Universidad de las Ciencias Informáticas.

Título:

Sistema de Agenda de Compromisos Telefónica (SACT).

Trabajo de Diploma para optar por el título de Ingeniería en Ciencias Informáticas

Autores:

Yeisell González Medina

Yohandy Casañas do Nacimento

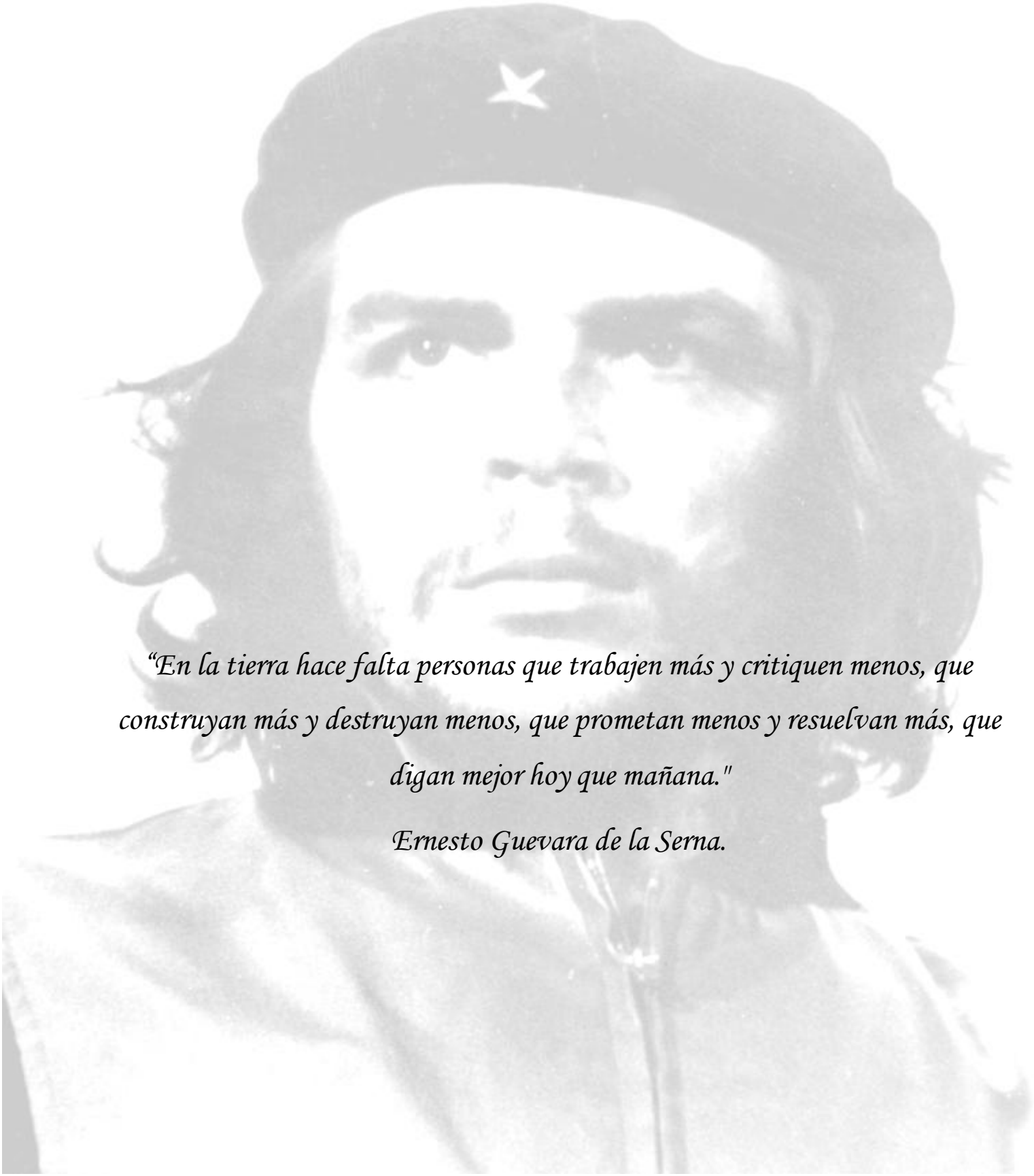
Tutor:

Ing. Arianna Pérez Carmenates

Co-tutor

Ing. Gerardo Rodríguez Fernández

La Habana, Junio 2011



"En la tierra hace falta personas que trabajen más y critiquen menos, que construyan más y destruyan menos, que prometan menos y resuelvan más, que digan mejor hoy que mañana."

Ernesto Guevara de la Serna.

Declaración de autoría:

Declaramos que somos los únicos autores de este trabajo y autorizamos al Centro de Telemática (TLM) de la Facultad 2 y a la Universidad de las Ciencias Informáticas (UCI) a hacer el uso del mismo en su beneficio.

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

Yeisell González Medina **(Autor)**	Yohandy Casañas do Nacimento **(Autor)**
Ing. Arianna Pérez Carmenates **(Tutor)**	Ing. Gerardo Rodríguez Fernández **(Co-Tutor)**

Dedicatoria:

De dicado A:

A mi mamá por ser mi tesoro…

Yeisell.

A mis padres y mi hermano…

Yohandy.

Agradecimientos:

De los Autores:

A Denia, Javier, Yasmin, Ricardo y Ráiner por estar siempre a nuestro lado tanto en los buenos y como en los malos momentos durante estos cinco años de la carrera, a ellos: gracias por la amistad incondicional que nos han brindado.

A nuestra tutora Arianna por la paciencia, el apoyo y la ayuda de la que hemos sido objeto desde el primer instante que supo que era nuestra tutora.

A Norbel por estar ahí para ayudarnos cuando más se le necesitaba y por esa amistad tan hermosa.

A Orelvis, Denier y José Raúl por tener tiempo para nosotros, aun cuando estaban complicados con sus trabajos de diplomas.

A Rober por su ayuda.

Al profe Ariel por los consejos y mostrarnos los disímiles caminos del mundo de la programación.

Al profe Erick Castillo por guiarnos durante el inicio de este trabajo de diploma.

Al profe Abelito por la ayuda y la amistad.

A todos los amigos de los diferentes grupos por los que hemos pasado y en especial a los del 2302-2402.

A todas aquellas personas que han contribuido a nuestra formación como profesional y a la realización del presente trabajo...

Agradecimientos

De Yeisell:

A mi mamá por ser esa personita especial que me ha mostrado el camino a seguir, por respetar mis decisiones, por estar siempre a mi lado apoyándome incondicionalmente, pero sobre todo por la paciencia.

A Orlando por el cariño sincero que me ha brindado siempre.

A mis abuelos Ariel y Oralia por estar ahí para mí.

A Yany y Yunior por soportarme.

A mis tíos, primos, en fin a toda mi familia por el amor, la comprensión y el apoyo que he recibido constantemente.

A Yohandy por estar en todo momento a mi lado, comprenderme, ayudarme y demostrarme cada día que puedo ser capaz de lograr grandes cosas.

A Deivis por ser ese amigo que está cuando lo necesitas.

A Kathy y Dicelys por brindarme su amistad de forma incondicional, pero sobre todo por "soportarme", se que la tarea no ha sido fácil.

A Karinee, Ruby y Hernán por el cariño y apoyo que he recibido desde el día en que los conocí, por tratarme como si fuera parte de la familia.

A todas aquellas personas, a los que están y a los que ya no están gracias por haber contribuido de una forma u otra a alcanzar mis metas.

De Yohandys:

A mis padres por ser lo mejor del mundo, por estar siempre a mi lado en las buenas y en las malas.

A mi hermano por apoyarme en todo momento.

Agradecimientos

A mis abuelos por su apoyo incondicional.

A Maribel y Abad por apoyarme siempre.

A Yuri, Daniel, Yordanka, Yerandy, Kathy y Dicelys por su ayuda y amistad.

A las personas del barrio por estar siempre preocupados por mí.

A Yeisell por su cariño incondicional, por estar todos estos años soportándome, ayudándome en todo lo que necesité, tanto en las buenas como en las malas, demostrándome y enseñándome a ser una mejor persona.

A todas aquellas personas que han ayudado de una forma u otra a lograr mis metas.

A todas ellas, un millón de GRACIAS….

Los Autores…

Resumen:

Cuba no cuenta con un sistema de avisos para compromisos en la telefonía fija, existiendo solamente agendas tradicionales para satisfacer las necesidades. El principal problema que trae consigo el uso de las agendas tradicionales se encuentra centrado en el constante anotar y verificar los detalles de las reuniones, compromisos o eventos que son registrados. Aunque este no es el único inconveniente, pues la información plasmada en sus páginas es difícil de explotar y no dispone de medidas de protección frente a la confidencialidad de los datos.

La Empresa de Telecomunicaciones de Cuba SA (ETECSA) se encuentra enfrascada en la mejora de los servicios de telefonía, sumándose a la idea de integrar aplicaciones de ordenadores con los teléfonos en sistemas de agendas personales. Para un mayor entendimiento de las funciones de las agendas se realizó un análisis minucioso de las existentes a nivel mundial, tomando ideas y experiencias para la implementación de la solución propuesta.

Como resultado final se obtuvo una aplicación web capaz de gestionar las solicitudes de compromisos y envió de las solicitudes de compromiso, eliminando de esta forma la constante consulta de la información. Además la información almacenada en la aplicación es de fácil exploración y se dispone de medidas de protección contra accesos no autorizados.

Palabras Claves:

Agendas, compromisos, servicios, telefonía.

Índice

Índice

Índice

Introducción:

El hombre siempre ha sentido la necesidad de comunicarse a lo largo de su evolución como especie, para esto ha empleado diferentes formas, las cuales van desde la comunicación por señas, hasta la comunicación a distancia por medio de dispositivos tecnológicos. Este avance se ha logrado gracias a la invención de numerosos aparatos entre los cuales despunta el telégrafo eléctrico, que permitió enviar mensajes con letras y números, lo cual dio paso a la aparición del teléfono, con el que fue posible comunicarse mediante la voz. Estos artefactos han permitido la evolución de la forma de comunicarse entre las personas desde diferentes puntos del planeta, por lo que se ha acuñado un nuevo término en el hacer cotidiano mundial “telecomunicaciones”.

Se denomina telecomunicación “a la técnica de transmitir un mensaje desde un punto a otro, normalmente con el atributo típico adicional de ser bidireccional; esta palabra proviene del griego tele, que significa distancia. Por tanto, este término cubre todas las formas de comunicación a distancia, incluyendo radio, telegrafía, televisión, telefonía, transmisión de datos e interconexión de ordenadores” (1). La evolución de estas a nivel mundial ha ocurrido de formas que no era posible imaginar en el momento en que surgieron, los avances logrados en esta área han permitido que el hombre se desempeñe de una manera más eficiente en su vida cotidiana.

En Cuba el Ministerio de la Informática y las Comunicaciones (MIC) es el “órgano regulador de las redes, servicios de informática y comunicaciones, el cual tiene la misión de ordenar la operación e impulsar el desarrollo de las tecnologías de la información y las comunicaciones” (2). Dentro de este ministerio existen varias empresas e instituciones que mantienen una estrecha relación entre ellas; las cuales se encargan del desarrollo de las telecomunicaciones en el país. Entre ellas se encuentra la Empresa de Telecomunicaciones de Cuba SA (ETECSA); organismo que ha venido realizando un arduo trabajo en la realización de diferentes proyectos que contribuyan al incremento de los servicios telefónicos. Por otro lado se encuentra la Universidad de las Ciencias Informáticas (UCI) con sus diferentes centros de producción, donde se puede mencionar como un puntero en las soluciones informáticas en esta materia el Centro de Telemática (TLM) de la facultad 2.

En estos momentos se trabaja en la mejora e incremento de los servicios de la telefonía en Cuba. Uno de los servicios que se quiere comenzar a brindar es el Sistema de Agenda de Compromisos Telefónica para telefonía fija, servicio que hasta el momento sólo lo brindan los móviles, algunas aplicaciones como el correo electrónico o programas para computadoras que establecen sistemas de calendarios sincronizados con alarmas.

Introducción

En la actualidad Cuba no cuenta con un sistema de avisos para compromisos en la telefonía fija, existiendo solamente agendas tradicionales para satisfacer las necesidades de las personas. El principal problema que trae consigo el uso de las agendas tradicionales se encuentra centrado en el constante anotar y verificar los detalles de las reuniones, compromisos o eventos que son registrados.

Aunque este no es el único inconveniente, pues la información plasmada en sus páginas es difícil de explotar y no dispone de medidas de protección frente a la confidencialidad de los datos, de modo que cualquier persona puede acceder a ella.

Otras molestias son las pérdidas, el deterioro o el espacio limitado que poseen para anotar los compromisos. Sin embargo la mayor dificultad que acarrean es la de no disponer de recordatorios, por lo que la inclusión de este nuevo servicio en el paquete de facilidades que brinda ETECSA agilizará el trabajo de aquellas personas que comiencen a utilizarlo por lo que se define como **problema a resolver**:

- ✓ ¿Cómo prestar servicios de Agenda de Compromisos para telefonía fija en Cuba?

Teniendo como **objeto de estudio**: Servicios de Agendas de Compromisos en el **campo de acción**: Los servicios de agenda telefónica sobre la plataforma de telefonía Asterisk.

El **objetivo general** que se persigue con la realización del presente trabajo es:

- ✓ Desarrollar el servicio de Agenda de Compromiso Telefónica sobre la plataforma Asterisk para la telefonía fija.

Además de contar con los **objetivos específicos** siguientes:

- ✓ Desarrollar una aplicación Web que permita gestionar las solicitudes de avisos de compromisos.
- ✓ Desarrollar una AGI (Asterisk Gateway Interface) que permita grabar los mensajes de compromisos.
- ✓ Desarrollar un Proceso (Demonio) para el envío y lectura de peticiones de los usuarios.

Para dar cumplimiento a los objetivos antes expuestos se proponen las siguientes **tareas de la investigación:**

- ✓ Estudio de aplicaciones similares a la que se desea desarrollar para tomar experiencias e ideas.
- ✓ Estudio del Marco de Trabado CEDRUX con el objetivo de comprender mejor el funcionamiento y trabajo con el mismo.
- ✓ Realización de un estudio de la plataforma telefónica Asterisk y la forma de conexión con aplicaciones web.

- ✓ Estudio de las metodologías y herramientas de desarrollo de software.

Con lo anteriormente planteado se ha llegado a la siguiente **idea a defender:**

El desarrollo del Sistema de Agenda de Compromisos Telefónica (SACT) permitirá brindar el servicio de agenda de compromisos telefónicos sobre la plataforma de telefonía Asterisk.

Los Métodos Científicos se dividen en Filosóficos, Empíricos y Teóricos. Para el establecimiento de una guía de la investigación fueron usados el Empírico y el Teórico.

El Empírico fue utilizado para la recolección de los datos necesarios, dentro de este se utilizaron las técnicas de:

- ✓ Entrevista: a través de la cual se pueden conocer las necesidades del cliente y obtener los requisitos del sistema a tener en cuenta para la solución de la investigación.
- ✓ Medición: está presente en las pruebas realizadas, ya que son una forma de medir si la implementación del software fue satisfactoria.
- ✓ Observación: Este fue utilizado tomar experiencias de aplicaciones similares.

