. A continuación se muestra la descripción de algunos casos de uso.

Caso de Uso	
CU_2	Gestionar Publicidad
Propósito:	Añadir, Eliminar y Mostrar Datos de la publicidad.
Actores:	Usuario del Sistema
Prioridad:	Crítico.
Resumen:	El caso de uso se inicia cuando el Usuario del Sistema necesita insertar, eliminar o consultar los datos de una publicidad. Permite introducir en el sistema los datos de la publicidad, así como eliminarla y consultar el horario de envío de una determinada publicidad.

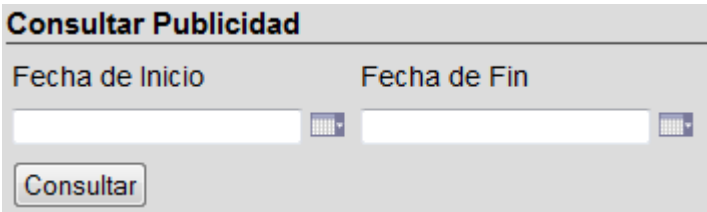
CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

Precondiciones:	El usuario debe estar autenticado en el sistema.	
Referencias:	RF 2	
Acción del Actor	Respuesta del Sistema	
1. Selecciona la opción "Publicidad".	2. Muestra las siguientes opciones: <ul style="list-style-type: none"> ✓ Añadir publicidad. ✓ Eliminar publicidad. ✓ Consultar Horario de Publicidad Enviada. 	
3. Selecciona una de las siguientes opciones: <ul style="list-style-type: none"> ✓ Añadir publicidad(Ver Sección: "Añadir publicidad") ✓ Eliminar publicidad (Ver Sección: "Eliminar publicidad") ✓ Consultar Horario de Publicidad Enviada(Ver Sección: "Consultar Horario de Publicidad Enviada") 		
Sección: "Añadir publicidad"		
Prototipo:		
		
	4. Muestra la interfaz para añadir la publicidad solicitando los siguientes datos: <ul style="list-style-type: none"> ✓ Nombre de Publicidad. ✓ IP Servidor. 	

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

	<ul style="list-style-type: none"> ✓ Fichero a Cargar. ✓ Descripción de la publicidad. 										
5. Introduce los datos solicitados.											
6. Presiona el botón “Aceptar”.	7. Valida que los datos introducidos son correctos.										
	8. Valida que los campos obligatorios no están vacíos.										
	9. Añade la publicidad.										
Flujos Alternos											
Flujo Alterno al Paso 7 “Datos incorrectos”											
	7.1. Muestra un mensaje indicándole al usuario que hay datos incorrectos.										
Flujo Alterno al Paso 8 “Campos vacíos ”											
	8.1. Muestra un mensaje indicándole al usuario que hay campos vacíos.										
Sección: “Eliminar publicidad”											
Prototipo:											
<div style="border: 1px solid gray; padding: 5px;"> <p>Eliminar Publicidad</p> <p>Nombre: <input type="text"/></p> <p>Fecha <input type="text"/> </p> <p><input type="button" value="Buscar"/></p> <hr/> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 5%;"></th> <th style="width: 20%;">Nombre</th> <th style="width: 40%;">Descripción</th> <th style="width: 20%;">Ip Servidor</th> <th style="width: 15%;">Fecha</th> </tr> </thead> <tbody> <tr> <td><input checked="" type="checkbox"/></td> <td>Cremas</td> <td>Cremas de piel</td> <td>10.51.17.19</td> <td>2012-05-02</td> </tr> </tbody> </table> <p><input type="button" value="Eliminar"/></p> </div>			Nombre	Descripción	Ip Servidor	Fecha	<input checked="" type="checkbox"/>	Cremas	Cremas de piel	10.51.17.19	2012-05-02
	Nombre	Descripción	Ip Servidor	Fecha							
<input checked="" type="checkbox"/>	Cremas	Cremas de piel	10.51.17.19	2012-05-02							
	4. Muestra una interfaz para realizar una búsqueda de la publicidad que se desea eliminar de las publicidades a enviar con los siguientes datos: <ul style="list-style-type: none"> ✓ Nombre de Publicidad. ✓ Fecha. 										
5. Introduce los datos requeridos para la búsqueda y presiona el botón “Buscar”.	6. Muestra una interfaz con los archivos encontrados										

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

	que cumplen con el criterio de búsqueda.
7. Selecciona la publicidad que desea eliminar de las publicidades a enviar y presiona el botón “Eliminar”.	8. Muestra un mensaje de confirmación.
9. Presiona el botón “Aceptar”.	10. Guarda la publicidad seleccionada en el historial de publicidad.
	11. Elimina la publicidad seleccionada de las publicidades a enviar.
	12. Notifica al servidor para que elimine la publicidad de si lista de publicidades a enviar.
Flujos Alternos	
Flujo Alterno al Paso 9 “Cancelar Operación”	
9.1. Presiona el botón “Cancelar”.	9.2. Cancela la operación y redirecciona a la interfaz de inicio.
Sección: “Consultar Horario de la Publicidad Enviada”	
Prototipo:	
	
	4. Muestra la interfaz para consultar el horario de la publicidad enviada con los siguientes campos: <ul style="list-style-type: none"> ✓ Fecha de Inicio. ✓ Fecha de Fin.
5. Introduce los datos requeridos y presiona el botón “Consultar”.	6. Valida que no hayan campos vacíos.
	7. Muestra una interfaz con el listado de publicidades enviadas que se encuentran en el período de tiempo especificado.
Flujos Alternos	

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

Flujo Alternativo al Paso 7 "Campos vacíos"	
	6.1. Muestra un mensaje indicándole al usuario que debe llenar todos los campos.
Poscondiciones:	

Tabla 2: Descripción del Caso de Uso Gestionar Publicidad.

Caso de Uso	
CU_3	Enviar Publicidad.
Propósito:	Realizar el envío de publicidad a los clientes de las entidades de venta.
Actores:	Reloj.
Prioridad:	Crítico.
Resumen:	El caso de uso se inicia cuando se decide enviar la publicidad. Permite realizar el envío de publicidad a los clientes de las entidades de venta.
Precondiciones:	La publicidad debe estar añadida a la base de datos.
Prototipo:	
Referencias:	RF 3
Acción del Actor	Respuesta del Sistema
1. Se cumple el tiempo de espera para iniciar el envío.	2. Cargar las publicidades a enviar.
	3. Buscar los dispositivos móviles dentro del área de cobertura.
	4. Verifica que no se le haya enviado la publicidad en un tiempo menor al especificado para volver a

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

	enviar a un mismo dispositivo.
	5. Establecer conexión con el dispositivo móvil.
	6. Enviar petición de autorización para iniciar la transferencia del archivo de publicidad.
	7. Esperar confirmación del dispositivo móvil.
	8. Comenzar transferencia de archivo de publicidad.
	9. Guardar los datos de la publicidad en el historial de publicidad.
	10. Chequea que no existieron errores en el envío de la publicidad.
Flujos Alternos	
Flujo Alterno al Paso 8 “Terminar conexión”	
	8.1. Guardar los datos de la publicidad en el historial de publicidad.
Flujo Alterno al Paso 10 “Registrar error”	
	10.1. Registra el error en el registro de errores.
	10.2. Notifica al administrador la ocurrencia del error.
Poscondiciones:	

Tabla 3: Descripción del Caso de Uso Enviar Publicidad.

2.8 Conclusiones Parciales

Durante el capítulo se expuso la descripción de la propuesta del sistema, se definieron los requisitos funcionales y los no funcionales, se mostró el modelo de dominio, el diagrama de casos de uso del sistema, la descripción de los actores y una descripción de dos de los casos de uso más significativos.

CAPÍTULO 3: DISEÑO DEL SISTEMA

3.1 Introducción

En el presente capítulo se describen los procesos que se llevan a cabo durante el diseño del sistema en base a los casos de uso, donde se representarán diagramas de paquetes, diagramas de clases del diseño, diagramas de secuencia y el diagrama entidad relación de la base de datos. También se muestra la arquitectura del sistema y los patrones de diseño.

3.2 Arquitectura

La arquitectura de software es, a grandes rasgos, una vista del sistema, constituye un puente entre el requerimiento y el código. Es la organización fundamental de un sistema encarnada en sus componentes, las relaciones entre ellos y el ambiente y los principios que orientan su diseño y evolución. [27]

3.2.1 Cliente-Servidor

Esta arquitectura se divide en dos partes claramente diferenciadas, la primera es la parte del servidor y la segunda la de un conjunto de clientes. Normalmente el servidor es una máquina que actúa de depósito de datos. Por otro lado los clientes suelen ser estaciones de trabajo que solicitan varios servicios al servidor. Ambas partes deben estar conectadas entre sí mediante una red. [28]

Para comprender con mayor claridad lo explicado anteriormente se muestra la siguiente imagen:

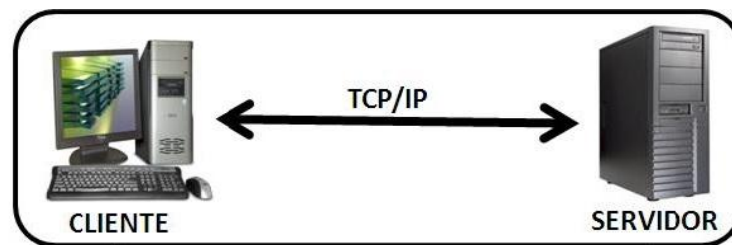


Figura 7: Arquitectura Cliente - Servidor.