Los Teóricos fueron utilizados para realizar análisis, síntesis y modelaciones, dentro de este se utilizó:

- ✓ El Análisis Histórico-Lógico para realizar un análisis de las tecnologías y herramientas a utilizar.
- ✓ El Analítico–Sintético durante la revisión y la consulta de la bibliografía necesaria para tener un mejor entendimiento del problema a resolver.
- ✓ La Modelación para realizar todos los modelados de la aplicación a desarrollar.
- ✓ Inductivo-deductivo para generalizar o particularizar el conocimiento.
- ✓ Sistémico durante la conexión de la aplicación con la plataforma de telefonía Asterisk.

El presente documento está estructurado con introducción, desarrollo y conclusiones en cada uno de los cinco capítulos con los que cuenta el documento, a continuación se muestra la estructura de cada capítulo:

Capítulo 1: "Fundamentación Teórica": En este capítulo se exponen los principales conceptos asociados al problema, los distintos sistemas de agendas de compromisos existentes a nivel mundial y nacional, las metodologías, los lenguajes de programación y las herramientas utilizadas en el desarrollo de la aplicación.

Capítulo 2: "Exploración y Planificación": En este capítulo se analizaran las características tanto funcionales como no funcionales del sistema a desarrollar, así como un análisis detallado de los artefactos generados durante la fase de exploración y planificación de la metodología de desarrollo XP.

Capítulo 3: "Diseño del Sistema": En este capítulo se exponen las principales características y artefactos del flujo de diseño de la metodología de desarrollo XP.

Capítulo 4: "Implementación y Pruebas": En este capítulo se aborda lo relacionado con los procesos de implementación y pruebas del sistema a desarrollar.

Capítulo 5:"Estudio de la Factibilidad": En este capítulo se realiza un análisis de la factibilidad del sistema a desarrollar.

Capítulo 1: Fundamentación Teórica

Introducción:

En el presente capítulo se tratarán conceptos relacionados con el desarrollo del Sistema de Agenda de Compromisos Telefónica (SACT), así como el estudio del arte realizado a las aplicaciones que prestan servicios de agendas, las metodologías de desarrollo y las herramientas que se utilizarán en el desarrollo de la aplicación.

1.1 Conceptos Relacionados.

Para un mejor entendimiento del problema a resolver es necesario que se expliquen algunos términos importantes cuando de implementar aplicaciones basadas en la telefonía se trata, estos términos son explicados a continuación:

1.1.1 AGI (Asterisk Gateway Interface).

La Interfaz de Red de Asterisk o Asterisk Gateway Interface (AGI) “provee una interfaz estándar para que programas externos puedan controlar el plan de marcación. Los lenguajes más comunes de programación de scripts AGI son: PHP, Python y Perl, aunque se puede utilizar cualquier otro lenguaje.” (3)

Permite enviar parámetros a un programa externo, ejecutarlo y luego regresar a Asterisk e incluso recibir el resultado de dicho programa.

1.1.2 Asterisk.

Este proyecto fue desarrollado por Mark Spencer, miembro fundador de la compañía Digium. Es un programa de software libre que convierte un ordenador en una central telefónica multifuncional o Private Branch Exchange (PBX), ofreciendo las mismas características y servicios que los caros sistemas propietarios PBX. Este software se encarga de integrar funcionalidades de telefonía clásica con nuevas capacidades derivadas de su flexible y potente arquitectura, pudiendo conectar un número determinado de teléfonos para hacer llamadas entre sí.

1.1.3 Proceso (Demonio).

Un demonio es un tipo especial de proceso informático o programa en ejecución, este actúa en un segundo plano sin ser controlado directamente por el usuario. Este tipo de programas trabajan de forma continua y aunque se intente cerrar o terminar el proceso, este continuará o se reiniciará automáticamente.

1.1.4 Sistema Integral de Gestión de Entidades (CEDRUX).

El Sistema Integral de Gestión de Entidades (CEDRUX) es una solución de software cubana desarrollada por la UCI basada en los sistemas de Planeación de Recursos Empresariales o Enterprise Resource Planning (ERP); básicamente está diseñado para incrementar la eficiencia de las empresas mediante un conjunto de módulos adaptables a las necesidades de cada cual.

1.2 Agendas.

Desde tiempos remotos el hombre ha tratado de dejar plasmado sus recuerdos ya sea en las paredes de las cavernas por medio de las pinturas rupestres, con jeroglíficos tallados en piedra o en papel; evidenciando la necesidad del ser humano de apuntar sus actividades diarias, surgiendo con el transcurso de los años un nuevo término para denominar esta forma de almacenar información: “Las agendas tradicionales” que no son más que un libro o cuaderno donde se anotan los datos que una persona desea recordar y luego con la evolución de las tecnologías surge una nueva modalidad de agendas, ”las agendas electrónicas”, estas últimas son dispositivos o programas que permiten almacenar apuntes.

De la misma forma que la humanidad necesitó almacenar los recuerdos también necesitó comunicarse, por lo que la invención del teléfono simplificó la forma de realizarlo ya que este aparato permite a las personas situadas en diferentes puntos comunicarse entre sí mediante la telefonía móvil o la telefonía fija.

La telefonía fija o convencional: hace referencia a las líneas y equipos que se encargan de la comunicación entre terminales telefónicos no portables.

La telefonía móvil o telefonía celular: está formada por dos grandes partes, una red de comunicaciones y los terminales que permiten el acceso a dicha red.

En la actualidad la modalidad de agendas electrónicas y los teléfonos han sido combinados, surgiendo así las agendas telefónicas, estas no son más que programas con un conjunto de funciones las cuales permiten almacenar información en los teléfonos.

1.2.1 Sistemas de Agendas a nivel mundial.

A nivel mundial existen diferentes tipos de sistemas de agendas, estos se ven presentes tanto en la telefonía como en ordenadores, en estos últimos existen diferentes aplicaciones que se encargan de la gestión de alertas como el **ATnotes**, las **tareas programadas** del sistema operativo Windows o la herramienta **Euro Suite Utilities** utilizadas para establecer alarmas de citas o sistemas de avisos programados, permitiendo volver a recordar la notificación transcurridos unos minutos; este tipo de sistemas solo se puede utilizar a nivel de computadoras, trayendo como principal desventaja la necesidad de estar sentado frente al ordenador para recibir el aviso.

Por el lado de la telefonía se cuenta con agendas telefónicas que permiten recordar en un espacio de memoria cierta cantidad de números telefónicos, además existen agendas telefónicas con interfaces web que interaccionan con la plataforma de telefonía Asterisk, entre ellas se puede mencionar **Click to Dial** que es una aplicación usada para establecer llamadas entre dos participantes mediante una interfaz web donde el usuario selecciona el número de teléfono al que desea llamar.

Sin embargo ninguna de las aplicaciones antes mencionada combina una agenda de un ordenador con la telefonía, por lo que la necesidad de buscar alternativas para las personas que no tienen acceso permanente a las computadoras ha motivado a numerosas instituciones a fomentar la combinación de las aplicaciones de los ordenadores con los teléfonos. Un ejemplo de ellas es la Universidad Técnica "Federico Santa María" de Chile la cual propuso en el 2006 la realización de una **Agenda de Compromisos,** aplicación capaz de integrar un ordenador con la plataforma telefónica Asterisk, esta establece un sistema de avisos configurables mediante una web y recibidos en el teléfono.

1.2.2 Sistemas de Agendas en Cuba.

En Cuba ETECSA como su proveedor de servicios telefónicos brinda servicios de telefonía fija y al igual que en el resto del mundo cuenta con agendas que permiten recordar en memoria números telefónicos, sin embargo no cuenta con una aplicación que integre agendas con teléfonos fijos, por lo que se encuentra enfrascada en conjunto con TLM en la mejora continua de los servicios brindados, sumándose a la idea mundial de integrar aplicaciones de ordenadores con los teléfonos fijos en sistemas de agendas personales.

1.3 Metodologías de desarrollo de software.

Las metodologías de desarrollo de software definen Quién, Qué, Cuándo y Cómo hacerlo, haciendo referencia a un conjunto de procedimientos basados en los principios lógicos utilizados para alcanzar los objetivos.

En el proceso de desarrollo de cualquier software es importante la búsqueda de la metodología más adecuada, la que mejor se adapte a las condiciones existentes. La elección de una determinada metodología depende en gran medida de las características del proyecto, el equipo de desarrollo y la complejidad de la solución informática a desarrollar.

1.3.1 Metodologías de desarrollo de Software tradicionales.

Las **metodologías tradicionales o pesadas** ponen gran énfasis en la planificación y un uso exhaustivo de documentación, mediante una rigurosa definición de roles, actividades, artefactos, etc. Por su enfoque

tradicional no se adaptan a los cambios constantes, por lo que no son metodologías adecuadas para proyectos con un entorno de desarrollo cambiante.

1.3.2 Metodologías de desarrollo de Software ágiles.

Las **metodologías ligeras o ágiles** están orientadas a proyectos pequeños donde exista una gran incertidumbre, requisitos desconocidos o variables. El cliente es parte del proceso de desarrollo, lo que posibilita la retroalimentación constante y las respuestas rápidas a los cambios en el negocio debido a su gran capacidad de respuesta a los cambios.

1.3.3 Características de las Metodologías Ágiles:

- ✓ Los trabajadores del equipo de desarrollo son más importantes que los procesos y herramientas de desarrollo.
- ✓ Las funcionalidades del Software son más importante que una documentación exhaustiva.
- ✓ La estrecha relación entre el cliente y el equipo de desarrollo es más importante que la negociación de contratos.
- ✓ Seguir un plan es una cuestión importante, pero la respuesta ante el cambio es más importante que el seguimiento del plan trazado.
- ✓ La prioridad es satisfacer al cliente mediante tempranas y continuas entregas de software.
- ✓ Producir código claro y robusto es la clave para avanzar más rápidamente.

1.3.4 Metodología seleccionada: eXtreme Programming (XP)

Al ser dos personas en el equipo de desarrollo y no contar con un contrato tradicional con requisitos de implementación bien definidos, además de que el cliente es parte del equipo de desarrollo se ha decidido el uso de la metodología ágil Programación Extrema o eXtreme Programming (XP). Esta se encuentra centrada en potenciar las relaciones entre los miembros del equipo de desarrollo como clave fundamental para el éxito, basándose en una retroalimentación continua entre el cliente y el equipo de desarrollo. Utiliza las historias de usuario para especificar los requisitos del software y está especialmente preparada para enfrentar con rapidez los cambios que se originen durante el desarrollo del software.

El ciclo de vida de la metodología de desarrollo XP tiene hace un gran énfasis en el carácter iterativo e incremental del desarrollo de un software, las iteraciones de esta metodología son cortas con el objetivo de entregar versiones del producto al cliente para obtener una mayor retroalimentación con la cual se asegura una mejor calidad a largo plazo. El ciclo de vida ideal consta de seis fases, las cuales son explicadas a continuación:

- ✓ **Exploración:** Los clientes plantean en las historias de usuarios lo que constituye la principal prioridad para la primera versión del producto.
- ✓ **Planificación de la Entrega (*Release*):** el cliente establece la prioridad de cada historia de usuario, datos que son usados por los programadores para realizar una estimación del esfuerzo necesario en cada una de ellas.
- ✓ **Iteraciones:** Incluye varias iteraciones sobre el sistema antes de ser entregado. El plan de entrega está compuesto por iteraciones de no más de tres semanas.
- ✓ **Producción:** En esta fase se realizan pruebas adicionales y revisiones de rendimiento antes de que el sistema sea llevado al entorno del cliente.
- ✓ **Mantenimiento:** Se plantean las tareas de soporte para el cliente. La fase de mantenimiento requiera adicionar personal al equipo y realizar cambios en su estructura.
- ✓ **Muerte del Proyecto:** Esta etapa es de suma importancia pues es cuando el cliente no tiene más historias para ser incluidas en el sistema. Lo que requiere que se satisfagan las necesidades del cliente en otros aspectos como rendimiento y confiabilidad del sistema.

1.4 Sistemas Gestores de Bases de Datos.

Un **Sistema Gestor de Base de Datos (SGBD)** "es un conjunto de programas que permiten crear y mantener Bases de datos, asegurando la integridad, confidencialidad y seguridad de los datos." (4)

Algunas de las características que debe tener un Sistema Gestor de Base de Datos son las siguientes:

- ✓ "Control de la redundancia: la redundancia de datos tiene varios efectos negativos (duplica el trabajo al actualizar, desperdicia espacio en disco, puede provocar inconsistencia de datos) aunque a veces es deseable por cuestiones de rendimiento.
- ✓ Cumplimiento de las restricciones de integridad: el SGBD ha de ofrecer recursos para definir y garantizar el cumplimiento de las restricciones de integridad." (5)

Para el sistema a implementar se ha decidido la utilización de **PostgreSQL** en su versión 8.3 porque es un sistema de gestión de base de datos relacional orientado a objetos de software libre que puede ser utilizado en los principales sistemas operativos: Linux, Unix, BSDs, Mac OS, Beos, Windows.

1.4.1 Ventajas de PostgreSQL.

Existe gran cantidad de documentación organizada, pública y libre, es un gestor altamente adaptable a las necesidades del cliente. Brinda soporte para los lenguajes de programación: PHP, C, C++, Perl, Python. Incluye características de la orientación a objetos, como puede ser la herencia entre tablas (aunque no

entre objetos, ya que no existen), tipos de datos, funciones, restricciones, disparadores y reglas. Permite la gestión de diferentes usuarios y los permisos asignados a estos. Puede ser utilizado, modificado y distribuido por cualquiera gratuitamente.

1.5 Herramienta de base de datos Mapeador de Doctrine.

En la generación de la base de datos y los demás archivos que serán usados durante el desarrollo de la aplicación SACT, se utilizará el software de mapeo de bases de datos Mapeador de Doctrine, este permitirá ahorrar tiempo y esfuerzos en la realización del mapeo de la base de datos. Esta herramienta se conecta a la base de datos y permite el mapeo de todas las tablas de la misma. Su única desventaja se encuentra centrada en que no es capaz de reconocer las relaciones entre las clases, por lo que es necesario redefinírselas para que el mapeo tenga la calidad requerida.

1.6 Lenguajes de programación WEB.

La necesidad de diferentes plataformas y el desarrollo de las tecnologías trajeron consigo la aparición de diferentes lenguajes de programación web.

Para el desarrollo de la aplicación se propone la utilización de tecnología Web, ya que esta tecnología posee ventajas tales como:

- ✓ Se necesita solamente un navegador web y conexión al sistema.
- ✓ Se pueden ejecutar varios clientes simultáneamente.
- ✓ Se necesitan pocos recursos para que la aplicación trabaje correctamente.
- ✓ Toda la información se encontrará de forma centralizada en el servidor.

En el desarrollo de la aplicación SACT se ha decidido la utilización del lenguaje de programación **Hypertext Pre-Processor (PHP)** del lado del servidor, mientras que del lado del cliente se propone utilizar los lenguajes **HyperText Markup Language (HTML)** y **Java script.**

1.6.1 Lenguaje HTML (HyperText Markup Language, Lenguaje de Marcas de Hipertexto).

Es un lenguaje estático para el desarrollo de sitios web que permite describir hipertexto presentando el texto de forma estructurada y agradable.

Ventajas:

- ✓ Es admitido por todos los navegadores.
- ✓ Cuenta con archivos pequeños.
- ✓ Despliegue rápido.

- ✓ Lenguaje de fácil aprendizaje.

Desventajas:

- ✓ Lenguaje estático de diseño lento.
- ✓ Etiquetas limitadas.
- ✓ La interpretación en cada navegador puede ser diferente.
- ✓ Guarda muchas etiquetas que pueden dificultar la corrección.

1.6.2 Lenguaje Java script.

"Es un lenguaje interpretado que no requiere de compilación. Es similar a Java, aunque este lenguaje no es orientado a objetos, ni dispone de herencias" (6). Es interpretado por la mayoría de los navegadores y puede ser integrado dentro de las páginas web.

Ventajas:

- ✓ Es seguro y fiable.
- ✓ Cuenta con scripts de capacidades limitadas por razones de seguridad.
- ✓ Se ejecuta en el cliente.

Desventajas:

- ✓ El en algunos casos el código puede ser visible por cualquier usuario, pudiendo ser descargado completamente.

1.6.3 Lenguaje PHP (PHP Hypertext Pre-processo).

"Es un lenguaje de script interpretado en el lado del servidor utilizado para la generación de páginas web dinámicas, embebidas en páginas HTML y ejecutadas en el servidor." (7) Este lenguaje no necesita ser compilado para ejecutarse.

Ventajas:

- ✓ Se caracteriza por ser un lenguaje muy rápido.
- ✓ Soporta en cierta medida la orientación a objeto: clases y herencia.
- ✓ Es un lenguaje multiplataforma.
- ✓ Capacidad de conexión con la mayoría de los gestores de base de datos: MySQL, PostgreSQL, Oracle, MS SQL Server.
- ✓ Capacidad de expandir su potencial utilizando módulos.

- ✓ Posee documentación en su página oficial la cual incluye descripción y ejemplos de cada una de sus funciones.
- ✓ Es libre, por lo que es de fácil acceso para todos.
- ✓ Incluye gran cantidad de funciones.
- ✓ No requiere definición de tipos de variables ni manejo detallado del bajo nivel.

Desventajas:

- ✓ Todo el trabajo lo realiza el servidor y no delega al cliente. Por tanto puede ser más ineficiente a medida que las solicitudes aumenten de número.
- ✓ La legibilidad del código puede verse afectada al mezclar sentencias HTML y PHP.

1.7 Servidores para aplicaciones web.

Un servidor web "es un programa que se ejecuta en un ordenador manteniéndose a la espera de peticiones por parte de un cliente, además sirve de contenido estático a un navegador cargando un archivo y sirviéndolo a través de la red al navegador; este intercambio es mediado por el navegador y el servidor que se comunican mediante el protocolo HTTP." (8)

Para el desarrollo de la aplicación se propone la utilización del servidor web Apache ya que "es un servidor de páginas web de código fuente abierto, altamente configurable y flexible" (9)

1.7.1 Características de Apache:

- ✓ Permite la restricción a determinados sitios web.
- ✓ Corre en una multitud de Sistemas Operativos.
- ✓ Es un servidor altamente configurable de diseño modular. Por lo que puede ser adaptado a diferentes entornos y necesidades.
- ✓ Apache permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor.

1.8 Framework.

"Los Framework ayudan en el desarrollo de software proporcionando una estructura definida y creando aplicaciones con mayor rapidez, además son desarrollados con el objetivo de brindar una mejor organización." (10)

Un framework permite separar en capas la aplicación:

- ✓ La lógica de presentación que administra las interacciones entre el usuario y el software.

- ✓ La lógica de dominio o de negocio, que manipula los modelos de datos de acuerdo a los comandos recibidos desde la presentación.
- ✓ La lógica de datos que permite el acceso a un agente de almacenamiento persistente u otros.

Un framework PHP proporciona la estructura necesaria como para que un proyecto de software basado en PHP pueda reutilizarla y obtener provecho de las librerías o funcionalidades que se han desarrollado.

El framework seleccionado es CEDRUX que es un ambiente de trabajo que está integrado por diferentes framework como son ExtJS, Doctrine, Zend Framework y UCID, este último desarrollado por la Unidad de Compatibilidad Integrada de Desarrollo (UCID) que es el encargado de trabajar con las vistas y está integrado por ExtJS.

ExtJS es una librería Java script que permite construir aplicaciones web complejas utilizando componentes predefinidos brindando facilidades como la consistencia sobre cualquier navegador; la reutilización de código y una orientación a la programación de interfaces de tipo desktop en la web, además de poseer un sistema dual de licencias: comercial y código abierto.

Doctrine que es el encargado del acceso a la base de datos, debido a que "es un potente y completo sistema ORM (Object Relational Mapper) para PHP 5.2+ que incorpora una DBL (capa de abstracción a base de datos) y es usado por la habilidad de escribir opcionalmente las preguntas de la base de datos orientado a objeto, esto les proporciona una alternativa poderosa a diseñadores de SQL que mantiene un máximo de flexibilidad sin requerir la duplicación del código innecesario". (11)

La utilidad que aporta Zend Framework es que además de trabajar con modelo vista controlador, "cuenta con módulos para manejar archivos PDF y canales de RSS, trabaja con PHP5 y a su vez estandariza los procesos más frecuentes, dotándolos de gran robustez, facilita el mantenimiento de las aplicaciones y tiene el respaldo de la propia ZEND, creadora de PHP, lo que asegura su continuidad futura tanto como la del propio lenguaje PHP". (12)

El hecho de que se beneficie de estas importantes características con que cuentan cada uno de los diferentes Framework que lo integran lo hace un ambiente de trabajo ideal para el desarrollo de aplicaciones web.

1.9 IDE de Desarrollo.

Un entorno integrado de desarrollo o Integrated Development Environment (IDE) "es editor de código compuesto por un conjunto de herramientas de programación, este puede ser una aplicación por si sola o pueden ser parte de aplicaciones existentes" (13); los IDEs proveen un marco de trabajo amigable para que el desarrollo de cualquier aplicación sea factible.

Para el desarrollo de la aplicación se ha decidido utilizar el **Zend Studio Neon,** este IDE de desarrollo de aplicaciones web "es un editor de texto para páginas PHP que proporciona un buen número de ayudas desde la creación y gestión de proyectos hasta la depuración del código" (14), además está basado en Eclipse PHP Development Tools (PDT) que es una extensión de Eclipse. Posee una inserción automática de paréntesis y corchetes de cierre, detectando errores de sintaxis en tiempo real.

1.9.1 Ventajas de Zend Studio Neon

- ✓ Agiliza el trabajo de desarrollo.
- ✓ Cuenta con un buen depurador.
- ✓ Posee opciones que permiten un desarrollo profesional de aplicaciones.

1.10 Control de Versiones.

Se llama control de versiones a los distintos cambios que se realizan sobre los elementos de algún fichero y son almacenados en un repositorio central. Este repositorio es como un servidor de ficheros ordinario, salvo que recuerda todos los cambios que se han realizado a los ficheros y directorios, permitiendo la recuperación de versiones anteriores de los ficheros.

Como cliente para el control de versiones en el desarrollo de la aplicación SACT se ha decidido utilizar TortoiseSVN que "es un cliente gratuito de código abierto para el control de versiones, capaz de manejar ficheros y directorios" (15)

1.11 Conclusiones parciales:

Con la realización de este capítulo se adquirió experiencia e ideas del funcionamiento de las agendas telefónicas y las diferentes aplicaciones existentes en el mundo, tomándolas como referencia para poder brindar este servicio en la telefonía fija, además con el estudio del arte se fue capaz de seleccionar las metodologías y herramientas más se adecúan a las necesidades de la aplicación que se desea desarrollar, primando el concepto de software libre durante la realización de la búsqueda y los criterios de selección.

Capítulo 2: Exploración y Planificación.

Introducción:

En este capítulo se abordarán los temas relacionados con la fase de exploración y planificación de la metodología de desarrollo XP, así como las características con que va a contar el sistema a implementar y su arquitectura, además se tratarán los principales artefactos generados y la planificación del tiempo y el esfuerzo de las fases posteriores.

2.1 Objeto de Automatización.

El uso de agendas tradicionales es un proceso que debe ser automatizado ya que resulta engorrosa la manera en que se tratan los compromisos de la forma tradicional. Traer consigo en todo momento la agenda propicia que, ante cualquier descuido será expuesta a situaciones especiales como por ejemplo: que caiga en manos indebidas, se extravíe o en el menor de los casos se olvide el compromiso anotado. Para tratar de erradicar los inconvenientes de las agendas tradicionales se desea desarrollar una aplicación que permita a los usuarios de las agendas un estricto control de la planificación de los compromisos y que se libere de la tensión que implica planificarlos en agendas tradicionales.

2.2 Propuesta del Sistema.

El Sistema de Agendas de Compromisos Telefónica (SACT) es un sistema de soluciones informáticas que se encargará de lo referente a las agendas de compromisos telefónicas. Se propone una aplicación web a la que podrá acceder el cliente y el administrador, este último será el encargado de manejar los permisos de los usuarios que harán uso de esta aplicación. Esto se garantizará mediante el uso del módulo de seguridad que tiene implementado el CEDRUX, el cual trae incorporado las funcionalidades gestionar usuario y gestionar roles de usuario. El cliente podrá realizar las peticiones, modificaciones o cancelaciones de compromisos mediante la web. El sistema será el encargado mediante un proceso (Demonio) de realizar las peticiones de las solicitudes de compromiso a la base de datos, estas peticiones se realizarán cada un minuto. La AGI será la encargada de grabar las notificaciones de compromisos; estas notificaciones no son más que un mensaje que recibirá el cliente en una fecha especificada; de esta forma se dará solución al inconveniente de no tener agendas telefónicas para telefonía fija, facilitando a los usuarios del SACT todo el trabajo relacionado con los compromisos.

2.3 Características no funcionales del sistema.

Las características no funcionales de una aplicación son muy importantes ya que son las cualidades que todo sistema debe poseer y determinan la arquitectura de cualquier software a desarrollar, para la realización de la aplicación propuesta se han identificado las siguientes características no funcionales.

2.3.1 Apariencia o interfaz externa.

La interfaz gráfica debe ser amigable al usuario con un diseño sencillo que permita realizar cualquier tipo de interacción con el sistema de manera fácil, para garantizar esto se propone el uso de una sola interfaz para el trabajo de los compromisos, además en la interfaz se deben mostrar mensajes con información al usuario, estos mensajes pueden ser: alertas, errores o ayuda.

2.3.2 Usabilidad

La aplicación propuesta será usada por personas que pueden o no tener conocimientos básicos de informática, por lo que el sistema debe ser de fácil uso. Para un mejor entendimiento del funcionamiento de la aplicación se entregará una documentación completa de la aplicación, la cual incluye manuales de usuario y ayuda.

2.3.3 Rendimiento

Para un mejor funcionamiento se seguirán técnicas de elaboración de sitios web como el equilibrio de proporciones y escalas, la agrupación de elementos, la alineación de diseño y la compatibilidad multinavegador. El rendimiento del SACT estará dado en gran medida por el aprovechamiento de los recursos; la propuesta debe ser rápida con un tiempo de respuesta que deberá estar entre los 6 y 10 segundos.

2.3.4 Portabilidad

La herramienta podrá ser usada bajo cualquier distribución de Linux o Windows. El servidor web, el servidor de base de datos y el de Asterisk deberán conformar una arquitectura distribuida con los servidores en máquinas diferentes garantizado un mejor funcionamiento de los servidores al no estar sobrecargados y facilitar el mantenimiento.

2.3.5 Seguridad

Confiabilidad: La información manejada por el sistema debe estar protegida de acceso no autorizado por lo cual se establecerá un nivel de acceso a la aplicación mediante roles.

Integridad: La información manejada por el sistema debe ser objeto de una cuidadosa protección contra la corrupción y estados de inconsistencia por lo que se necesita que los accesos a la aplicación y los recursos sean controlados por el servidor, esto se garantizará con el uso de una arquitectura modular.

Disponibilidad: La aplicación deberá estar disponible en todo momento para aquellas personas con acceso, por lo que las funciones y responsabilidades estarán distribuidas, haciéndose uso de un sistema distribuido con servidores independientes. Facilitando de esta forma el mantenimiento, actualización y traslado de los mismos sin incurrir en afectaciones a la disponibilidad de la aplicación

2.3.6 Software

En las computadoras de los usuarios solo se requiere de un navegador Web Mozilla Firefox en una versión igual o superior a 3.0 u otro que brinde soporte para DOM 2.0 y Java Script, este último debe de estar habilitado en todos los navegadores, además de cualquier distribución de Linux o Windows. En el caso del servidor de aplicaciones web se deberá tener instalado cualquier versión igual o superior al Apache 2.0 y en el servidor de base de datos se debe tener instalado PostgreSQL 8.3.

2.3.7 Hardware

En el cliente se requiere una máquina con 256 MB de RAM como mínimo, un procesador igual o mayor a 1.40 GHZ y una tarjeta de red, todos los servidores deben tener 1 GB de RAM, 160 GB de disco duro mínimo, además deben tener UPS, un lector de DVD o CD y una tarjeta de red.

2.4 Arquitectura

La arquitectura no es más que una vista estructural de alto nivel que ocurre muy tempranamente en el ciclo de vida y define los estilos o grupos de estilos adecuados para cumplir con las características no funcionales de un software. Luego de analizar las diferentes características no funcionales del sistema se propone para el desarrollo de la aplicación una arquitectura cliente servidor, pues esta al ser una arquitectura distribuida permitirá tener la aplicación dividida en módulos garantizando que no exista una sobrecarga en los servidores.

2.4.1 Arquitectura Cliente Servidor.

La arquitectura Cliente/Servidor al ser una extensión de programación modular que separa los módulos del software, brindando una mayor usabilidad, flexibilidad, interoperabilidad y escalabilidad en las comunicaciones.

La utilización de las diferentes aplicaciones o servicios de Internet se lleva a cabo respondiendo a este modelo, que es una forma de especializar terminales y programas para que las actividades y tareas se ejecuten con la mayor eficiencia posible. Mediante esta arquitectura el usuario puede acceder a la información sin tener en cuenta su ubicación física y donde pueda estar alojada la misma.