Algunas características de los clientes: [29]

- Componente del sistema que interactúa con el usuario.
- No comparte sus recursos con otros clientes.
- No suelen tener restricciones especiales respecto a rendimiento, fiabilidad y escalabilidad.
- Debe dar soporte a restricciones relativas a facilidad de uso y evitar comprometer los demás componentes.

Características de los servidores: [29]

- Componente del sistema que presta servicios a los clientes.
- Gestiona y comparte sus recursos con los clientes a los que sirve.
- Suele tener restricciones especiales respecto a rendimiento, fiabilidad, escalabilidad y seguridad, entre las que se encuentran:
 - Capacidad suficiente para atender múltiples clientes.
 - Los fallos en el servidor son críticos e invalidan el sistema.
 - El número de clientes (peticiones) puede ser muy variable y aumentar si así se requiere.
 - Evitar comprometer la seguridad de los recursos o datos gestionados y de los clientes.

3.2.2 Patrón Arquitectónico N Capas

La lógica de aplicación se reparte en diferentes capas ubicadas entre el cliente y los datos. Las capas intermedias se proporcionan servicios entre sí (cada nivel se comunica sólo con los niveles contiguos a través de interfaces bien definidas). [29]

Algunas ventajas de este tipo de arquitectura son las siguientes: [29]

- Elementos críticos de la lógica de negocio ubicados en nivel medio (más cercanos a la capa de datos por lo que hay más eficiencia en el acceso).
- Mayor flexibilidad.
- Escalabilidad: facilita añadir recursos para soportar mayor número de clientes.
- Extensibilidad: facilita añadir nuevas funcionalidades al sistema sin afectar a los clientes existentes.
- Seguridad: facilidad para propagar autenticación y permisos a través de las distintas capas.
- Facilidades de desarrollo y administración:
 - Reusabilidad de componentes.
 - Aislamiento frente a cambios en otras capas.
 - Independencia frente a cambios en base de datos.

En el servidor se utilizará este patrón arquitectónico, específicamente dos (2) capas ya que esta aplicación no requiere de interfaz de usuario, por lo que tendrá una capa de negocio con las clases que manipulan la lógica del sistema y otra de acceso a datos con las clases que manipulan los datos, prescindiendo de la capa de presentación. La siguiente imagen muestra cómo queda adaptado el patrón al sistema:



Figura 8: Patrón N Capas (2 Capas).

3.2.3 Patrón Arquitectónico MVC (*Modelo Vista Controlador*)

La arquitectura de software Modelo Vista Controlador será utilizado en el cliente por las características que presenta el mismo. Separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos. La finalidad del modelo es mejorar la reusabilidad por medio del desacople entre la vista y el modelo.

A continuación se muestra una imagen del patrón Modelo Vista Controlador:

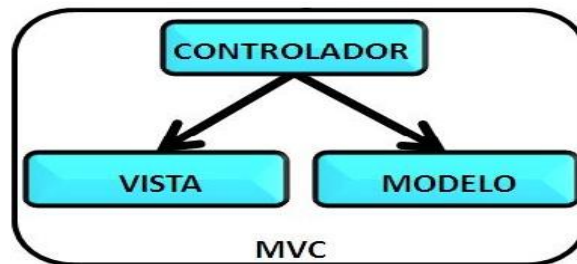


Figura 9: Patrón Modelo Vista Controlador.

Los elementos del patrón son los siguientes: [30]

El modelo:

- Accede a la capa de almacenamiento de datos.
- Define las reglas de negocio (la funcionalidad del sistema).
- Lleva un registro de las vistas y controladores del sistema.
- Un modelo activo, notificará a las vistas los cambios que en los datos pueda producir un agente externo.

El controlador:

- Recibe los eventos de entrada.
- Contiene reglas de gestión de eventos. Estas acciones pueden suponer peticiones al modelo o a las vistas.

Las vistas:

- Reciben datos del modelo y los muestran al usuario.
- Tienen un registro de su controlador asociado.

Algunas de las ventajas de utilizar el MVC son las siguientes: [31]

- Es posible tener diferentes vistas para un mismo modelo.
- Es posible construir nuevas vistas sin necesidad de modificar el modelo subyacente.
- Proporciona un mecanismo de configuración a componentes complejos mucho más tratable que el basado en eventos.

Luego de analizar los patrones arquitectónicos y la arquitectura a utilizar en el desarrollo del sistema se muestra a continuación la imagen que representa la arquitectura general y donde se utilizará cada uno de los patrones arquitectónicos:

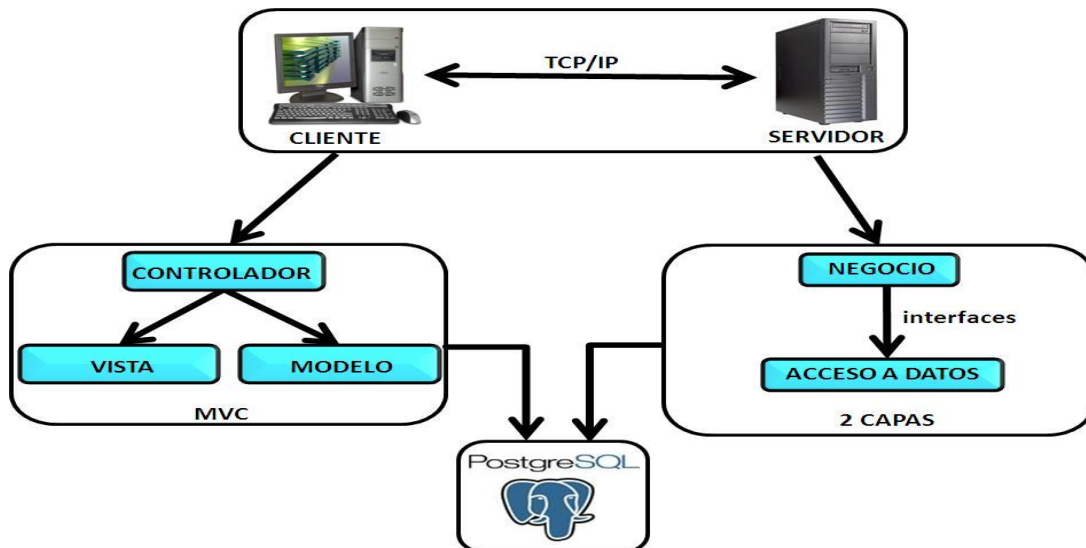


Figura 10: Arquitectura del Sistema.

3.3 Patrones de Diseño

Los patrones de diseño son descripciones de clases cuyas instancias colaboran entre sí. Cada patrón es adecuado para ser adaptado a un cierto tipo de problema.

3.3.1 Patrones GRASP (*General Responsibility Assignment Software Patterns*).

Los patrones GRASP son parejas de problema solución con un nombre, que codifican buenos principios y sugerencias relacionados frecuentemente con la asignación de responsabilidades. Describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones. [32]

3.3.1.1 Experto.

Para el desarrollo de un sistema se pueden definir muchas clases de software, y una aplicación tal vez requiera el cumplimiento de varias responsabilidades. Si estas se asignan en forma adecuada, los sistemas tienden a ser más fáciles de entender, mantener y ampliar, y da la oportunidad de reutilizar los componentes en futuras aplicaciones. Este patrón se usa para asignar responsabilidades a clases que cuentan con la información necesaria para cumplirlas. [32]

La siguiente figura muestra la utilización del patrón Experto en el desarrollo del sistema:

```
public class PublicidadDao extends SimpleJdbcDaoSupport implements IPublicidadDao
{
    public void insertarPublicidad(Publicidad publicidad) {}
    public void eliminarPublicidad(int idpublicidad) {}
    public List<Publicidad> obtenerPublicidadPorNombre(String nombre) {}
    public List<Publicidad> obtenerPublicidad(String nombre_publicidad, Date fecha) {}
    public List<Publicidad> listarPublicidad() {}
    public List<Publicidad> consultarPublicidad(Date fecha_inicio, Date fecha_fin) {}
    public List<Publicidad> obtener() {}
    public List<Publicidad> obtenerPorId(int id) {}
    private static class PublicidadMapper implements ParameterizedRowMapper<Publicidad> {}
}
```

Figura 11: Patrón Experto.

3.3.1.2 Creador.

La creación de objetos es una de las actividades más frecuentes en un sistema orientado a objetos. Puede soportar un bajo acoplamiento, una mayor claridad, el encapsulamiento y la reutilizabilidad. Guía la asignación de responsabilidades relacionadas con la creación de objetos. El propósito fundamental del patrón es encontrar un creador que debemos conectar con el objeto producido en cualquier evento. [32]

La siguiente figura muestra la utilización del patrón Creador en el desarrollo del sistema:

```
public class GestionarPublicidad implements IGestionarPublicidad
{
    private IPublicidadDao publicidaddao;
```

Figura 12: Patrón Creador.

3.3.1.3 Bajo Acoplamiento.

Significa que una clase no depende de muchas clases. Este tipo de patrones no se afectan por cambios de otros componentes, son fáciles de entender por separado y fáciles de reutilizar. [32]

3.3.1.4 Alta Cohesión.

Caracterizan a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme. Mejoran la claridad y facilidad con que se entiende el diseño, se simplifica el mantenimiento y las mejoras de funcionalidad. Genera un bajo acoplamiento y soporta mayor capacidad de reutilización. [32]

3.3.1.5 Controlador.

Es un objeto de interfaz no destinada al usuario que se encarga de manejar un evento del sistema. Define además el método de su operación. Poseen un mayor potencial de los componentes reutilizables. [32]

La siguiente figura muestra la utilización del patrón Controlador en el desarrollo del sistema:

```
public class AddPublicidadController extends SimpleFormController
{
    private IGestionarPublicidad gestionarpublicidad;
    public IGestionarPublicidad getGestionarpublicidad() {
    public void setGestionarpublicidad(IGestionarPublicidad gestionarpublicidad) {
    public AddPublicidadController()
    protected ModelAndView onSubmit(HttpServletRequest request,
}
```

Figura 13: Patrón Controlador.