2.4.1.1 Ventajas de la arquitectura Cliente/Servidor:

- ✓ "La arquitectura Cliente/Servidor facilitará la integración entre sistemas diferentes permitiendo integrar ordenadores con sistemas medianos y grandes, sin necesidad de que todos tengan que utilizar el mismo sistema operativo.
- ✓ La estructura modular facilitará la integración de nuevas tecnologías y el crecimiento de la infraestructura, favoreciendo así la escalabilidad de la solución.
- ✓ La centralización del control permitirá que los accesos, recursos y la integridad de los datos sean controlados por el servidor de forma que un programa cliente defectuoso o no autorizado no pueda dañar el sistema.
- ✓ Al estar distribuidas las funciones y responsabilidades entre varios ordenadores independientes, es posible reemplazar, reparar, actualizar o incluso trasladar un servidor sin que los clientes se vean afectados por el cambio o la afectación sea mínima." (16)

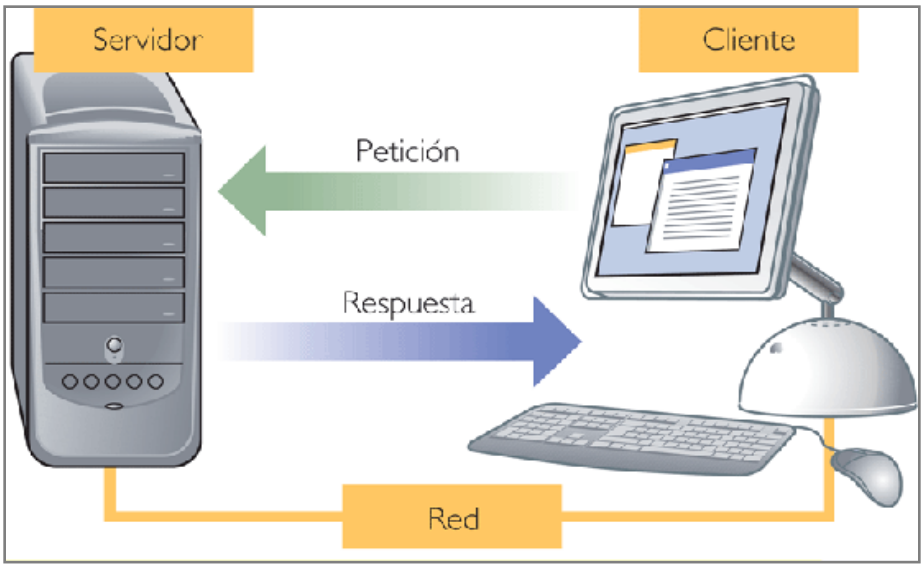


Figura 1 Arquitectura Cliente Servidor.

2.4.2 Modelo Vista Controlador (MVC).

Para la solución de la aplicación se propone la utilización del patrón arquitectónico Modelo Vista Controlador (MVC). El MVC es aplicable al desarrollo de cualquier aplicación independientemente del lenguaje de programación elegido y separa los datos, la interfaz de usuario y la lógica de control en tres componentes distintos:

La capa modelo es la representación específica de la información con la cual se operará en el sistema, siendo la encargada de todo el acceso a los datos y las funciones; llamada también lógica de negocio. Esta capa lleva un registro de cada acceso a los datos, poniéndolo en una función individual para garantizar que si cambia el gestor de bases de datos solo se afecte a este tipo de funciones y no al resto de la aplicación, maximizando de esta forma la adaptabilidad a los cambios de la aplicación.

La capa Vista es la encargada de la interacción con el usuario, mostrando la información de la capa modelo al cliente. Al tener la capa vista separada de la capa controladora se pueden realizar cambios en esta sin tener que tocar nada más que una parte delimitada de código.

La capa controlador es la encargada de la unión de las capas vista y modelo, esta capa es la que escucha los cambios en la vista y se los envía al modelo, el cual le regresa los datos a la vista.

El framework utilizado propone una abstracción entre las capas modelo y vista, siendo la capa controladora la encargada de establecer la comunicación entre ellas.

Figura 2 Modelo Vista Controlador

2.4.2.1 Ventajas del uso del MVC:

La ventaja fundamental de esta división en capas está dada en la medida en que cada una de estas capas puede ser sustituida sin afectar a las otras, ya que provee una separación total entre la lógica del negocio y presentación, además pueden existir diferentes vistas para un mismo modelo trayendo como resultado que la división de código de este estilo arquitectónico haga más fácil la portabilidad y la adaptación a los requerimientos del usuario. Al existir la separación de vistas, controladores y modelos es más sencillo realizar labores de mejora como:

- ✓ Agregar nuevas vistas.
- ✓ Agregar nuevas formas de recolectar las órdenes del usuario.
- ✓ Modificar los objetos de negocios bien sea para mejorar el rendimiento o para migrar la tecnología.
- ✓ Las labores de mantenimiento simplificadas.
- ✓ Las correcciones solo se deben hacer en un solo lugar.

2.4.3 Arquitectura propuesta.

La arquitectura propia del SACT se basada en la arquitectura cliente servidor con el estilo Modelo Vista Controlador (MVC), donde un usuario sentado frente a un ordenador accede mediante la interfaz web de la aplicación SACT donde realizará peticiones a un servidor de aplicaciones web apache, este servidor de aplicaciones se conectará con el servidor de base de datos de forma que almacenará las solicitudes de compromisos efectuadas por el cliente, como parte de las solicitudes que el cliente desea realizar se conectará con el servidor Asterisk para grabar los mensajes personalizados del usuario. Entre el servidor Asterisk y la base de datos existirá una aplicación encargada de monitorear los cambios en la base de datos e informar a Asterisk de estos, para que este servidor pueda avisar al cliente mediante el teléfono de la solicitud de compromiso.

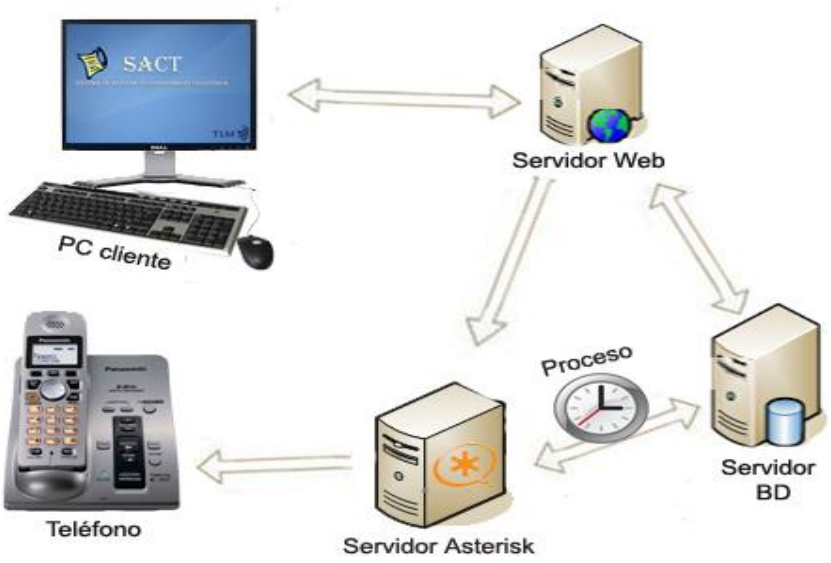


Figura 3 Arquitectura propuesta

2.5 Fase de Exploración.

La metodología de desarrollo XP comienza con la fase de exploración, en esta fase el cliente define lo que necesita mediante las historias de usuario, mientras que el equipo de desarrollo se familiariza con las

herramientas y tecnologías que han sido seleccionadas para el desarrollo de la aplicación, además es en esta fase donde los programadores estiman el tiempo de desarrollo.

2.5.1 Flujo de procesos del sistema propuesto.

El usuario se autentica en la página web donde se verifica que los datos introducidos sean correctos, después de realizada esta operación podrá realizar las solicitudes, modificaciones y cancelaciones de los compromisos que estime convenientes, este proceso será registrado en la base de datos. Cuando un compromiso es creado, se enviará a la hora y la fecha especificada un mensaje de aviso previamente escogido por el solicitante, la persona que lo recibe será la encargada de enviar una confirmación al sistema para poner fin al aviso, en el caso que no ser recibida se repetirá el mensaje cada un tiempo determinado, transcurrido este tiempo si no se ha recibido la confirmación por parte del usuario, el mensaje dejará de ser enviado.

2.5.2 Involucrados en el sistema.

Se definen como involucrados en el sistema todos aquellos que realizan una función o interactúan con él de una forma u otra.

Tabla 1 Involucrados en el sistema.

Personas Relacionadas con el Sistema	Justificación.
Cliente	Es la persona que se encarga de solicitar, modificar y eliminar las solicitudes de compromiso.
Sistema	Es el encargado de almacenar los datos y enviar, repetir y terminar las solicitudes de compromiso.

2.5.3 Historias de Usuario.

Las Historias de Usuario son la forma en que se especifican en XP los requisitos funcionales del sistema, realizándose una por cada característica principal del sistema; tienen el mismo propósito que los casos de uso en las metodologías de desarrollo de software pesadas, aunque no son lo mismo ya que son escritas por los propios clientes desde su perspectiva del sistema, por lo que serán descripciones cortas y escritas en el lenguaje del usuario. Durante la fase de exploración se identificaron 7 historias de usuario para la realización de la aplicación SACT, las cuales se describen a continuación:

Tabla 2 Historia de usuario: Adicionar Solicitud de Compromiso.

Historia de Usuario

Número: 1	**Usuario:** Cliente
Nombre historia: Adicionar Solicitud de Compromiso.	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alto
Puntos estimados: 1	**Iteración asignada:** 1
Programador responsable: Yohandys Casañas - Yeisell G. Medina	
Descripción: El usuario introducirá en la aplicación web SACT los datos de la solicitud (número de teléfono, hora de ejecución, fecha de ejecución, la planificación (semanal, mensual, para una hora específica o con una frecuencia de ejecución determinada), el mensaje y una breve descripción del compromiso).	
Observaciones: Para que el compromiso se adicione satisfactoriamente el usuario deberá grabar un mensaje o escoger uno de los existentes.	

Tabla 3 Historia de usuario: Modificar Solicitud de Compromiso.

Historia de Usuario	
Número: 2	**Usuario:** Cliente
Nombre historia: Modificar Solicitud de Compromiso.	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Medio
Puntos estimados: 1	**Iteración asignada:** 1
Programador responsable: Yohandys Casañas - Yeisell G. Medina	
Descripción: El usuario tendrá la posibilidad de modificar los datos de la solicitud (número de teléfono, hora de ejecución, fecha de ejecución, la planificación (semanal, mensual, para una hora específica o con una frecuencia de ejecución determinada), el mensaje y una breve descripción del compromiso).	
Observaciones:	

Tabla 4 Historia de usuario: Cancelar Solicitud de Compromiso.

Historia de Usuario	
Número: 3	**Usuario:** Cliente
Nombre historia: Cancelar Solicitud de Compromiso.	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Medio
Puntos estimados: 1	**Iteración asignada:** 1

Programador responsable: Yohandys Casañas - Yeisell G. Medina
Descripción: En el caso de que el usuario no desee recibir su solicitud de compromiso por cualquier motivo, este podrá cancelar la solicitud realizada en la aplicación web.
Observaciones:

Tabla 5 Historia de usuario: Grabar Mensaje del Usuario.

Historia de Usuario	
Número: 4	**Usuario:** Cliente
Nombre historia: Grabar Mensaje del Usuario	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Medio
Puntos estimados: 4/7	**Iteración asignada:** 2
Programador responsable: Yohandys Casañas - Yeisell G. Medina	
Descripción: El usuario podrá grabar el mensaje que desee escuchar cuando sea avisado del compromiso.	
Observaciones: Esta grabación se realiza como parte de la solicitud de compromiso.	

Tabla 6 Historia de usuario: Enviar Mensaje de Compromiso.

Historia de Usuario	
Número: 5	**Usuario:** Sistema
Nombre historia: Enviar Mensaje de Compromiso	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Alta
Puntos estimados: 1	**Iteración asignada:** 2
Programador responsable: Yohandys Casañas - Yeisell G. Medina	
Descripción: El sistema leerá los datos de la base de datos (número de teléfono, hora de ejecución, fecha de ejecución, la planificación (semanal, mensual, para una hora específica o con una frecuencia de ejecución determinada), el mensaje y una breve descripción del compromiso) y enviará un mensaje al usuario, este mensaje será un sonido que escuchará en el teléfono.	
Observaciones:	

Tabla 7 Historia de usuario: Repetir Mensaje de Compromiso.

Historia de Usuario	
Número:6	**Usuario:** Sistema
Nombre historia: Repetir Mensaje de compromiso	
Prioridad en negocio: Medio	**Riesgo en desarrollo:** Medio
Puntos estimados: 1	**Iteración asignada:** 3
Programador responsable: Yohandys Casañas - Yeisell G. Medina	
Descripción: En este caso, el mensaje se repetirá una vez cada sesenta segundos para un total de tres repeticiones en ciento ochenta segundos, esta acción se realizará con el objetivo de buscar una confirmación.	
Observaciones:	

Tabla 8 Historia de usuario: Terminar Mensaje de Compromiso.

Historia de Usuario	
Número: 7	**Usuario:** Cliente
Nombre historia: Terminar Mensaje de Compromiso	
Prioridad en negocio: Bajo	**Riesgo en desarrollo:** Medio
Puntos estimados: 1	**Iteración asignada:** 3
Programador responsable: Yohandys Casañas - Yeisell G. Medina	
Descripción: El mensaje terminará por dos razones: 1- El usuario escuchó el mensaje 2- El mensaje se repitió tres veces en ciento ochenta segundos.	
Observaciones:	

2.6 Planificación

La fase de planificación de la metodología ágil XP como su nombre lo indica se encarga de planificar el proceso de desarrollo de una aplicación de software, en esta se realiza una estimación del esfuerzo por historia de usuario. Para la realización de la estimación en XP las métricas son libres, por lo que puede utilizarse cualquier criterio para medir el desempeño de un proyecto, aunque la métrica más utilizada en

este tipo de metodología es la medida por puntos de estimación; un punto de estimación es considerado como una semana de trabajo.

2.6.1 Estimación de esfuerzo por historias de usuarios.

Para el desarrollo de la aplicación propuesta se he realizado una estimación de esfuerzo por cada una de las historias de usuario identificadas, resultados que se muestran a continuación:

Tabla 9 Estimación del esfuerzo por HU.

Historia de Usuario	Puntos de estimación
Adicionar Solicitud de Compromiso	1
Cancelar Solicitud de Compromiso	1
Modificar Solicitud de Compromiso	1
Grabar Mensaje del Usuario	4/7
Enviar Mensaje de Compromiso	1
Repetir Mensaje de Compromiso	1
Terminar Mensaje de Compromiso	1

2.6.2 Plan de Iteraciones.

Después de ser identificadas las historias de usuario y estimado el esfuerzo dedicado a la realización de cada una de estas historias de usuario se procede a la realización de la planificación de la etapa de implementación de SACT. Para un mejor desempeño del equipo de desarrollo se ha establecido una división de la implementación en 3 iteraciones:

2.6.3 Iteración 1

La iteración 1 será la encargada de la implementación de las historias de usuario 1, 2 y 3, las cuales son las de mayor prioridad para el negocio. Al finalizar esta iteración se contará con la primera versión de prueba de la aplicación web SACT.

2.6.4 Iteración 2

La iteración 2 tendrá como principal objetivo la implementación de las historias de usuario 4 y 5 que serán las encargadas de todo lo referente al trabajo con el Proceso(Demonio) y la AGI, al culminar esta iteración se contará con una versión de prueba lista para su integración con la aplicación web SACT.

2.6.5 Iteración 3

La iteración 3 por ser la última tendrá como tarea la implementación de las historias de usuario 6 y 7 que son las encargadas de complementar el resultado de lo implementado en las iteraciones anteriores. Al

terminar esta iteración se contará con una versión 1.0 del producto final y como resultado de esta, el sistema se pondrá en funcionamiento para evaluar su desempeño.

2.6.6 Plan de duración de las iteraciones.

El plan de duración de las iteraciones es el encargado de mostrar las historias de usuario que serán implementadas en cada una de las iteraciones, así como la duración estimada y el orden de implementación de cada una de ellas.

Tabla 10 Plan de duración de iteraciones.