3.3.2 Patrón Inversión de Control.

3.3.2.1 Patrón Inyección de Dependencia

La Inyección de Dependencia (en inglés Dependency Injection, DI) es un patrón de diseño orientado a objetos, en el que se inyectan objetos a una clase en lugar de ser la propia clase quien cree el objeto. Es no instanciar las dependencias explícitamente en su clase, sino declarativamente expresarlas en la definición de la clase. La esencia de la inyección de dependencias es contar con un componente capaz de obtener instancias válidas de las dependencias del objeto y pasárselas durante la creación o inicialización del objeto. [33]

La siguiente figura muestra la utilización del patrón Inyección de Dependencia en el desarrollo del sistema:

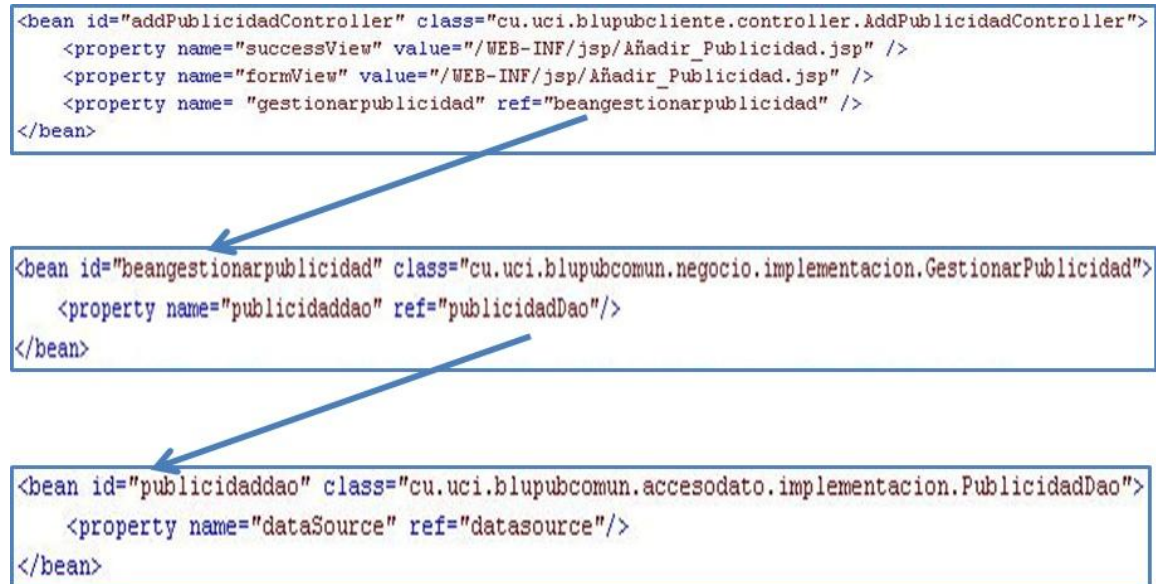


Figura 14: Patrón Inyección de Dependencia.

3.3.3 Patrón DAO (*Data Access Object*)

Este patrón cuenta con diversas fuentes de datos (base de datos, archivos, servicios externos), de tal forma que se encapsula la forma de acceder a la fuente de datos. Se trata de que el software cliente se centre en los datos que necesita y se olvide de cómo se realiza el acceso a los datos o de cuál es la fuente de almacenamiento. [30]

Un DAO define la relación entre la lógica de presentación por una parte y por otra los datos. Tiene un interfaz común, sea cual sea el modo y fuente de acceso a datos. [30]

Algunas características del patrón DAO: [30]

- Resulta útil que implemente una interfaz. De esta forma los objetos cliente tienen una forma unificada de acceder a los datos.
- Accede a la fuente de datos y la encapsula para los objetos clientes. Entendiendo que oculta tanto la fuente como el modo (JDBC) de acceder a ella.

La siguiente figura muestra la utilización del patrón DAO en el desarrollo del sistema:

```

public class PublicidadDao extends SimpleJdbcDaoSupport implements IPublicidadDao
{
    public void insertarPublicidad(Publicidad publicidad)[]
    public void eliminarPublicidad(int idpublicidad)[]
    public List<Publicidad> obtenerPublicidadPorNombre(String nombre) []
    public List<Publicidad> obtenerPublicidad(String nombre_publicidad, Date fecha) []
    public List<Publicidad> listarPublicidad() []
    public List<Publicidad> consultarPublicidad(Date fecha_inicio, Date fecha_fin)[]
    public List<Publicidad> obtener()[]
    public List<Publicidad> obtenerPorId(int id)[]
    private static class PublicidadMapper implements ParameterizedRowMapper<Publicidad>[]
}

```

Figura 15: Patrón DAO.

3.4 Diseño

El diseño tiene el propósito de formular los modelos que se centran en los requisitos no funcionales, así como el dominio de la solución que prepara la implementación y prueba del sistema. Pretende crear un plano del modelo de implementación. [34]

3.5 Diagrama de Paquetes

Es una colección de clases, relaciones, realizaciones de casos de uso, diagramas y otros paquetes que estén de alguna forma relacionados. Es usado para estructurar el modelo de diseño dividiéndolo en partes más pequeñas. Los paquetes de diseño deben usarse fundamentalmente como herramienta organizacional del modelo para agrupar elementos relacionados. [34]

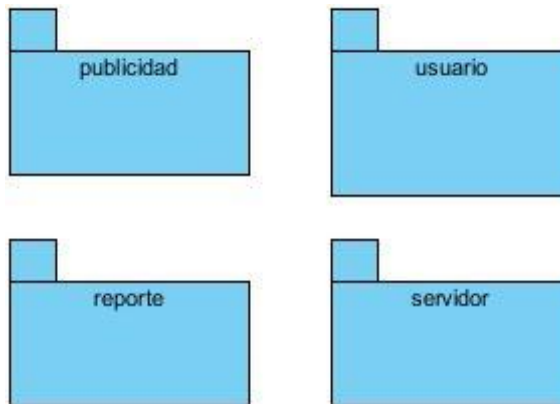


Figura 16: Diagrama de Paquetes.

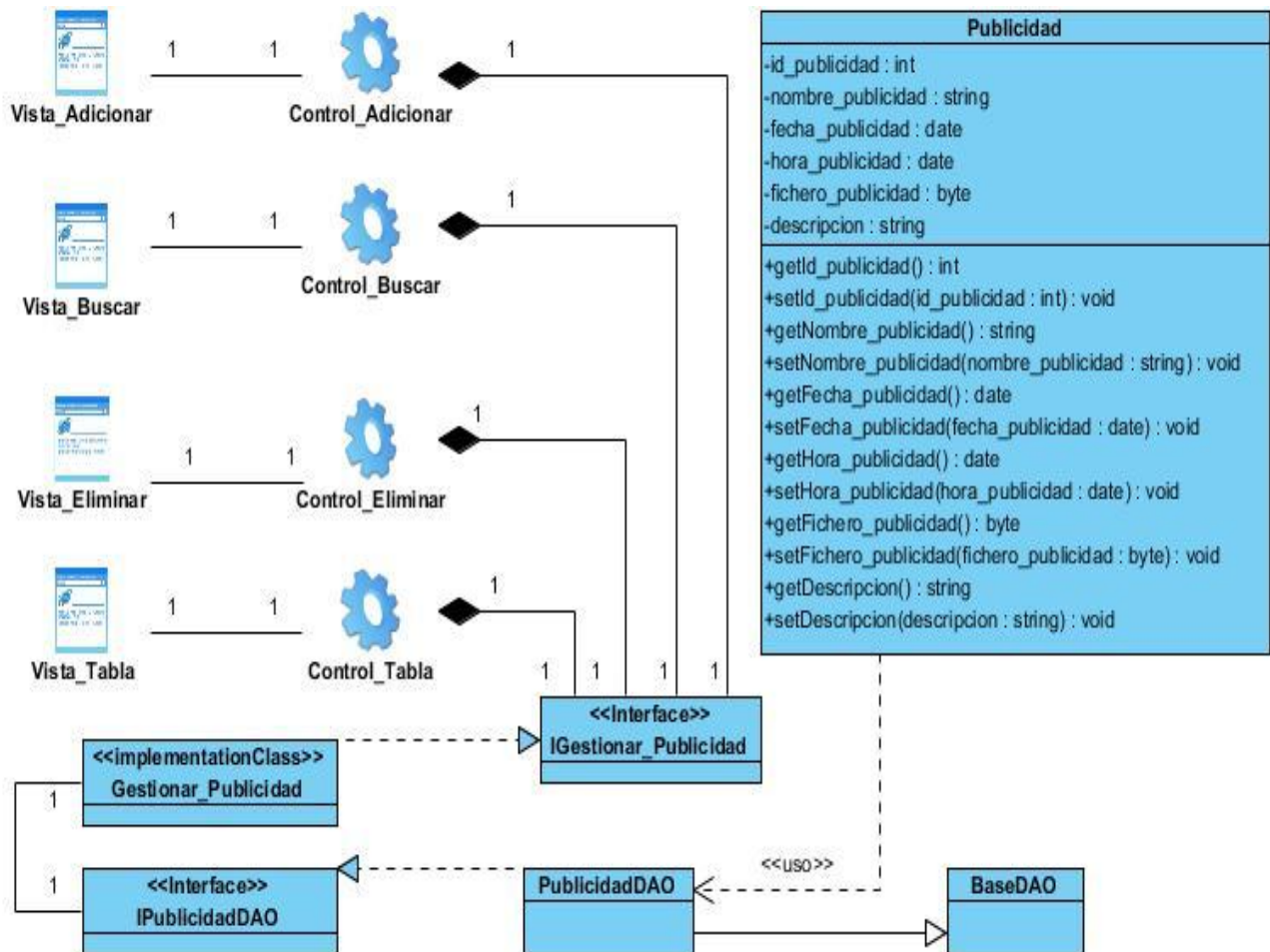
3.6 Diagrama de Clases del Diseño

Un diagrama de clases se utiliza para visualizar las relaciones entre las clases que involucran el sistema, las cuales pueden ser asociativas, de herencia, de uso y de agregación, ya que una clase es una descripción de un conjunto de objetos que comparten los mismos atributos, operaciones, métodos, relaciones y semántica;

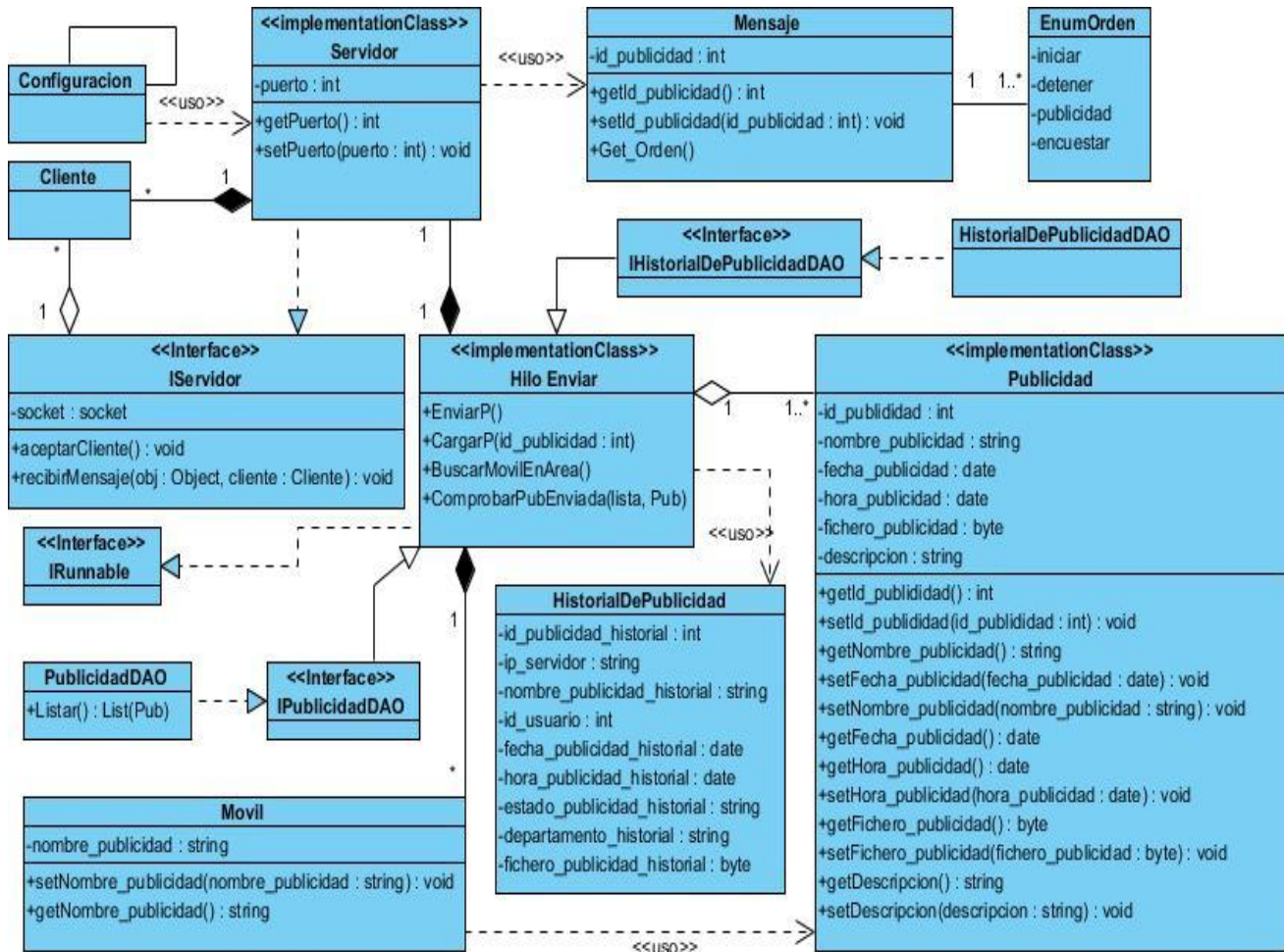
mostrando un conjunto de elementos que son estáticos, como las clases y tipos junto con sus contenidos y relaciones. Un diagrama de clases está compuesto por los siguientes elementos: [35]

- Clase: atributos, métodos y visibilidad.
- Relaciones: Herencia, Composición, Agregación, Asociación y Uso.

3.6.1 Diagrama de Clases del Caso de Uso Gestionar Publicidad.



3.6.2 Diagrama de Clases del Caso de Uso Enviar Publicidad.

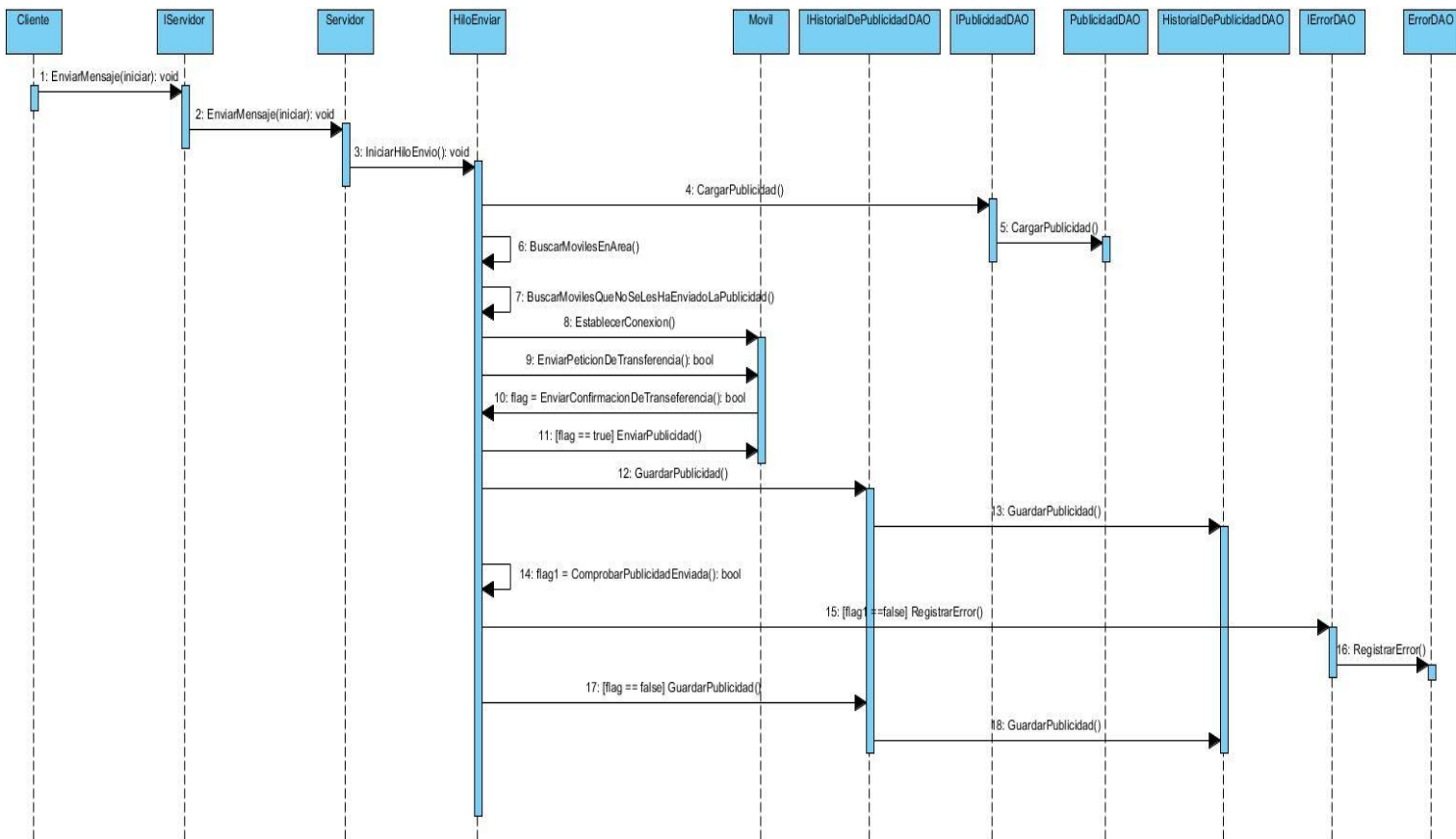


3.7 Diagramas de interacción. Diagrama de secuencia

Los diagramas de interacción se utilizan para modelar los aspectos dinámicos de un sistema. Esto implica modelar instancias concretas o prototípicas de clases, interfaces, componentes y nodos, junto con mensajes enviados entre ellos, todo en el contexto de un escenario que ilustra un comportamiento. Un diagrama de secuencia es un diagrama de interacción que destaca la ordenación temporal de los mensajes. [34]

Un diagrama de secuencia expone los objetos, relaciones y mensajes enviados entre ellos. A continuación se muestra el diagrama de secuencia de un escenario de uno de los casos de usos más significativos, para visualizar los diagramas de secuencia de otros casos de uso remitirse al epígrafe Anexo 3.