Iteración	Orden de la Historias de usuario a implementar	Duración total
1	Adicionar Solicitud de Compromiso Cancelar Solicitud de Compromiso Modificar Solicitud de Compromiso	3 semanas
2	Grabar Mensaje del Usuario Enviar Mensaje de Compromiso	2 semanas
3	Repetir Mensaje de Compromiso Terminar Mensaje de Compromiso	2 semanas

2.6.7 Plan de entregas.

En este epígrafe se presenta el plan de entregas estimado para la fase de implementación, como producto del presente plan se realizarán versiones entregables del sistema al finalizar cada iteración.

Tabla 11 Plan de entregas

Módulo	Final de la Iteración 1 (17 de marzo del 2011)	Final de la Iteración 2 (28 de abril del 2011)	Final de la Iteración 3 (25 de mayo del 2011)
SACT	Web v0.1	Proceso v0.1 AGI v0.1	SACT v1.0

2.7 Conclusiones parciales:

En el presente capítulo se propuso la arquitectura de la aplicación a desarrollar, esta propuesta se basó en las características no funcionales requeridas, además fueron identificadas las principales funcionalidades a desarrollar, las cuales quedaron plasmadas en las historias de usuario y para una mejor organización del software se realizó el plan de iteraciones por el cual se regirá el proceso de desarrollo del mismo.

Capítulo 3: Diseño del Sistema

Introducción:

En la metodología de desarrollo XP el diseño debe ser una solución simple que pueda funcionar y ser implementada con la mayor rapidez posible; en este capítulo se abordará el diseño del sistema y los diferentes artefactos generados durante este.

3.1 Patrones de Diseño.

Un patrón de diseño es una descripción de clases y objetos comunicándose entre sí adaptada para resolver un problema de diseño general en un contexto particular, el cual es el encargado de identificar clases, instancias, roles, colaboraciones y la distribución de responsabilidades. El conocimiento de los patrones de diseño implementados por CEDRUX es vital para entender el funcionamiento del mismo y por ende lograr los objetivos trazados.

3.1.1 Patrones GOF (Gang of Four).

Los patrones GOF son aquellos que abstraen el proceso de creación de instancias, se ocupan de componer estructuras de mayor tamaño y asignan responsabilidades entre objetos. A continuación se explican los patrones GOF que implementa el CEDRUX:

- ✓ **Decorator:** Como complemento del CEDRUX el Zend Framework implementa este patrón, el cual es el encargado de asignarle responsabilidades a objetos de manera dinámica y añadir funcionalidades de manera dinámica.
- ✓ **Singleton:** Garantiza la existencia de una única instancia para una clase y la creación de un mecanismo de acceso global a dicha instancia. Zend Framework posee una instancia única del controlador frontal disponible mediante este patrón para lograr una vía de entrada única a las solicitudes.
- ✓ **Facade:** Permite utilizar una interfaz común para un conjunto de interfaces del módulo, haciendo que el mismo sea más fácil de usar.
- ✓ **Factory:** Proporciona una interfaz para la creación de objetos interdependientes o interrelacionados, sin especificar sus clases concretas. Zend Framework utiliza este patrón en la implementación de un conjunto de clases para el acceso a datos.

3.1.2 Patrones para Asignar Responsabilidades (GRASP).

Los Patrones Generales de Software para Asignar Responsabilidades o General Responsibility Assignment Software Patterns (GRASP) son aquellos que describen los principios fundamentales de la asignación de responsabilidades a objetos en una aplicación de software, a continuación se explican los patrones de asignación de responsabilidades que implementa el CEDRUX:

- ✓ **Creador:** Se aplica en la clase controladora para la asignación de responsabilidades a las clases relacionadas con la creación de objetos, de forma tal que una instancia de un objeto sólo pueda ser creada por el objeto que contiene la información necesaria para ello.
- ✓ **Experto:** Se aplica para la asignación de responsabilidades a las clases de forma tal que las mismas contengan la información necesaria para poder ejecutar una acción específica. El uso de este patrón permitirá a los objetos valerse de su propia información para hacer lo que se les pide, favorece la existencia de mínimas relaciones entre las clases, lo que permite contar con un sistema robusto y fácil de mantener.
- ✓ **Bajo Acoplamiento:** El acoplamiento mide la fuerza con que una clase está conectada a otra, de esta forma una clase con bajo acoplamiento debe tener un número mínimo de dependencia con otras clases. Las diferentes clases controladoras sólo dependen de un único controlador frontal para realizar sus funcionalidades. Este patrón se tuvo presente debido a la importancia que se le atribuye a realizar un diseño de clases independientes que puedan soportar los cambios de una manera fácil y que a su vez permitan la reutilización.
- ✓ **Alta cohesión:** Se aplica para realizar un diseño que evite contener clases con un alto grado de abstracción, que asuman responsabilidades que podían haber delegado a otros objetos o que tengan responsabilidades muy complejas. Se tienen las clases controladoras que se encargan de ejecutar acciones de acuerdo a las peticiones que le llegan y las clases de acceso a datos que interactúan con el modelo, de forma tal que se elimina la sobrecarga de funcionalidades en las clases controladoras.
- ✓ **Controlador:** Se aplica para realizar las asignaciones en cuanto al manejo de los eventos del sistema y definir sus operaciones.

3.2 Diagramas de clases.

La metodología XP para el diseño de las aplicaciones no requiere la representación del sistema mediante diagramas de clases utilizando notación UML. En su lugar se usan variantes como las tarjetas Cargo o Clase, Responsabilidad y Colaboración (CRC), siendo estas una extensión informal a UML.

3.2.1 Tarjetas Cargo o Clase, Responsabilidad y Colaboración (CRC).

Esta técnica se usa para guiar el sistema a través de análisis donde las clases se examinan, se filtran y se refinan en base a sus responsabilidades con respecto al sistema. El nombre de la clase se coloca a modo de título en la tarjeta, las responsabilidades se colocan a la izquierda, y las clases que se implican en cada responsabilidad a la derecha, en la misma línea que su requerimiento correspondiente. Las tarjetas determinan el comportamiento de cada actividad, durante el diseño de la aplicación a desarrollar fueron identificadas 9 tarjetas CRC las cuales son mostradas a continuación:

Tabla 12 Tarjeta CRC Clase: Vista.

Clase: Vista	
Responsabilidad	**Colaboración**
Enviar datos del Compromiso	AgendaController

Tabla 13 Tarjeta CRC Clase: NomCompromiso.

Clase: NomCompromiso	
Responsabilidad	**Colaboración**
Adicionar Compromiso Modificar Compromiso Eliminar Compromiso	NomMensaje NomPlanificacion NomCompromisoModel

Tabla 14 Tarjeta CRC Clase: NomPlanificación.

Clase:NomPlanificación	
Responsabilidad	**Colaboración**
Adicionar Planificación Modificar Planificación Eliminar Planificación	NomPlanSemanal NomPlanMensual NomPlanFrecuencia NomPlanificacionModel

Tabla 15 Tarjeta CRC Clase: NomUsuario.

Clase: NomUsuario	
Responsabilidad	**Colaboración**
Adicionar Usuario	NomUsuarioModel

Tabla 16 Tarjeta CRC Clase: NomPlanSemana

Clase: NomPlanSemana	
Responsabilidad	**Colaboración**
Adicionar Plan Semana Modificar Plan Semana Eliminar Plan Semana	NomPlanSemanaModel,

Tabla 17 Tarjeta CRC Clase: AgendaController.

Clase: AgendaController	
Responsabilidad	**Colaboración**
Adicionar Solicitud de Compromiso Modificar Solicitud de Compromiso Eliminar Solicitud de Compromiso	Vista NomCompromiso NomMensaje NomPlanificacion NomPlanSemanal NomPlanMensual NomPlanFrecuencia NomUsuario

Tabla 18 Tarjeta CRC Clase: NomPlanFrecuencia.

Clase: NomPlanFrecuencia	
Responsabilidad	**Colaboración**
Adicionar Plan Frecuencia Modificar Plan Frecuencia Eliminar Plan Frecuencia	NomPlanSemanal NomPlanMensual NomPlannModel NomPlanFrecuencia

Tabla 19 Tarjeta CRC Clase: NomMensaje.

Clase: NomMensaje	
Responsabilidad	**Colaboración**
Grabar Mensaje	NomMensajeModel NomUsuario

Tabla 20 Tarjeta CRC Clase: NomPlanMensual.

Clase: NomPlanMensual	
Responsabilidad	**Colaboración**
Adicionar Plan Mensual Modificar Plan Mensual Eliminar Plan Mensual	NomPlanMensualModel

3.3 Diseño de la Base de datos

La construcción de la base de datos es una de las tareas más importantes en el diseño de las aplicaciones, pues en esta se reflejan todas las relaciones y datos necesarios para el correcto funcionamiento de la aplicación. Para el desarrollo de la aplicación se identificaron siete tablas de la base de datos con sus respectivas relaciones. A continuación se muestra el Diagrama-Entidad-Relación (DER) diseñado para la aplicación SACT:

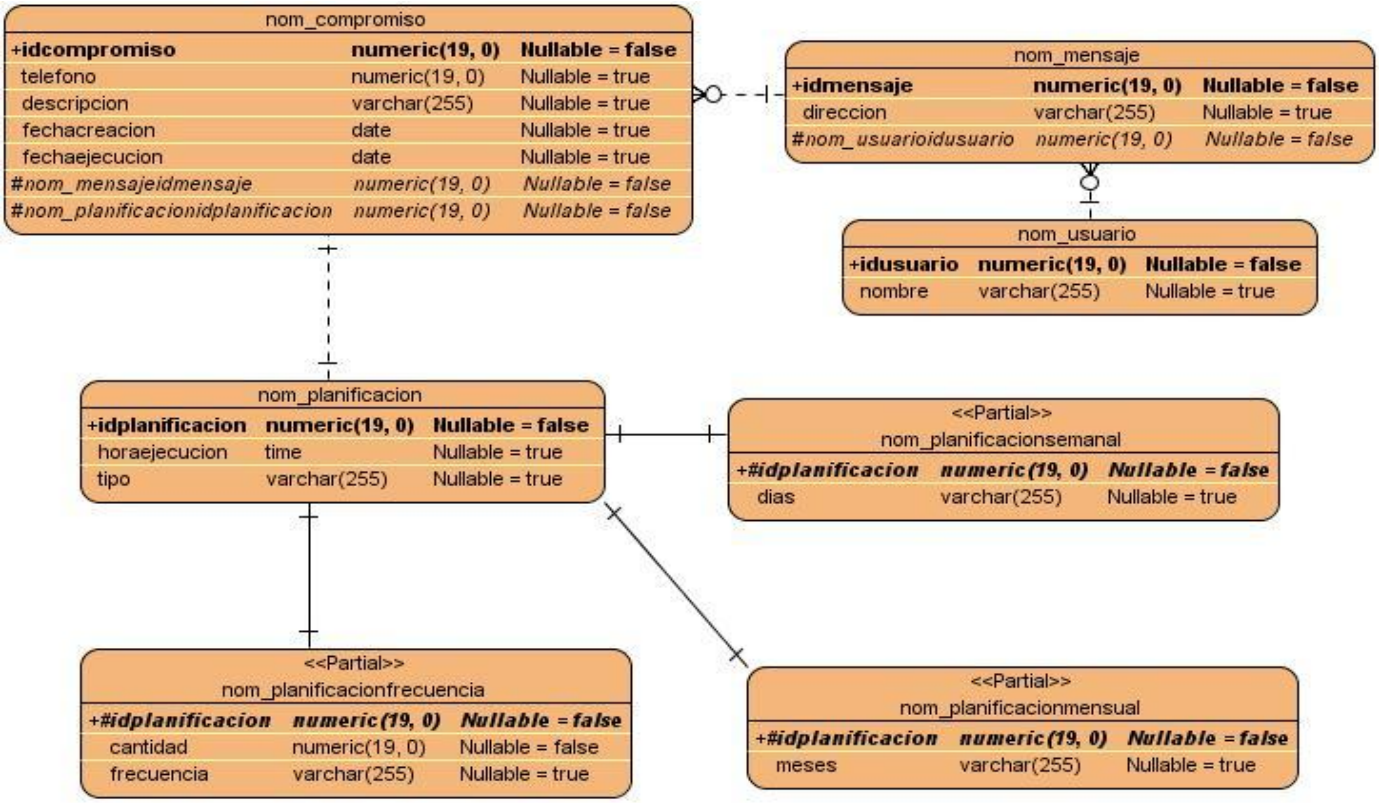


Figura 4 Diagrama Entidad Relación de la Base de Datos.

3.4 Conclusiones parciales:

En este capítulo fueron analizados los patrones de diseño asociados a la arquitectura propuesta, los cuales darán mayor independencia a las clases y facilitarán la implementación. Como parte del diseño del sistema fue identificado el diagrama entidad relación de la base de datos, permitiendo de esta forma almacenar los datos necesarios.

Capítulo 4: Implementación y Prueba

Introducción:

En este capítulo se abordarán dos de las fases más importantes en el ciclo de vida de cualquier software: Implementación y Prueba, en estas fases se materializará el software mediante su codificación y se realizarán las pruebas necesarias para comprobar si fueron cumplidos los objetivos trazados durante la concepción del Sistema de Agenda de Compromisos Telefónica (SACT).

4.1 Fase de Implementación.

Como parte de la metodología ágil escogida durante el inicio de cada iteración se revisa el plan de iteraciones y se expresan las tareas de programación, donde a cada una de ellas le es asignada los programadores responsables, estas tareas son descritas en un lenguaje técnico que no tienen por qué ser entendible para el cliente. Como parte de la planificación realizada en el capítulo anterior se detallan a continuación las iteraciones y las historias de usuario implementadas en cada iteración.

4.1.1 Iteración # 1.

En esta iteración se implementaron las historias de usuarios referentes a las principales funcionalidades de la aplicación web con el objetivo de mostrar al cliente el primer prototipo funcional del producto SACT.

Tabla 21 Historias de usuario implementadas en la primera iteración.

Historia de Usuario	Estimación	Real
Adicionar Solicitud de Compromiso	1	2
Modificar Solicitud de Compromiso	1	1
Cancelar Solicitud de Compromiso	1	1
Total	3	4

4.1.1.1 Tareas de las Historias de Usuario implementadas en la primera iteración:

Tabla 22 Tarea #1 de la historia de usuario: Adicionar Solicitud de Compromiso.

Tarea de Ingeniería	
Número Tarea: 1	**Número Historia de Usuario:**1
Nombre Tarea: Diseño de la interfaz Adicionar Solicitud de Compromiso.	

Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1/7
Fecha Inicio: 14 de febrero 2011	**Fecha Fin:** 15 de febrero 2011
Programador Responsable: Yohandy Casañas – Yeisell G. Medina	
Descripción: Se diseña la interfaz de la funcionalidad Adicionar Solicitud de Compromiso que permite introducir los datos de las solicitudes de compromisos.	

Tabla 23 Tarea #2 de la historia de usuario: Adicionar Solicitud de Compromiso.

Tarea de Ingeniería	
Número Tarea: 2	**Número Historia de Usuario:**1
Nombre Tarea: Implementar la funcionalidad Adicionar Solicitud de Compromiso.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 15 de febrero 2011	**Fecha Fin:** 21 de febrero 2011
Programador Responsable: Yohandy Casañas – Yeisell G. Medina	
Descripción: Se implementa la funcionalidad Adicionar Solicitud de Compromiso, permitiendo que se introduzcan los datos (número de teléfono, hora de ejecución, fecha de ejecución, la planificación (semanal, mensual, para una hora específica o con una frecuencia de ejecución determinada), el mensaje y una breve descripción del compromiso) de cada compromiso, estos son guardados en sus tablas correspondientes de la base de datos.	

Tabla 24 Tarea #1 de la historia de usuario: Modificar Solicitud de Compromiso.

Tarea de Ingeniería	
Número Tarea: 1	**Número Historia de Usuario:** 2
Nombre Tarea: Diseñar la interfaz Modificar Solicitud de Compromiso	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1/7
Fecha Inicio: 2 de marzo de 2011	**Fecha Fin:** 3 de marzo de 2011
Programador Responsable: Yohandy Casañas – Yeisell G. Medina	
Descripción: Se diseña la interfaz de la funcionalidad Modificar Solicitud de Compromiso que permite modificar los datos de los compromisos.	

Tabla 25 Tarea #2 de la historia de usuario: Modificar Solicitud de Compromiso.

Tarea de Ingeniería	
Número Tarea: 2	**Número Historia de Usuario:** 2
Nombre Tarea: Implementar funcionalidad Modificar Solicitud de Compromiso.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 3 de marzo de 2011	**Fecha Fin:** 10 de marzo de 2011
Programador Responsable: Yohandy Casañas – Yeisell G. Medina	
Descripción: Se implementa la funcionalidad Modificar Solicitud de Compromiso la cual permite realizar modificaciones a los datos (número de teléfono, hora de ejecución, fecha de ejecución, la planificación (semanal, mensual, para una hora específica o con una frecuencia de ejecución determinada), al mensaje o a la breve descripción del compromiso) de las solicitudes de compromisos.	

Tabla 26 Tarea #1 de la historia de usuario: Cancelar Solicitud de Compromiso.

Tarea de Ingeniería	
Número Tarea: 1	**Número Historia de Usuario:** 3
Nombre Tarea: Diseño de la interfaz Cancelar Solicitud de Compromiso	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1/7
Fecha Inicio: 22 de febrero 2011	**Fecha Fin:** 23 de febrero 2011
Programador Responsable: Yohandy Casañas – Yeisell G. Medina	
Descripción: Se diseña la interfaz la funcionalidad Cancelar Solicitud de Compromiso que permite cancelar los datos de las solicitudes de compromisos.	

Tabla 27 Tarea #1 de la historia de usuario: Cancelar Solicitud de Compromiso.

Tarea de Ingeniería	
Número Tarea: 1	**Número Historia de Usuario:** 3
Nombre Tarea: Implementar Cancelar Solicitud de Compromiso	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 23 de febrero 2011	**Fecha Fin:** 1 de marzo 2011
Programador Responsable: Yohandy Casañas – Yeisell G. Medina	

Descripción: Se implementa la funcionalidad Cancelar Solicitud de Compromiso la cual se debe seleccionar en el componente GridPanel para poder cancelar la solicitud del compromiso seleccionado.

4.1.2 Iteración # 2.

En esta iteración se implementaron las historias de usuarios referentes a las principales funcionalidades del Proceso y la AGI con el objetivo de mostrar al cliente un prototipo funcional integrado del producto SACT.

Tabla 27 Historias de usuario implementadas en la segunda iteración.

Historia de Usuario	Estimación	Real
Grabar Mensaje del Usuario	1	2
Enviar Mensaje de Compromiso	1	1
Total	**2**	**2**

4.1.2.1 Tareas de las Historias de Usuario implementadas en la segunda iteración.

Tabla 28 Tarea #1 de la historia de usuario: Grabar Mensaje del Usuario.

Tarea de Ingeniería	
Número Tarea: 1	**Número Historia de Usuario:** 4
Nombre Tarea: Implementar la funcionalidad Grabar Mensaje del Usuario	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 4/7
Fecha Inicio: 4 de abril de 2011	**Fecha Fin:** 7 de abril de 2011
Programador Responsable: Yohandy Casañas – Yeisell G. Medina	
Descripción: Se implementará la funcionalidad que permitirá al usuario grabar sus mensajes personalizados.	

Tabla 29 Tarea #2 de la historia de usuario: Grabar Mensaje del Usuario.

Tarea de Ingeniería	
Número Tarea: 2	**Número Historia de Usuario:** 4
Nombre Tarea: Almacenar el mensaje grabado	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 2/7

Fecha Inicio: 7 de abril del 2011	**Fecha Fin:** 8 de abril del 2011
Programador Responsable: Yohandy Casañas – Yeisell G. Medina	
Descripción: Se almacena el mensaje grabado en la carpeta de sonidos predeterminados de Asterisk para su posterior utilización.	

Tabla 30 Tarea #2 de la historia de usuario: Enviar Mensaje de Compromiso.

Tarea de Ingeniería	
Número Tarea: 2	**Número Historia de Usuario:** 5
Nombre Tarea: Implementar la funcionalidad Enviar Mensaje de Voz	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 15 de abril del 2011	**Fecha Fin:** 21 de abril del 2011
Programador Responsable: Yohandy Casañas – Yeisell G. Medina	
Descripción: Se implementa la funcionalidad que permitirá enviar al usuario su mensaje de aviso de compromiso a partir de los datos almacenados en la carpeta de sonidos predeterminados de Asterisk.	

4.1.3 Iteración # 3

En esta iteración se implementaron las historias de usuarios encargadas de complementar el resultado de lo implementado en las iteraciones anteriores con el objetivo de mostrar al cliente el producto final.

Tabla 31 Historias de usuario implementadas en la segunda iteración

Historia de Usuario	Estimación	Real
Repetir Mensaje de Compromiso	1	1
Terminar Mensaje de Compromiso	1	1
Total	**2**	**2**

4.1.3.1 Tareas de las Historias de Usuario implementadas en la segunda iteración.

Tabla 32 Tarea #1 de la historia de usuario: Repetir Mensaje de Compromiso.

Tarea de Ingeniería	
Número Tarea: 1	**Número Historia de Usuario:** 6
Nombre Tarea: Implementar funcionalidad Repetir Mensaje de Compromiso	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 5 de mayo de 2011	**Fecha Fin:** 11 de mayo de 2011
Programador Responsable: Yohandy Casañas – Yeisell G. Medina	
Descripción: Se implementa la funcionalidad que permite repetir el mensaje de compromiso al usuario, el mensaje se repetirá una vez cada sesenta segundos para un total de tres repeticiones en ciento ochenta segundos.	

Tabla 33 Tarea #1 de la historia de usuario: Terminar Mensaje de Compromiso.

Tarea de Ingeniería	
Número Tarea: 1	**Número Historia de Usuario:** 7
Nombre Tarea: Implementar la funcionalidad Terminar Mensaje de Compromiso	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1
Fecha Inicio: 12 de mayo de 2011	**Fecha Fin:** 18 de mayo de 2011
Programador Responsable: Yohandy Casañas – Yeisell G. Medina	
Descripción: Mediante esta funcionalidad se terminará el envío del mensaje de compromiso, ya sea porque el usuario escuchó el mensaje o porque fue repetido tres veces en ciento ochenta segundos.	

4.2 Pruebas.

Las pruebas permiten comprobar la eficacia de un sistema de software, siendo las encargadas de verificar si los objetivos trazados fueron cumplidos en la etapa de implementación. Con ellas es reducido el número de errores no detectados durante la implementación, así como el tiempo entre la introducción de este error en el sistema y su detección; son las encargadas de aumentar la seguridad y de evitar efectos colaterales no deseados a la hora de realizar modificaciones y refactorizaciones en la aplicación.

La metodología de desarrollo XP permite dividir las pruebas en dos grupos: pruebas unitarias, las cuales se encargan de verificar el código y son diseñadas por los programadores y las pruebas de aceptación o pruebas funcionales, estas se encuentran destinadas a evaluar si al final de una iteración se consiguió la funcionalidad requerida diseñada por el cliente final.

4.2.1 Pruebas de Aceptación.

Las pruebas de aceptación son pruebas de caja negra que se crean a partir de las historias de usuario y son realizadas por el cliente y los usuarios finales de la aplicación. En ellas serán probadas las funcionalidades exigidas por el cliente. Una historia de usuario puede tener todas las pruebas de aceptación que necesite para asegurar su correcto funcionamiento. Luego de haber superado las pruebas de aceptación podrá considerarse que la aplicación es apta para el uso y despliegue dentro del proyecto.

Las pruebas de aceptación se llevarán a cabo de la forma siguiente:

1. Se redactarán los casos de prueba teniendo en cuenta el orden de las historias de usuarios y los niveles de prioridad dados a las funcionalidades.

2. Se hará la planificación con el cliente de cuándo y cuáles pruebas serán llevadas a cabo.

3. Se reunirán los miembros del proyecto seleccionados para realizar las pruebas.

4. Se completará cada uno de los campos de la tabla de pruebas de aceptación con el resultado de cada prueba.

Para un mayor entendimiento de las pruebas realizadas a la aplicación SACT, estas son mostradas en el Anexo I.

4.3 Conclusiones Parciales:

En el presente capítulo se abordaron las tareas de ingeniería, las cuales permitieron describir las tareas realizadas para implementar las funcionalidades, luego de la etapa de implementación se dio paso las pruebas, con estas fueron probados los requisitos trazados.

Capítulo 5: Estudio de Factibilidad

Introducción:

La estimación es un tema muy importante en el ciclo de vida de cualquier software ya que da una aproximación de los gastos de salario, tecnologías u otros que se incurrirá en el desarrollo del mismo, es por lo que en este capítulo se realizará el estudio de la factibilidad de la solución informática SACT, mediante este estudio se tendrá una estimación del tamaño del software, el esfuerzo y el tiempo necesario para llevarlo a cabo, la planificación necesaria para desarrollarlo y el costo del proyecto, al concluir con este estudio se tendrá la medida de lo factible que sería desarrollar o no el software.

5.1 Modelo matemático COCOMO II.

COCOMO o el Modelo Constructivo de Costes es un modelo matemático utilizado para estimar esfuerzo, costes de software y tiempo requerido en un proyecto de software. Este modelo está compuesto por tres submodelos de estimación de coste, los cuales tienen en cuenta las necesidades de cada sector y el tipo de información disponible en cada etapa del ciclo de vida de desarrollo. Estos tres submodelos se denominan:

- ✓ Modelo de Composición de Aplicaciones
- ✓ Modelo Post-Arquitectura
- ✓ Modelo de Diseño Anticipado.

El estudio de factibilidad del sistema SACT se realizó por el modelo de diseño anticipado, este modelo se utiliza para obtener estimaciones aproximadas del coste de un proyecto para lo cual se utiliza un pequeño conjunto de drivers de coste nuevo y nuevas ecuaciones de estimación basándose fundamentalmente en puntos de función sin ajustar o KSLOC (Miles de líneas de código fuente), utilizándose en las primeras etapas del desarrollo. Ese nivel de detalle en este modelo es consistente con el nivel general de información disponible y con el nivel general de estimación detallada que es necesaria en estas etapas, lo que concuerda con el uso de puntos de función. Para estimar tamaño usa puntos de función no ajustados como métrica de medida y un número reducido de factores de costo.

El nivel de detalle de este modelo puede ser consistente con el nivel general de información disponible y con el nivel general de aproximación de la estimación requerida en esta etapa.

5.2 Características del Proyecto.

Para llevar a cabo una correcta estimación el primer paso a ejecutar es la obtención de los puntos de función desajustados, estos puntos se obtienen de la suma de las entradas, salidas y consultas externas que el software realiza, además estas entradas también son sumadas a las consultas que se realizan a los archivos lógicos internos y las interfaces externas necesarias para su correcto funcionamiento.

5.2.1 Entradas externas.

Se definen como entradas externas el proceso mediante el cual los datos cruzan la frontera del sistema de afuera hacia adentro. En el caso particular de la aplicación propuesta se identifican como entradas todas aquellas acciones que realiza el cliente mediante las cuales introduce datos al sistema, estas entradas son especificadas en la siguiente tabla:

Tabla 34 Entradas externas.

Nombre de la entrada externa	Cantidad de ficheros	Cantidad de elementos de datos	Clasificación (Simple, Media. Compleja)
Adicionar Solicitud de Compromiso.	1	6	Simple
Cancelar Solicitud de Compromiso.	1	1	Simple
Modificar Solicitud de Compromiso.	1	6	Simple
Grabar Mensaje del Usuario	1	1	Simple
Terminar Mensaje de Compromiso	1	1	Simple
Total		**15**	

5.2.2 Salidas Externas.

Se definen como salidas externas todo proceso mediante el cual los datos cruzan la frontera del sistema desde adentro hacia afuera. En el caso particular de la aplicación propuesta identifican como salidas todas aquellas acciones que realiza el sistema mediante las cuales envía datos al usuario, estas salidas son especificadas en la siguiente tabla:

Tabla 35 Salidas externas.

Nombre de la salida externa.	Cantidad de ficheros.	Cantidad de elementos de datos.	Clasificación (Simple, Media. Compleja)
Enviar Mensaje de Compromiso.	1	1	Simple
Repetir Mensaje de compromiso	1	1	Simple
Total		**2**	

5.2.3 Consultas Externas

Se definen como un proceso elemental con componentes de entrada y de salida donde se utilizan datos de uno o más Archivos Lógicos Internos o Archivos de Interfaz Externos. En el caso de la aplicación a desarrollar no fue identificada ninguna consulta externa.

Tabla 36 Consultas externas.

Nombre de la Petición.	Cantidad de ficheros.	Cantidad de elementos de datos.	Clasificación (Simple, Media. Compleja)
Total	**0**	**0**	

5.2.4 Archivos Lógicos Internos.

Se definen como archivos lógicos internos un grupo de datos relacionados lógicamente e identificables por el usuario, estos archivos se nutren de las entradas externas del sistema. En el caso particular de la aplicación propuesta se identifican como archivos lógicos internos todos aquellos que incluyen un grupo lógico de datos que son generados, usados o mantenidos por el sistema siendo obtenidos de las entradas externas y realizan una función determinada en el sistema, estos archivos son identificados en la siguiente tabla:

Tabla 37 Fichero lógico interno.

Nombre de la petición.	Cantidad de ficheros.	Cantidad de elementos de datos.	Clasificación (Simple, Media. Compleja)
Base Batos: Agenda	1	7(tablas)	Simple.
Proceso(Demonio)	1	7(tablas)	Simple
Total		**14**	

5.2.5 Archivos de Interfaz Externos.

Son un grupo de datos relacionados lógicamente e identificables por el usuario, que se utilizan solamente para fines de referencia. Los datos residen enteramente fuera de los límites del sistema y se mantienen por las Entradas Externas de otras aplicaciones. En el caso de la aplicación SACT no fue identificado ningún un Archivos de interfaz externa.

Tabla 38 Interfaces externas

Nombre de la interfaz externa.	Cantidad de ficheros.	Cantidad de elementos de datos.	Clasificación (Simple, Media. Compleja)
Total	**0**	**0**	

5.2.6 Puntos de función desajustados.

En la tabla siguiente se muestran los puntos de función desajustados, los cuales son basados en las características del sistema. Estos puntos se obtienen del producto de la cantidad existente de cada una de las características y el peso correspondiente a las mismas, dando como resultado final los puntos de función desajustados. Para un mayor entendimiento de cómo fueron obtenidos los puntos de función desajustados consultar la tabla de peso del factor de complejidad en el Anexo III.

Tabla 39 Puntos de función desajustados.