3.7.1 Diagrama de Secuencia del Caso de Uso Enviar Publicidad.



Descripción del Diagrama de Secuencia del Caso de Uso Enviar Publicidad:

La secuencia a seguir en el caso de uso Enviar Publicidad comienza una vez levantado el hilo de envío de publicidad cargando las publicidades existentes en la base de datos, luego se realiza una búsqueda de los dispositivos móviles que se encuentren en el área de cobertura que cuenten con tecnología Bluetooth y que tengan la misma encendida en modo visible. Una vez reconocidos los dispositivos, se procede a comparar los identificadores de los mismos con los identificadores de los móviles a los cuales se les ha enviado ya la publicidad, con el objetivo de no volver a repetir el envío a un mismo móvil más de una vez en el tiempo establecido. Luego se establece la conexión con el dispositivo y se le envía la petición de transferencia de archivo; una vez que es aceptada la transferencia se realiza el envío y se procede a guardar los datos de la transferencia en el historial de envío con estado aceptada. Si la transferencia fue rechazada se procede a

guardar en el historial de envío con estado rechazada. En caso de ocurrir algún error en la transferencia, se guarda el error en el registro de errores.

3.8 Diseño de la base de datos

Una de las tareas más importantes en el desarrollo de un sistema que necesita una base de datos es el diseño de la misma. Se diseña con el fin de acceder a los datos cuando es necesario. Una base de datos correctamente diseñada permite obtener acceso a información exacta y actualizada. Un diseño correcto es esencial para lograr los objetivos fijados ya que posibilita mayor calidad en la implementación del sistema. [36]

3.8.1 Modelo de Datos

Describe la representación lógica y física de los datos persistentes usados por el sistema. A continuación se presenta el modelo de datos del sistema:

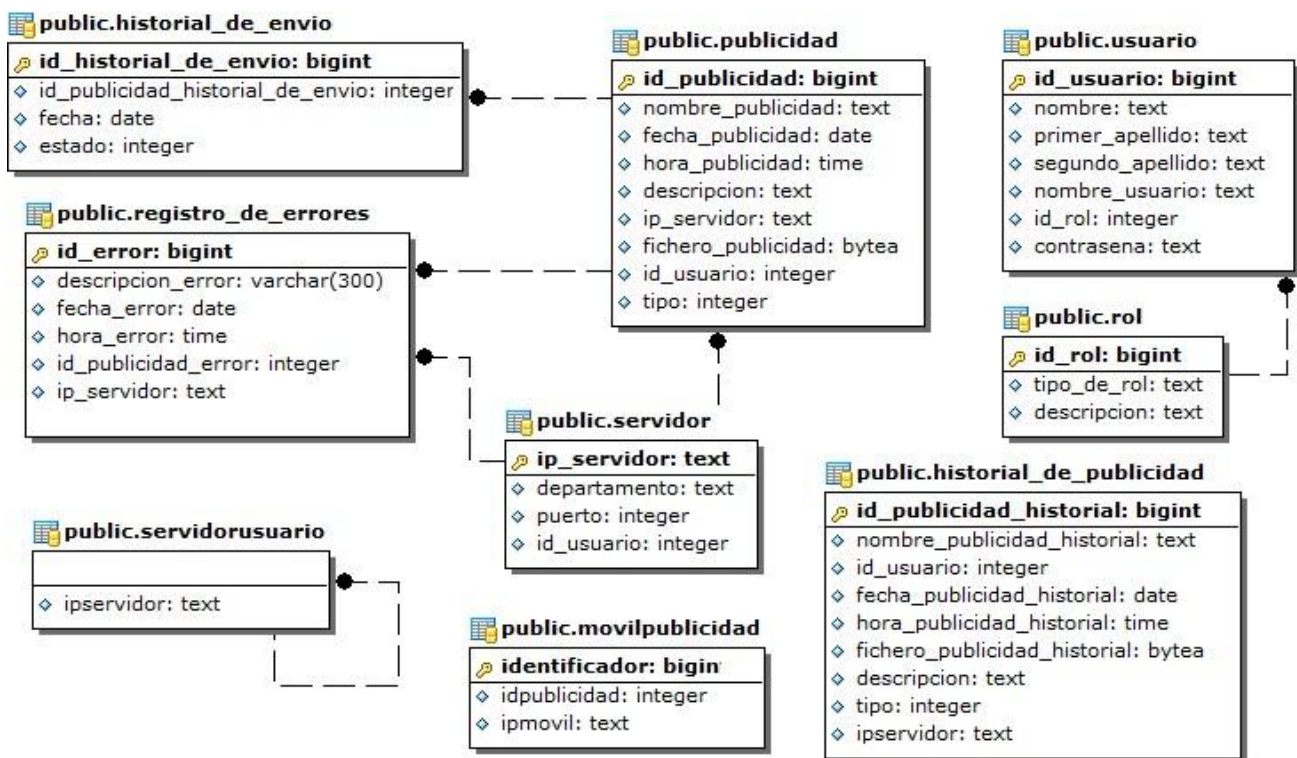


Figura 17: Modelo de Datos.

Para una mejor comprensión del modelo de datos se describen a continuación las tablas que lo componen:

- `historial_de_envio`: almacena los datos de las publicidades enviadas por el sistema y se relaciona con la tabla `publicidad`.

- **publicidad:** contiene los datos de las publicidades existentes en la base de datos y se relaciona además con las tablas registro_de_errores y servidor.
- **servidor:** contiene los datos de los servidores con los que cuenta el sistema y se relaciona con la tabla registro_de_errores.
- **registro_de_errores:** contiene los datos de los errores ocurridos en el sistema para futuros reportes estadísticos.
- **servidorusuario:** representa la relación existente entre la tabla servidor y la tabla usuario.
- **usuario:** contiene los datos de los usuarios que acceden al sistema y tiene relación además con la tabla rol.
- **rol:** contiene los roles que puede tener un usuario del sistema.
- **historial_de_publicidad:** contiene los datos de las publicidades que ya no se encuentran en el sistema, con el objetivo de conservar un historial de las publicidades que han sido eliminadas.
- **movilpublicidad:** contiene los datos de los móviles a los cuales se les ha enviado determinada publicidad, de la cual se registra su identificador, para evitar repetir el envío de la misma a un mismo móvil más de una vez en un tiempo determinado.

3.9 Conclusiones Parciales

En el desarrollo del capítulo se elaboraron los diagramas de paquetes, además los diagramas de clases del diseño y de secuencia de los casos de uso, lo que ha permitido obtener una visión del sistema. Se mostró además el modelo de datos, el estilo y patrones arquitectónicos presentes en la aplicación y los patrones de diseño a usar durante el desarrollo del sistema.

CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBA

4.1 Introducción

El presente capítulo tiene como objetivo abordar la implementación en base a las funcionalidades, donde se describe cómo se implementan en términos de componentes los elementos del modelo de diseño. Se modelan los diagramas de componentes y de despliegue, dando una visión de cómo quedará desarrollado el sistema. Además se especifica el tipo de prueba realizada a todas las funcionalidades necesarias.

4.2 Implementación

Durante la implementación del sistema se define la organización del código, se implementan los elementos del diseño, se integran los resultados en un sistema ejecutable y se distribuyen físicamente asignando componentes ejecutables a nodos.

4.2.1 Diagrama de despliegue

“El modelo de despliegue es utilizado para capturar los elementos de hardware y las conexiones entre ellos. Se utiliza para visualizar la distribución de los componentes de software en los nodos físicos.” [34]

A continuación se muestra el diagrama de despliegue del sistema:

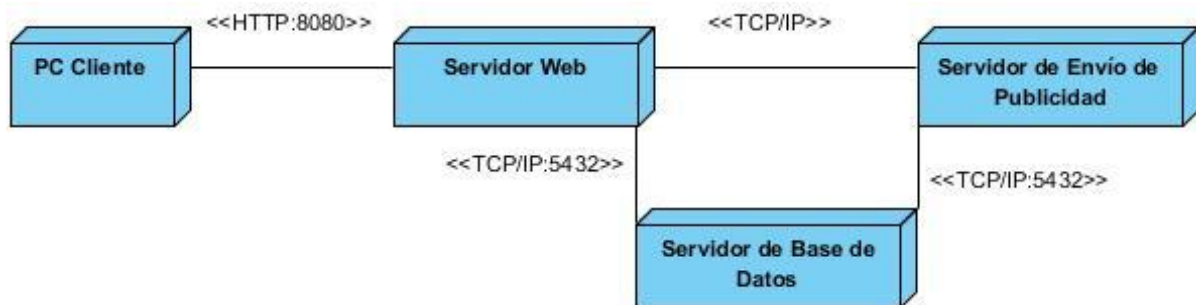


Figura 18: Diagrama de Despliegue.

- PC Cliente: se conecta al servidor de aplicaciones para visualizar las aplicaciones web.
- Servidor Web: contiene la aplicación a la cual se accederá desde las máquinas clientes.
- Servidor de Envío de Publicidad: cuenta con el dispositivo Bluetooth que se encarga de realizar el envío de los ficheros de publicidad hacia los dispositivos móviles que se encuentran dentro del área de cobertura.

- Servidor de Base de Datos: contiene la base de datos donde se encuentran registradas las publicidades, los servidores, los usuarios y los historiales de las publicidades enviadas por el sistema.