Elementos	Simple		Medio		Complejo		Subtotal
	No.	Peso	No.	Peso	No.	Peso	
Entradas externas.	15	3	0	4	0	6	45

Salidas externas.	2	4	0	5	0	7	8
Consultas externas.	0	3	0	4	0	6	0
Fichero lógico interno.	14	7	0	10	0	15	98
Fichero interfaz externo.	0	5	0	7	0	10	0
Total (UFP):							**151**

5.3 Cálculo de instrucciones fuentes.

Cuando los puntos de función desajustados pertenecientes al proyecto son identificados el siguiente paso a llevar a cabo es el cálculo de la cantidad de instrucciones fuentes, este paso es descrito a continuación:

Ecuación 1:

$SLOC = UFP\ x\ ratio$

$SLOC$ = 151x59

$SLOC$ = 8909

Donde se tienen como variables:

UFP : Puntos de función desajustados.

$ratio$: Conversión de puntos de función desajustados a líneas de código para el lenguaje PHP, el cual es una constante con un valor igual a 59.

5.4 Cálculo del tamaño del software

Ecuación 2:

$KSLOC = SLOC/1000$

$KSLOC$ = 8909/1000

$KSLOC$=8.9 KSLOC

Nota: $KSLOC$: Miles de líneas de código fuente.

Los valores obtenidos del cálculo de los puntos de función desajustados, la cantidad de instrucciones fuentes por puntos de función, así como las instrucciones fuentes han quedado plasmados en la tabla siguiente:

Tabla 40 Características del sistema

Características	Valor
Puntos de función desajustados	151
Lenguaje (PHP)	59
Instrucciones fuentes por puntos de función	8909 SLOC
Instrucciones fuentes	8.9 KSLOC

5.5 Cálculo de esfuerzo nominal.

La ecuación de cálculo del esfuerzo nominal para un proyecto tiene como entradas la medida del desarrollo del software, una constante A y un factor de escala B. La medida está en unidades de líneas de código fuente (KSLOC), el factor de escala explica el ahorro o gasto relativo de escala encontrado en proyectos software de distintos tamaños. La constante A se usa para cortar los efectos multiplicativos de esfuerzo en proyectos de tamaño incremental. El esfuerzo nominal es calculado a continuación:

Ecuación 3:

$$PM_{Nominal} = Ax(Size)^{B}$$

$$PM_{Nominal} = 2,94\ x(8,90)^{1,00}$$

$$PM_{Nominal} = 26,17\ meses/hombre$$

$$PM_{Nominal} \approx 26\ meses/hombre$$

Donde se tienen como variables:

$PM_{Nominal}$: Esfuerzo nominal requerido en meses – hombres.

$Size$: Tamaño estimado del software en Puntos de Función sin Ajustar

A: Constante que se utiliza para capturar los efectos multiplicativos en el esfuerzo requerido de acuerdo al crecimiento de tamaño del software con valor igual a 2.94.

B: Factor escalar y su valor está dado por la resultante de los aspectos positivos sobre los negativos que presenta el proyecto, calculándose de la siguiente forma:

Ecuación 4:

$$B = 0,91 + 0,01\ x \sum (W_i)$$

B =0.91 + 0.01 x 10.22

B =**1.00**

Donde se tiene como variables:

W_i: Variables escalares que indican las características que el proyecto presenta en lo que a su complejidad y entorno de desarrollo se refiere.

5.6 Factores de escala

5.6.1 Precedentes (PREC).

El factor de precedencia (PREC) toma en cuenta el grado de experiencia que se tiene en relación al producto a desarrollar, tanto en aspectos organizacionales como en el conocimiento del software y hardware a utilizar.

5.6.2 Flexibilidad de desarrollo (FLEX).

El factor de flexibilidad (FLEX) considera el nivel de exigencia en el cumplimiento de los requerimientos preestablecidos, plazos de tiempos y especificaciones de interfaz.

5.6.3 Cohesión del equipo (TEAM).

El factor de escala Cohesión del Equipo tiene en cuenta las dificultades y desincronización que ocurren entre los participantes de un proyecto, es decir incomunicaciones entre usuarios, clientes y desarrolladores. Estas dificultades pueden surgir por diferencias culturales, dificultad en la conciliación de objetivos, falta de experiencia y familiaridad con el trabajo en equipo.

5.6.4 Solución de riesgos (RESL).

Este factor involucra aspectos relacionados al conocimiento del riesgo crítico y el modo de abordarlos dentro del proyecto.

5.6.5 Madurez del proceso (PMAT).

El período de tiempo ideal para medir la madurez de un proceso es el momento en el que el proyecto comienza, por lo que el procedimiento para determinar el PMAT es establecer el porcentaje de cumplimiento de cada una de las áreas evaluando el grado de cumplimiento de las metas correspondientes.

A continuación se muestra una tabla con los valores que han sido asignados a cada uno de los factores de escala:

Tabla 41 Factores de escala.

Nombre	Valor	Justificación
PREC	3.72	Solo existe un proyecto similar que involucra agendas y teléfonos, además de existir algunas aplicaciones para computadoras. A nivel nacional no se cuenta con ninguna aplicación de este tipo.
FLEX	1.01	El sistema cuenta con alta flexibilidad en cuanto a los requerimientos establecidos inicialmente.
TEAM	1.10	El equipo de desarrollo presenta una alta cohesión.
RESL	2.83	Para el desarrollo de la aplicación los riesgos que se identificaron no tienen gran trascendencia.
PMAT	1.56	El equipo de desarrollo cuenta con la experiencia necesaria para que el software cumpla con las funcionalidades requeridas por el cliente.
Total(SF)	**10.22**	

Para un mayor entendimiento de cómo fueron obtenidos los factores de escala consultar la tabla de factores de escala en el Anexo III.

5.7 Ajuste del esfuerzo nominal.

El esfuerzo es un valor nominal y debe ser ajustado para lo cual se tiene un conjunto de Multiplicadores de Esfuerzo (MEi) que representan las características del proyecto y expresan su impacto en el desarrollo total del producto de software. Cada nivel de medida del esfuerzo nominal tiene un peso asociado, este peso se llama Multiplicador de Esfuerzo (ME), el cual se usa para ajustar el esfuerzo meses-hombre de un proyecto determinado. Los multiplicadores del esfuerzo para el modelo de diseño anticipado se dividen en 7 categorías, las cuales son mostradas en la tabla siguiente:

Tabla 42 clasificación de los multiplicadores del esfuerzo.

Multiplicadores del esfuerzo	
Del Producto	RCPX: Confiabilidad y Complejidad del Producto
	RUSE: Reusabilidad Requerida
De la Plataforma	PDIF: Dificultad de la Plataforma
Del Proyecto	FCIL: Facilidades
	SCED: Cronograma de Desarrollo Requerido
Del Personal	PERS: Aptitud del Personal
	PREX: Experiencia del Personal

Tabla 43 Multiplicadores de esfuerzo aplicados al software.

Nombre	**Valor**	**Justificación**
RCPX	1.30	La confidencialidad del sistema es alta.
RUSE	1.00	Se pretende la reutilización de una parte del código.
PDIF	1.00	La plataforma de desarrollo es estable, el uso de la memoria y almacenamiento es normal.
PREX	0. 87	Existe un nivel de experiencia en el personal en cuanto a la utilización del lenguaje y herramientas.
PERS	0.83	La aptitud del personal es alta.
FCIL	0.73	La utilización de entornos de desarrollo integrados facilita en gran medida el desarrollo de la aplicación.
SCED	1.00	El sistema se desarrolló en el tiempo establecido.
Total(ME)	**0.69**	

Ecuación 5:

$$PM_{ajustado} = PM_{Nominal} x \prod (ME_t)$$

$PM_{ajustado}$ ***= 26.17 meses / hombre x 0.69***

$PM_{ajustado}$***=18.06 meses / hombre***

$PM_{ajustado}$***≈18 meses / hombre***

5.8 Calculo del tiempo de desarrollo del software.

El tiempo requerido para el desarrollo del proyecto está dado por:

Ecuación 6:

$$TDEV = C\ x\ (PM_{ajustado})^F$$

$$TDEV = 3.64\ x\ 18.06^{0.26}$$

$$TDEV = 7.72$$

$$TDEV \approx 8$$

Donde se tienen como variables:

C: Constante con valor igual 3.64.

$PM_{ajustado}$ = 18.06 meses/hombre.

F: Es un valor que se calcula de la siguiente forma:

Ecuación 7:

$$F = D + 0.2\ x\ 0.01 x \sum SF$$

F =0.24 + 0.2 x 0.01 x 9.42

F ≈0.26

Donde se tiene como variables:

D : Constante cuyo valor es 0.24.

SF : Valor de los factores de escala.

5.9 Cálculo del costo total del proyecto.

Para el cálculo del costo total correspondiente al proyecto en cuestión, COCOMO II propone:

Ecuación 8:

$C = CHM \; x \; PM$

$C =$ **234 x 18.06**

$C =$ **4226.04**

$C \approx$ **4226**

Donde se tiene como variables:

C: Costo total.

CHM**:** Costo teniendo en cuenta salario de todos los obreros, el cual se calcula de la siguiente forma:

Ecuación 9:

$CHM = CH \; x \; sal$

CHM =2.34 x 100.

$CHM = 234$

Donde se tiene como variables:

sal : Salario medio por cada trabajador estimado en 100.

CH : Cantidad de personas destinadas al proyecto:

5.10 Cálculo de la cantidad de personas destinadas al proyecto

Ecuación 10:

$CH = {}^{PM}/_{TDEV}$

$CH = 18.06/7.72$

CH =2.34

$CH \approx 2$

Donde se tiene como variables:

PM**=**18.06

$TDEV = 7.72$

5.11 Resultados.

A continuación son mostrados en una tabla los resultados obtenidos luego efectuados todos los cálculos necesarios para determinar el costo y esfuerzo requeridos para el desarrollo de la aplicación SACT.

Tabla 44 Resultados.

Cálculo de:	Valor
Esfuerzo	18 meses/hombre.
Tiempo de desarrollo	8 meses.
Cantidad de hombres	2 hombres.
Salario medio	100 pesos.
Costo	4226 pesos.

5.12 Análisis de costo.

Todo desarrollo de cualquier producto siempre tiene un coste asociado, por lo que la justificación de dicho costo se basa fundamentalmente en los beneficios que reportará la obtención del producto final. El Sistema de Agenda de Compromisos Telefónica (SACT) no trae consigo grandes gastos en cuanto a costo monetario se refiere, pues al usarse gran cantidad de herramientas de software libre no se incurren en los gastos de tener que pagar las licencias de dichas herramientas, además sólo se incluye el gasto del salario de los desarrolladores, por lo cual la implementación de dicho sistema es factible.

5.13 Conclusiones.

En el presente capítulo se realizó un análisis detallado de la factibilidad de la solución propuesta, llegando a la conclusión de que el Sistema de Agenda de Compromisos Telefónico se plantea como una solución informática económica y viable para el desarrollo.

Conclusiones

A continuación se presentarán las conclusiones a las cuales se ha arribado a través de la realización del presente trabajo. Con el mismo se pudieron dar cumplimiento a los objetivos planteados en la concepción inicial del sistema.

- ✓ Se realizó un análisis minucioso de las agendas telefónicas a nivel mundial y las diferentes aplicaciones que funcionan como agendas, tomando de ellas ideas y experiencias para brindar el servicio de agendas telefónicas en la telefonía fija.
- ✓ Se realizó una búsqueda y estudio de las herramientas existentes seleccionando aquellas que más se adecuan a las necesidades de la solución que se desarrolló, además de primar el concepto del software libre en la elección.
- ✓ Se escogió la arquitectura del sistema adecuándose a las necesidades del cliente, permitiendo una mayor disponibilidad y un fácil mantenimiento.
- ✓ Se implementaron las funcionalidades necesarias para brindar el servicio de agenda de compromisos, las cuales fueron descritas en las historias de usuario, teniendo como resultado una aplicación Web permite gestionar las solicitudes de avisos de compromisos.
- ✓ Se realizaron las pruebas unitarias para comprobar que las funcionalidades descritas por el cliente fueran correctas.

Luego de la implementación y las pruebas de las diferentes funcionalidades se obtuvo como resultado final una aplicación capaz de encargarse de la gestión y el envío de las solicitudes de compromisos.

Recomendaciones

Luego de concluido el presente trabajo se recomienda:

- ✓ Ampliar los medios de difusión de la solicitud de compromiso, dígase Correo Electrónico, Jabber, Telefonía Móvil.
- ✓ Agregar un módulo de reportes a la aplicación SACT para realizar un seguimiento de los compromisos.

Referencias Bibliográficas

1. Universidad Perú. [En línea] [Citado el: 9 de Noviembre de 2010.] Disponible en: [www.universidadperu.com/telecomunicaciones-peru.php()]..

2. Sitio Oficial del MIC. [En línea] [Citado el: 5 de Octubre de 2010.] Disponible en: [http://www.mic.gov.cu/HThemEmp.aspx?4.]..

3. Asterisk Configuracion Avanzada. [En línea] 2009. [Citado el: 12 de Octubre de 2010.] Disponible en:[http://ws.edu.isoc.org%2Fdata%2F2008%2F30970195492812 00ae4f5%2F09.asterisk-configuracion-avanzada.ppt].

4. Base de Datos. [En línea] [Citado el: 5 de Diciembre de 2010.] http://www.error500.net/garbagecollector/archives/categorias/bases_de_datos/sistema_gestor_de_base_de_datos_sgbd.php..

5. Sistema de Base de Datos. [En línea] 2006. [Citado el: 28 de Noviembre de 2010.] Disponible en:[http://www.error500.net/garbagecollector/bases_de_datos/sistema_gestor_de_base_de_dato.php].

6. Maestrosdelweb. [En línea] [Citado el: 12 de Noviembre de 2010.] Disponible en:[http://www.desarrolloweb.com/articulos/392.php]..

7. Lenguajes de Programacion Web. [En línea] 2007. Disponible en:[http://www.latinozlife.com/index.php?topic=11488.0].

8. Curso para administracion de servidores web. [En línea] [Citado el: 10 de Diciembre de 2010.] Disponible en:[http://www.lima.wesped.es/02.html].

9. linux-ciberaula. ciberaula. [En línea] 2006. [Citado el: 5 de Diciembre de 2010.] Disponible en: [http://linux.ciberaula.com/articulo/linux_apache_intro.].

10. Framework. [En línea] [Citado el: 25 de Noviembre de 2010.] Disponible en: http://www.bynarius.com.ar/herramientas.html.

11. Doctrine. [En línea] [Citado el: 25 de Noviembre de 2010.] disponible en:[http://www.doctrine-project.org/. doctrine-projec].

12. **Alexa.** bds. [En línea] junio de 2009. [Citado el: 8 de Diciembre de 2010.] http://www.proyectosbds.com/software-y-programacion-a-medida/programacion-php-lamp-zf/mas-sobre-zend-framework/121/.

13. IDE, JCREATOR, NET BEANS, BLUE J. [En línea] [Citado el: 20 de Noviembre de 2011.] Disponible en:[http://programacion1itzel.blogspot.com/2009/03/ide-jcreator-net-beans-blue-j.html].

Referencias Bibliográficas:

14. Tufuncion. *Zen Studio.* [En línea] [Citado el: 15 de Noviembre de 2010.] Disponible en:[http://www.tufuncion.com/zend-studio].