4.2.2 Diagrama de componentes

Los diagramas de componentes muestran la organización de los componentes del sistema. Modelan los aspectos físicos, es decir, un conjunto de elementos físicos y sus relaciones. Un componente se corresponde con una o varias clases, interfaces o colaboraciones. Exponen los subsistemas de implementación y sus dependencias de importación y los subsistemas de implementación organizados en capas.

A continuación se expone el diagrama de subsistemas de implementación del sistema BluePub Sender, el diagrama de componentes de la Publicidad y algunos de los componentes presentes en las librerías del sistema.

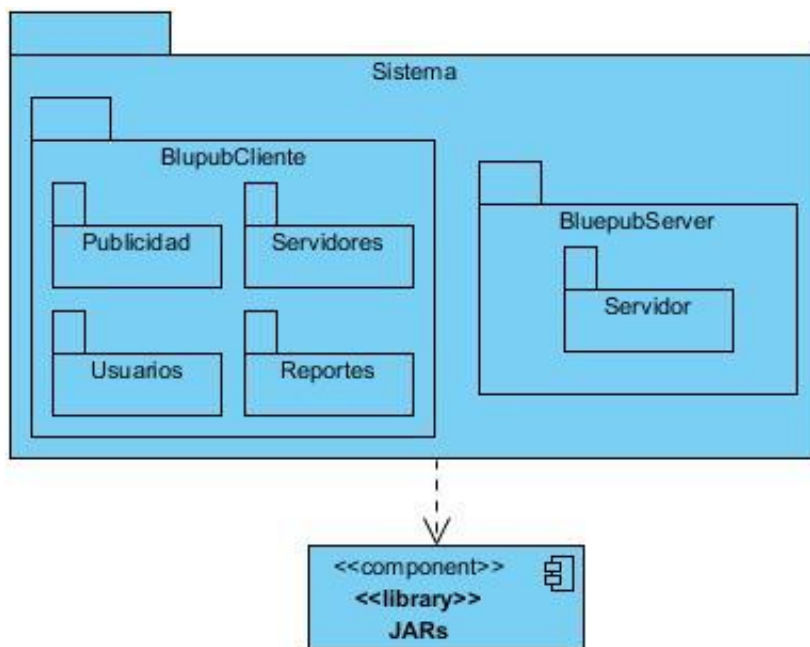


Figura 19: Diagrama de subsistemas de implementación.

Descripción de los elementos:

- Sistema: contiene los artefactos del sistema.
 - BlupubCliente: contiene los artefactos relacionados con las funcionalidades del cliente.
 - BluepubServer: contiene los artefactos relacionados con las funcionalidades del servidor.

- JARs: contiene las librerías usadas, tanto las de Spring como las creadas durante el desarrollo del sistema.

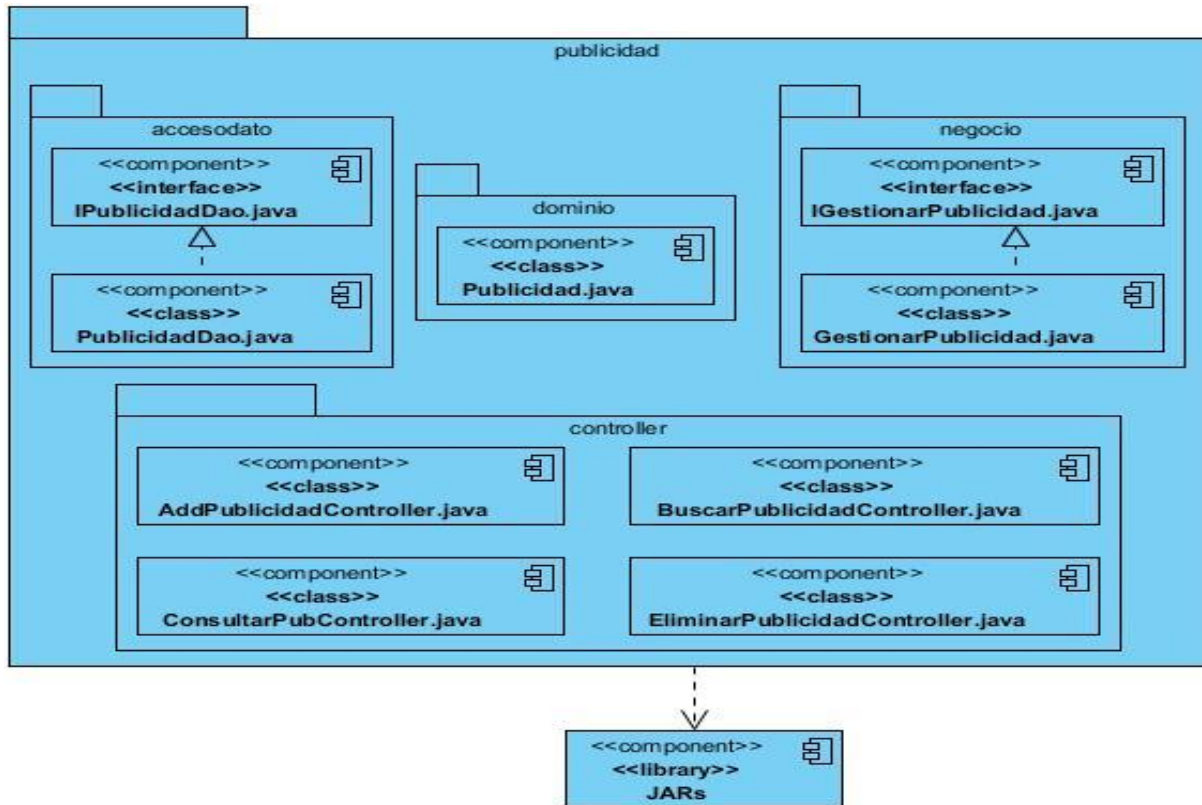


Figura 20: Diagrama de componentes de Publicidad.

Descripción de los elementos:

- accesodato: contiene las clases encargadas de acceder a los datos mediante JDBC.
- dominio: contiene las clases que caracterizan a la publicidad.
- negocio: contiene las clases que permiten relacionar las clases controladoras con las clases de acceso a datos.
- controller: contiene las clases controladoras encargadas de atender las peticiones de los usuarios.

4.3 Pruebas de software

Las pruebas de software son los procesos que permiten verificar la calidad de un producto de software, representando una inspección final de las especificaciones del diseño y la implementación, donde se identificarán posibles fallos. Una prueba no es más que una actividad en la cual un sistema o componente es ejecutado bajo unas condiciones específicas, los resultados son observados y registrados.

4.3.1 Pruebas Unitarias

Las pruebas unitarias son creadas por los propios desarrolladores, comprueban que el código hace lo que se espera de él. Primero se realizan las funcionalidades y luego sus tests.

Estas pruebas fueron utilizadas para probar los accesos a datos de las funcionalidades del sistema.

Por ejemplo para probar la clase PublicidadDao se creó la clase de prueba TestPublicidadDao, en la cual se creó un objeto de la clase PublicidadDao haciendo uso de la clase ApplicationContext; con el que se hace la llamada a los métodos de la clase PublicidadDao a los cuales se les introduce un juego de datos y se le da a la opción Run As Java Application la cual ejecuta el código y realiza la operación indicada, mostrando los datos en caso de que el método a probar devuelva algún dato, en caso de no devolver nada, se comprueba la eficiencia de la prueba en la base de datos.

A continuación se muestra una tabla donde se representan las clases que fueron probadas con su respectiva clase de prueba:

Clases Probadas	Clases de Prueba
GestionarPublicidad	TestGestionarPublicidad
GestionarServidor	TestGestionarServidor
GestionarUsuario	TestGestionarUsuario
HistorialDeEnvioN	TestHistorialDeEnvioN
RegistroDeErroresN	TestRegistroDeErroresN
ServidorDao	TestServidorDao
UsuarioDao	TestUsuarioDao
PublicidadDao	TestPublicidadDao

Tabla 4: Pruebas Unitarias.

4.3.2 Modelo de prueba. Pruebas de Funcionalidad

Las pruebas de funcionalidad aseguran el trabajo apropiado de los requisitos funcionales, incluyendo la navegación, entrada de datos, procesamiento y obtención de resultados. Se ejecuta en cada funcionalidad de caso de uso, corroborando la implementación adecuada de las reglas del negocio. Su uso permite

CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBA

conocer si mediante la introducción de datos válidos, se obtienen los resultados esperados, además de que sean exhibidos los mensajes apropiados de error y precaución cuando se usan datos inválidos.

Descripción de los casos de prueba

Los casos de prueba especifican una forma de probar el sistema, incluyendo las entradas con las que se ha de probar, los resultados esperados y las condiciones bajo las que ha de probarse.

Para comprobar el correcto funcionamiento del sistema BluePub Sender se definieron un conjunto de casos de pruebas que reúnen la mayor cantidad de escenarios posibles. A continuación se muestra el caso de prueba Gestionar Publicidad, los restantes se encuentran en los ANEXOS:

➤ CP Gestionar Publicidad:

Descripción General: Permite al usuario añadir, consultar y eliminar las publicidades.

Condiciones de Ejecución: El usuario debe estar autenticado, con nombre de usuario y contraseña introducidos correctamente.

1. SC AÑADIR PUBLICIDAD:

Escenario	Descripción	Nombre	Descripción	Ip Servidor	Fichero	Respuesta del sistema	Flujo central
<i>EC 1.1 Se añade la publicidad satisfactoriamente</i>	<i>El usuario introduce los datos de la publicidad a registrar.</i>	V	V	V	V	<i>El sistema inserta la publicidad con los datos especificados</i>	<i>1-El usuario ubica el cursor en la opción del menú "Publicidad".</i>
		V	V	V	V	<i>El sistema inserta la publicidad con los datos especificados</i>	<i>2-Da clic en la opción "Añadir Publicidad".</i> <i>3-El sistema muestra la interfaz de "Añadir Publicidad".</i> <i>4-Entra los datos</i>

CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBA

							solicitados. 5-Da clic en el botón "Aceptar".
EC 1.2 Campos Vacíos	El usuario deja de insertar algún dato.	V	NA	V	V	Se muestra un mensaje al lado del campo sin llenar con el siguiente texto: "Por favor, rellene este campo".	1-El usuario ubica el cursor en la opción del menú "Publicidad". 2-Da clic en la opción "Añadir Publicidad". 3-El sistema muestra la interfaz de "Añadir Publicidad". 4-Entra los datos solicitados. 5-Da clic en el botón "Aceptar".
EC 1.3 Caracteres Extraños	El usuario introduce datos con caracteres extraños.	V	V	I	V	Se muestra un mensaje: "Debe llenar los campos correctamente".	1-El usuario ubica el cursor en la opción del menú "Publicidad". 2-Da clic en la opción "Añadir Publicidad". 3-El sistema muestra la interfaz de

CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBA

							<p><i>“Añadir Publicidad”.</i></p> <p><i>4-Entra los datos solicitados.</i></p> <p><i>5-Da clic en el botón “Aceptar”.</i></p>
--	--	--	--	--	--	--	--

2. SC CONSULTAR PUBLICIDAD:

Escenario	Descripción	Fecha Inicio	Fecha Fin	Respuesta del sistema	Flujo central
<i>EC 2.1 Se consulta la publicidad satisfactoriamente</i>	<i>El usuario introduce los datos de la publicidad a consultar.</i>	<i>V</i>	<i>V</i>	<i>El sistema consulta la publicidad con los datos especificados</i>	<p><i>1-El usuario ubica el cursor en la opción del menú “Publicidad”.</i></p> <p><i>2-Da clic en la opción “Consultar Publicidad”.</i></p>
		<i>V</i>	<i>V</i>	<i>El sistema consulta la publicidad con los datos especificados</i>	<p><i>3-El sistema muestra la interfaz de “Consultar Publicidad”.</i></p> <p><i>4-Entra los datos solicitados.</i></p> <p><i>5-Da clic en el botón “Consultar Publicidad”.</i></p>

CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBA

<p><i>EC 2.2 Campos Vacíos</i></p>	<p><i>El usuario deja de insertar algún dato.</i></p>	<p>V</p>	<p>NA</p>	<p><i>Se muestra un mensaje con el siguiente texto: "Debe rellenar los campos vacíos."</i></p>	<p><i>1-El usuario ubica el cursor en la opción del menú "Publicidad".</i></p> <p><i>2-Da clic en la opción "Consultar Publicidad".</i></p> <p><i>3-El sistema muestra la interfaz de "Consultar Publicidad".</i></p> <p><i>4-Entra los datos solicitados.</i></p> <p><i>5-Da clic en el botón "Consultar Publicidad".</i></p>
------------------------------------	---	----------	-----------	--	--

3. SC ELIMINAR PUBLICIDAD:

Escenario	Descripción	Nombre	Fecha	Respuesta del sistema	Flujo central
<p><i>EC 3.1 Se elimina la publicidad satisfactoriamente</i></p>	<p><i>El usuario introduce los datos de la publicidad a eliminar.</i></p>	<p>V</p>	<p>V</p>	<p><i>El sistema elimina la publicidad con los datos especificados</i></p>	<p><i>1-El usuario ubica el cursor en la opción del menú "Publicidad".</i></p> <p><i>2-Da clic en la opción "Eliminar Publicidad".</i></p>
		<p>V</p>	<p>V</p>	<p><i>El sistema elimina la publicidad con los datos</i></p>	<p><i>3-El sistema muestra la interfaz de</i></p>

CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBA

				especificados	<p>“Eliminar Publicidad”.</p> <p>4-Entra los datos solicitados.</p> <p>5-Marca la publicidad que desea eliminar.</p> <p>6-Da clic en el botón “Eliminar”.</p>
EC 3.2 Datos Incorrectos.	El usuario introduce datos incorrectos de la publicidad a eliminar.	I	V	Se muestra un mensaje con el siguiente texto: “No se encontraron resultados”.	<p>1-El usuario ubica el cursor en la opción del menú “Publicidad”.</p> <p>2-Da clic en la opción “Eliminar Publicidad”.</p> <p>3-El sistema muestra la interfaz de “Eliminar Publicidad”.</p> <p>4-Entra los datos solicitados.</p> <p>5-Marca la publicidad que desea eliminar.</p> <p>6-Da clic en el botón “Eliminar”.</p>

4. DESCRIPCIÓN DE LAS VARIABLES:

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	Nombre	Campo de texto	No	Cadena de texto en la que no se aceptan caracteres especiales.
2	Descripción	Campo de texto	No	Cadena de texto que describe la publicidad.
3	Ip Servidor	Campo de texto	No	No se aceptan caracteres especiales.
4	Fichero	Archivo adjunto	No	Los adjuntos deben estar en formato de: texto, imagen o video.
5	Fecha	Campo de selección	No	Predeterminados con el siguiente formato dd-MM-yyyy

4.3.3 No conformidades detectadas en el sistema

A continuación se muestran la cantidad de no conformidades que han sido detectadas en el sistema, mostrando cuántas de estas proceden, cuántas no proceden y cuántas fueron resueltas; todas fueron no significativas.

Sistema	Total de no conformidades	# de no conformidades que proceden	# de no conformidades que no proceden	# de no conformidades resueltas
BluePub Sender	8	7	1	6

Tabla 5: No conformidades detectadas en el sistema.

4.4 Conclusiones Parciales

En este capítulo se han desarrollado los aspectos que muestran la implementación del sistema. Se realizó el diagrama de subsistemas de implementación así como los diagramas de componentes. Se realizó el diagrama de despliegue, el cual muestra cómo estará distribuido el sistema en un plano físico. Se realizaron pruebas para comprobar correcto funcionamiento del sistema que validan que el mismo cumple los requisitos establecidos.

CONCLUSIONES

En el presente trabajo se describieron todos los procesos identificados para el envío de publicidad haciendo uso de la tecnología Bluetooth, para mejorar el trabajo de los publicistas y propiciar el ahorro de recursos en las campañas publicitarias. Se seleccionaron las herramientas, tecnologías y metodologías de desarrollo de software que se utilizaron. Se definió la arquitectura a utilizar así como los patrones arquitectónicos para el diseño e implementación del sistema.

Con la realización del sistema de envío de publicidad se le da solución a una serie de problemas existentes en el proceso de realización de las campañas publicitarias:

- Los anuncios publicitarios no llegan directamente a la mayoría de los usuarios potenciales de las entidades de venta.
- No existe un sistema que haga llegar a los clientes las nuevas ofertas de las entidades de venta.
- Se desconoce el nivel de impacto de determinadas publicidades en las entidades de venta.
- No se tiene un control de la gestión de la publicidad.

Por todo lo expuesto anteriormente se concluye que se cumplieron los objetivos propuestos cumpliendo cada una de las tareas propuestas para desarrollar el sistema de envío de publicidad BluePub Sender.

RECOMENDACIONES

Después de terminado el desarrollo del sistema se detectaron algunas carencias que pueden incluirse como posibles mejoras al mismo. A continuación se detallan las recomendaciones realizadas al sistema:

1. La publicidad añadida al sistema se elimine automáticamente en un tiempo elegido por el usuario.
2. Incorporar otros servicios al sistema tales como el envío de mensajes SMS.
3. Generar reportes estadísticos que evidencien las preferencias de los clientes de las entidades de venta.
4. Ajustar el contenido (imágenes y video) a las dimensiones y requerimientos técnicos de los dispositivos móviles a los cuales se le envíe publicidad.
5. Extender la compatibilidad del sistema a dispositivos móviles que no cuentan con el servicio OBEX de transferencia de archivos.

REFERENCIAS BIBLIOGRÁFICAS

- [1] O. R. Gámez. (2005, 29 de mayo de 2012). Telefonía móvil celular: origen, evolución, perspectivas. *Revista de Ciencias Holguín* Available: <http://www.ciencias.holguin.cu/2005/marzo/articulos/ART12.htm>
- [2] G. NTC. (2010, 22 de noviembre de 2011). BLUETHOOTH por PC. Available: http://www.grupontc.com/admin/img_aux/imp_e2a39f1a80ac50e2e551a9b89a5daa79.pdf
- [3] (2009, 22 de noviembre de 2011). *Publicidad por Bluetooth Sendig*. Available: http://www.valor-agregado.com/index.php?option=com_content&view=article&id=49&Itemid=53
- [4] (2009, 22 de noviembre de 2011). *BlueMessenger: Sistema de publicidad bluetooth*. Available: <http://www.ditecom.com/bluetooth/index.htm>
- [5] (2009, 7 de diciembre de 2011). *Blue Space* Available: http://entumovil.cu/index.php?option=com_content&view=article&id=115&Itemid=107.
- [6] P. Borches. (2004, diciembre de 2011). Java 2 Micro Edition: Soporte Bluetooth. . Available: http://www.it.uc3m.es/celeste/docencia/j2me/tutoriales/bluetooth/EstudioTecnologico1_0.pdf
- [7] (2008, 29 de mayo de 2012). Lenguajes de programación. Available: http://guimi.net/descargas/Monograficos/G-Lenguajes_de_programacion.pdf
- [8] G. Álvarez. (1999, 28 de noviembre de 2011). *Características del lenguaje Java* Available: <http://www.iec.csic.es/criptonomicon/java/quesjava.html>
- [9] E. Hernández. (2 de diciembre de 2011). *El Lenguaje Unificado de Modelado (UML)*.
- [10] M. Pereira. (2007, 2 de diciembre de 2011). UML. Available: <http://es.scribd.com/doc/2080534/UML>
- [11] C. Walls, *Spring in Action* 2da ed. Greenwick: Manning editorial Publicaciones 2007.
- [12] (2007, 5 de diciembre de 2011). *Comparativa de Frameworks: Spring*. Available: <http://seamcity.madeinpain.com/archives/comparativa-spring>
- [13] (13 de enero de 2012). *Dojo Toolkit* Available: <http://dojotoolkit.org/>
- [14] (2011, 9 de enero de 2012). JDBC: acceso a bases de datos Available: <http://www.vc.ehu.es/jiwotvim/ISOFT2010-2011/Teoria/BloqueIV/JDBC.pdf>
- [15] (8 de junio de 2012). *Socket*. Available: <http://www.ecured.cu/index.php/Socket>
- [16] (2008, 15 de marzo de 2012). *BlueCove*. Available: <http://bluecove.org/>
- [17] R. P. I. Carrillo, A. Rodríguez. (2008, 6 de diciembre de 2011). Metodología de Desarrollo del Software. Available: http://www.google.com/cu/url?sa=t&rct=j&q=Metodolog%C3%ADa+de+Desarrollo+del+Software.+2008.&source=web&cd=2&ved=0CFcQFjAB&url=http%3A%2F%2Fsolusoft-g11.googlecode.com%2Ffiles%2FMetodologias%2520de%2520desarrollo.pdf&ei=wnTHT9PQLYqVqweE-8GODg&usq=AFQjCNFZ9_K-b0YEM-twoVcsjssFhxWgQQ
- [18] A. Molpeceres. (2002, 6 de diciembre de 2011). Procesos de Desarrollo: RUP, XP y FDD. Available: <http://www.willydev.net/descargas/Articulos/General/cualxpfdrup.PDF>
- [19] (6 de diciembre de 2011). *Eclipse, entorno de desarrollo integrado*. Available: http://www.ecured.cu/index.php/Eclipse,_entorno_de_desarrollo_integrado
- [20] (2010, 1 de diciembre de 2011). *Elección del servidor de aplicaciones web*. Available: <http://tomcat.apache.org>
- [21] R. L. Bagüés. (2007, 1 de diciembre de 2011). *Tomcat*. Available: <http://www.proactiva-calidad.com/java/herramientas/tomcat/index.html>
- [22] (28 de mayo de 2012). *PostGreSQL*. Available: <http://www.ecured.cu/index.php/PostGreSQL>

REFERENCIAS BIBLIOGRÁFICAS

- [23] (2010, 2 de diciembre de 2011). Herramientas Case Available: <http://es.scribd.com/doc/3062020/Capitulo-I-HERRAMIENTAS-CASE>
- [24] (2007, 2 de diciembre de 2011). *Paradigma visual para UML*. Available: http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%5Bcuenta_de_Plataforma_de_Java_14715_p/
- [25] M. Sierra. (2009, 5 de diciembre de 2011). Ingeniería del Software I, Trabajando con Visual Paradigm for UML. Available: <http://personales.unican.es/ruizfr/is1/doc/lab/01/is1-p01-trans.pdf>
- [26] (2011, 10 de enero de 2012). *JFreeChart* Available: <http://www.jfree.org/jfreechart/>
- [27] C. B. Reynoso. (2004, 06 de febrero de 2012). Introducción a la Arquitectura de Software Available: <http://www.willydev.net/descargas/prev/IntroArg.pdf>
- [28] S. Álvarez. (2007, 06 de febrero de 2012). *Arquitectura Cliente-Servidor* Available: <http://www.desarrolloweb.com/articulos/arquitectura-cliente-servidor.html>
- [29] (2009, 06 de febrero de 2012). *Arquitectura Cliente-Servidor* Available: <http://ccia.ei.uvigo.es/docencia/SCS/1011/transparencias/Tema1.pdf>
- [30] R. Lago. (2007, 06 de febrero de 2012). *Patrones de Diseño* Available: <http://www.proactiva-calidad.com/java/patrones/index.html>
- [31] (2010, 06 de febrero de 2012). *Modelo Vista Controlador – Definición y Características*. Available: <http://www.comusoft.com/modelo-vista-controlador-definicion-y-caracteristicas>.
- [32] M. Visconti and H. Astudillo. 08 de febrero de 2012). *Fundamentos de Ingeniería de Software*. Available: <http://www.inf.utfsm.cl/~visconti/ili236/Documentos/08-Patrones.pdf>
- [33] (06 de febrero de 2012). *Inyeccion De Dependencia* Available: http://www.dosideas.com/wiki/Inyeccion_De_Dependencia
- [34] (28 de mayo de 2012). *Flujo de Trabajo Análisis y Diseño*. Available: http://www.ecured.cu/index.php/Flujo_de_Trabajo_An%C3%A1lisis_y_Dise%C3%B1o
- [35] (28 de mayo de 2012). *Diagrama de Clase*. Available: http://www.ecured.cu/index.php/Diagrama_de_Clase
- [36] (2007). *Conceptos básicos del diseño de una base de datos*. Available: <http://office.microsoft.com/es-es/access-help/conceptos-basicos-del-diseno-de-una-base-de-datos-HA001224247.aspx>

BIBLIOGRAFÍA

1. B. Hopkins and R. Antony, "Bluetooth for Java," ed, 2003.
2. A. G. Brieba. Programación de dispositivos Bluetooth a través de Java.
3. A. G. Brieba. (2004). *JSR-82: Bluetooth desde Java*.
4. W. Giuliatti. *Panorama Actual de la Publicidad*.
5. R. Seris. Servicios Informativos para Centros Comerciales, Usando Tecnología Bluetooth.
6. J. M. C. García. (2009). *Sistema de Acceso Bluetooth*.
7. P. C. Morales. Desarrollo de Aplicaciones Distribuidas basadas en Tecnologías Web.
8. J. Gutierrez. Conociendo distintos tipos de protocolos.
9. P. D. B. Juzgado. (2004). *Java 2 Micro Edition: Soporte Bluetooth*.
10. C. Walls. (2008). *Spring in Action (Second Edition ed.)*.
11. J. E. Harmon. (2009). *Dojo Using the Dojo JavaScript Library to Build Ajax Applications*.
12. D. Gilbert. (2005). *The JFreeChart Class Library*.

GLOSARIO

- **Bluetooth:** es una especificación para Redes Inalámbricas que posibilita la transmisión de voz y datos entre diferentes dispositivos mediante un enlace por radiofrecuencia.
- **PDA:** del inglés (personal digital assistant), asistente digital personal, también denominado ordenador de bolsillo u organizador personal, es una computadora de mano originalmente diseñada como agenda electrónica (calendario, lista de contactos, bloc de notas y recordatorios).
- **API:** interfaz de programación de aplicaciones (del inglés Application Programming Interface) es el conjunto de funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción. Son usadas generalmente en las denominadas "librerías".
- **JSP:** permite generar contenido dinámico para web, en forma de documentos HTML, XML o de otro tipo. JavaServer Pages permiten la utilización de código Java mediante scripts, además es posible utilizar algunas acciones JSP predefinidas mediante etiquetas.
- **IDE:** entorno de desarrollo integrado, en inglés integrated development environment, es un programa informático compuesto por un conjunto de herramientas de programación. Puede dedicarse en exclusiva a un solo lenguaje de programación o bien poder utilizarse para varios.
- **BSD:** es una licencia de software libre permisiva, permite el uso del código fuente en software no libre. El autor, bajo esta licencia, mantiene la protección de los derechos de autor únicamente para la renuncia de garantía y para requerir la adecuada atribución de la autoría en trabajos derivados, pero permite la libre redistribución y modificación, permitiéndose el uso en software comercial.
- **JDBC:** en inglés Java Database Connectivity, es una API que permite la ejecución de operaciones sobre bases de datos desde el lenguaje de programación Java, independientemente del sistema operativo donde se ejecute o de la base de datos a la cual se accede.
- **GRASP:** son patrones generales de software para asignación de responsabilidades, es el acrónimo de "General Responsibility Assignment Software Patterns".
- **DAO:** un Data Access Object (Objeto de Acceso a Datos) es un componente de software que suministra una interfaz común entre la aplicación y uno o más dispositivos de almacenamiento de datos, tales como una Base de datos o un archivo. El término se aplica frecuentemente al Patrón de diseño Object.
- **Arquitectura:** la arquitectura es el diseño de más alto nivel de la estructura de un sistema.

- **Casos de Uso:** son una técnica para especificar el comportamiento de un sistema, un caso de uso es una secuencia de interacciones entre un sistema y sus actores.
- **Actor:** entidad externa al sistema que guarda una relación con éste y que le demanda una funcionalidad.
- **HTTP:** en inglés Hypertext Transfer Protocol, (protocolo de transferencia de hipertexto) define la sintaxis y la semántica que utilizan los elementos de software de la arquitectura web (clientes, servidores, proxies) para comunicarse.
- **CP:** Caso de prueba.
- **SC:** Escenario.
- **Orientación a Aspectos (AOP):** derivado de Programación Orientada a Aspectos (POA o AOP), es un paradigma de programación relativamente reciente cuya intención es permitir una adecuada modularización de las aplicaciones y posibilitar una mejor separación de conceptos.