15. **Stefan Küng, Lübbe Onken, y Simon Large.** TortoiseSVN. [En línea] 2009. [Citado el: 15 de Diciembre de 2010.] Disponible en:[ftp://10.0.0.22/software/Desarrollo/Control%20de%20versiones/Tortoise/TortoiseSVN-1.6.7-es(2).pdf].

16. Modelo Cliente Servidor. [En línea] [Citado el: 20 de Febrero de 2011.] http://marcosventuraosorio261v.blogspot.com/2009_04_01_archive.html Cliente servidor.

Bibliografía

1. Agenda Telefónica Click to Dial. [En línea] [Citado el: 10 de Noviembre de 2010.] Presentación Disponible en:[http://www.google.com.cu/#q=agendas+telefonicas+basadas+en+asterisk&hl=es&biw=1024&bih=576&ei=2V3hTKnvNaHtnQeYr_WJDw&start=10&sa=N&fp=123fa313d55e5dee].

2. Introducción a Apache. [En línea] [Citado el: 24 de Noviembre de 2010.] Disponible en: [http://linux.ciberaula.com/articulo/linux_apache_intro/].

3. Lenguajes de programacion web. [En línea] [Citado el: 25 de Noviembre de 2010.] Disponible en:[http://www.larevistainformatica.com/lenguajes-programacion-web.htm].

4. Metodologías de desarrollo de software. Capítulo 2. IAGP. [En línea] 2005. [Citado el: 11 de Octubre de 2010.] Disponible en: [http://www.um.es/docencia/barzana/IAGP/Iagp2.html].

5. **Espinosa, Humberto.** PresentacionES_PSQL. [En línea] [Citado el: 11 de Noviembre de 2010.] Disponible en: [http://www.lgs.com.ve/pres/PresentacionES_PSQL.pd f].

6. saludinova. [En línea] [Citado el: 9 de Noviembre de 2010.] Disponible en: [http://www.saludinova.com/practices/view/407].

7. Sistema gestor de base de datos. [En línea] 2004. [Citado el: 10 de Noviembre de 2010.] Disponible en: [http://www.error500.net/garbagecollector/archives/categorias/bases_de_datos/sistema_gestor_de_base_de_datos_sgbd.php].

8. Sitio Ocicial de PHP. [En línea] [Citado el: 25 de Noviembre de 2010.] Disponible en: [http://www.php.net].

9. Sitio oficial de Asterisk PBX. [En línea] [Citado el: 14 de Noviembre de 2010.] Disponible en: [www.asterisk.org].

10. Sitio Oficial del Apache. [En línea] [Citado el: 24 de Noviembre de 2010.] Disponible en:[http://apache.org/].

11. Sitio Oficial del Postgresql. [En línea] [Citado el: 25 de Octubre de 2010.] Disponible en: [http://www.postgresql.org].

12. **Acuña, Karenny Brito.** Selección de Metodologías de Desarrollo para Aplicaciones WEB en la Universidad de Cienfuegos. [En línea] 2009. [Citado el: 5 de Noviembre de 2010.] Disponible en: [http://www.eumed.net/libros/2009c/584/Metodologias%20tradicionales%20y%20metodologias%20agiles.htm].

13. **Cobranza, Software de.** softwaredecobranza. [En línea] [Citado el: 12 de Noviembre de 2010.] Disponible en:[http://www.softwaredecobranza.com/interactive-voice-response.aspx].

14. **Corp., Database System.** databasesystemscorp. [En línea] [Citado el: 11 de Noviembre de 2010.] Disponible en:[http://www.databasesystemscorp.com/pssurvey.htm].

15. **desarrolloweb.** desarrolloweb. [En línea] [Citado el: 16 de Noviembre de 2010.] Disponible en:[http://www.desarrolloweb.com/articulos/392.php].

16. **Jaime Alejandro Díaz Rojas, Tamara Jazmín Ramírez Andrade.** *Desarrollo de Aplicaciones y Soluciones en la Central Asterisk de Telefonía IP en el Departamento de Electrónica de la UTFSM.* Chile : Universidad Federico Santa María, 2006.

17. **José Antonio Plá Rodríguez, Damián Ilizastegui Arriba.** *Sistema para la integración continua de proyectos y el control de builds en la empresa Procyon Soluciones.* La Habana : Universidad de las Ciencias Informáticas (UCI), 2007.

18. **linux.ciberaula.** linux.ciberaula. [En línea] [Citado el: 23 de Noviembre de 2010.] Disponible en:[http://linux.ciberaula.com/articulo/linux_apache_intro/].

19. **Magazine, Estrategia.** ERP. [En línea] 2008. [Citado el: 16 de Noviembre de 2010.] Disponible en:[http://www.gestiopolis.com/administracion-estrategia/estrategia/que-es-erp.htm].

20. **Maite Rodríguez Corbea, Meylin Ordóñez Pérez.** *LA METODOLOGÍA XP APLICABLE AL DESARROLLO DEL SOFTWARE EDUCATIVO EN CUBA.* Ciudad de la Habana : Universidad de las Ciencias Informáticas (UCI), 2007.

21. **Manuel Alejandro Piña Tamayo, Yusel Marrero Acanda.** *Sistema de control de redes.* Ciudad de la Habana : Universidad de las Ciencias Infomáticas(UCI), 2010.

22. **Nicolás Montero Puñales, Luis Eduardo Acosta Aparicio.** *Sistema de Reportes y Facturación de PBX Asterisk.* La Habana : Universidad de las Ciencias Informáticas(UCI), 2009.

23. **programacionextrema.** programacionextrema. [En línea] [Citado el: 23 de Noviembre de 2010.] Disponible en:[http://www.programacionextrema.org/].

24. **Santos, Herminio Heredia.** maestrosdelweb. [En línea] [Citado el: 16 de Noviembre de 2010.] Disponible en:[http://www.maestrosdelweb.com/editorial/phpintro/].

25. **Valdés, Damián Peréz.** Los diferentes lenguajes de programación para la web. [En línea] [Citado el: 25 de Noviembre de 2010.] Disponible en: [http://www.maestrosdelweb.com/principiantes/los-diferentes-lenguajes-de-programacion-para-la-web/].

26. **Victoria.** definicionabc. [En línea] 11 de Enero de 2009. [Citado el: 12 de Noviembre de 2010.] Disponible en:[http://www.definicionabc.com/tecnologia/erp.php].

27. **Wells, Don.** Extreme Programming: A gentle introduction. [En línea] 2006. [Citado el: 9 de Noviembre de 2010.] Disponible en: [http://www.extremeprogramming.org].

28. **Yaidel Alfredo Carvajal Rondón, Héctor Alexander Pérez Coello.** *Herramienta para el diseño y soporte online de un Sistema de Respuesta de Voz Interactiva (PLATEL-IVR).* La Habana : Universidad de las Ciencias Informáticas (UCI), 2009.

29. XP . *Extreme Programming: A gentle introduction.* [En línea] [Citado el: 16 de diciembre de 2011.] Disponible en [http://www.extremeprogramming.org/]..

30. **Sommerville, Ian.** *Ingeniería del Software.* 2005.

31. **Beck, Kent.** *Extreme Programming Explained: Embrace Change.*

32. **Moreno, Ana M.** *COCOMO II.*

33. **Adriana Gómez, María del C.López,Silvina Migani, Alejandra Otazú.** *UN MODELO DE ESTIMACION DE PROYECTOS DE SOFTWARE. .*

Anexo I: Pruebas de Aceptación

Tabla 45 Prueba de Aceptación: HU Adicionar Solicitud de Compromiso

Caso de Prueba de Aceptación	
Código Caso de Prueba: SACT-01-01	**Nombre Historia de Usuario:** Adicionar Solicitud de Compromiso
Nombre de la persona que realiza la prueba: Yeisell G. Medina	
Descripción de la Prueba: Adicionar una solicitud de compromiso para comprobar que es insertada correctamente en la aplicación SACT.	
Condiciones de Ejecución: La aplicación debe de estar disponible y el servidor Asterisk debe de estar activo**.**	
Entrada / Pasos de ejecución: El usuario accede a Gestionar SACT, presiona la opción "Adicionar" e introduce el número de teléfono, la hora de ejecución, la fecha de ejecución, la planificación (semanal, mensual, para una hora específica o con una frecuencia de ejecución determinada), el mensaje, el cual puede ser seleccionado o grabado y una breve descripción del compromiso, en el caso de la planificación escogida sea "Frecuencia", se debe escoger con la cantidad de horas o minutos a los cuales se desea volver a recibir el mensaje.	
Resultado Esperado: El sistema verifica que la entrada de solicitud de compromiso es válida, creando una nueva solicitud y realiza la notificación de compromiso.	
Evaluación de la Prueba: Satisfactoria	

Tabla 46 Prueba de aceptación: HU Adicionar Solicitud de compromiso

Caso de Prueba de Aceptación	
Código Caso de Prueba: SACT-01-02	**Nombre Historia de Usuario:** Adicionar Solicitud de Compromiso
Nombre de la persona que realiza la prueba: Yeisell G. Medina	
Descripción de la Prueba: Adicionar una solicitud de compromiso para comprobar que es insertada correctamente en la aplicación SACT.	
Condiciones de Ejecución: La aplicación debe de estar disponible y el servidor Asterisk debe de estar activo**.**	
Entrada / Pasos de ejecución: El usuario accede a Gestionar SACT, presiona la opción "Adicionar"	

e introduce el número de teléfono, la hora de ejecución, la fecha de ejecución, la planificación (semanal, mensual, para una hora específica o con una frecuencia de ejecución determinada) el mensaje, el cual puede ser seleccionado o grabado y una breve descripción del compromiso, en el caso de la planificación escogida sea “Semanal”, se debe escoger los días de la semana a los cuales se desea ser avisado.
Resultado Esperado: El sistema verifica que la entrada de solicitud de compromiso es válida, creando una nueva solicitud y realiza la notificación de compromiso.
Evaluación de la Prueba: Satisfactoria

Tabla 47 Prueba de aceptación: HU Adicionar Solicitud de compromiso

Caso de Prueba de Aceptación	
Código Caso de Prueba: SACT-01-03	**Nombre Historia de Usuario:** Adicionar Solicitud de Compromiso
Nombre de la persona que realiza la prueba: Yeisell G. Medina	
Descripción de la Prueba: Adicionar una solicitud de compromiso para comprobar que es insertada correctamente en la aplicación SACT.	
Condiciones de Ejecución: La aplicación debe de estar disponible y el servidor Asterisk debe de estar activo**.**	
Entrada / Pasos de ejecución: El usuario accede a Gestionar SACT, presiona la opción “Adicionar“ e introduce el número de teléfono, la hora de ejecución, la fecha de ejecución, la planificación (semanal, mensual, para una hora específica o con una frecuencia de ejecución determinada), el mensaje, el cual puede ser seleccionado o grabado y una breve descripción del compromiso, en el caso de la planificación escogida sea “Mensual”, se debe escoger los meses en los que desea ser avisado.	
Resultado Esperado: El sistema verifica que la entrada de solicitud de compromiso es válida, creando una nueva solicitud y realiza la notificación de compromiso.	
Evaluación de la Prueba: Satisfactoria	

Tabla 48 Prueba de aceptación: HU Adicionar Solicitud de compromiso

Caso de Prueba de Aceptación	
Código Caso de Prueba:	**Nombre Historia de Usuario:** Adicionar Solicitud de Compromiso

SACT-01-04	
Nombre de la persona que realiza la prueba: Yeisell G. Medina	
Descripción de la Prueba: Adicionar una solicitud de compromiso para comprobar que es insertada correctamente en la aplicación SACT.	
Condiciones de Ejecución: La aplicación debe de estar disponible y el servidor Asterisk debe de estar activo**.**	
Entrada / Pasos de ejecución: El usuario accede a Gestionar SACT, presiona la opción "Adicionar" e introduce el número de teléfono, la hora de ejecución, la fecha de ejecución, la planificación (semanal, mensual, para una hora específica o con una frecuencia de ejecución determinada), el mensaje, el cual puede ser seleccionado o grabado y una breve descripción del compromiso, en el caso de la planificación escogida sea "Hora Específica", será avisado a esa hora.	
Resultado Esperado: El sistema verifica que la entrada de solicitud de compromiso es válida, creando una nueva solicitud y realiza la notificación de compromiso.	
Evaluación de la Prueba: Satisfactoria	

Tabla 49 Prueba de aceptación: HU Adicionar Solicitud de compromiso

Caso de Prueba de Aceptación	
Código Caso de Prueba: SACT-01-05	**Nombre Historia de Usuario:** Adicionar Solicitud de Compromiso
Nombre de la persona que realiza la prueba: Yeisell G. Medina	
Descripción de la Prueba: Introducir datos que no sean correctos con el objetivo de comprobar que no son insertados en la base de datos compromisos con datos erróneos.	
Condiciones de Ejecución: La aplicación debe de estar disponible y el servidor Asterisk debe de estar activo**.**	
Entrada / Pasos de ejecución: El usuario accede a Gestionar SACT, presiona el botón "Adicionar" e introduce valores incorrectos ya sean: el número de teléfono, la hora de ejecución, la fecha de creación, la fecha de ejecución o la planificación, si es de tipo "Frecuencia", el número de minutos u horas.	
Resultado Esperado: El sistema para el número de teléfono no permite la entrada de ningún valor que no sea un número y muestra una raya roja cuando la longitud de este no es la correcta, mientras que para la cantidad de minutos u horas no permite ninguna entrada por teclado que no sea un número, además los minutos tienen que estar entre cinco y sesenta y las horas entre cero y	

doce. En el caso de la fecha de ejecución solo se podrá escoger una mayor o igual que la del día en que se está realizando la solicitud de compromiso; para la hora de ejecución en el caso de que la fecha se la del día en que se realiza la solicitud solo se podrán escoger iguales o superiores a la hora del momento, además se muestra un mensaje indicando que se debe seleccionar la fecha primero que la hora. Al no ser correctos los datos no son insertados en la Base de Datos.
Evaluación de la Prueba: Satisfactoria

Tabla 50 Prueba de aceptación: HU Adicionar Solicitud de compromiso

Caso de Prueba de Aceptación	
Código Caso de Prueba: SACT-01-06	**Nombre Historia de Usuario:** Adicionar Solicitud de Compromiso
Nombre de la persona que realiza la prueba: Yeisell G. Medina	
Descripción de la Prueba: Introducir datos que no sean correctos con el objetivo de comprobar que no son insertados en la base de datos compromisos con datos erróneos.	
Condiciones de Ejecución: La aplicación debe de estar disponible y el servidor Asterisk debe de estar activo**.**	
Entrada / Pasos de ejecución: El usuario accede a Gestionar Solicitud, presiona el botón “Adicionar” y deja cualquier campo en blanco.	
Resultado Esperado: El sistema muestra una raya roja que indica que se ha dejado un campo obligatorio vacio. Al no ser correctos los datos no son insertados en la Base de Datos.	
Evaluación de la Prueba: Satisfactoria	

Tabla 51 Prueba de aceptación: HU Modificar Solicitud de compromiso

Caso de Prueba de Aceptación	
Código Caso de Prueba: SACT-02-01	**Nombre Historia de Usuario:** Modificar Solicitud de Compromiso
Nombre de la persona que realiza la prueba: Yeisell G. Medina	
Descripción de la Prueba: Modificar una solicitud de compromiso para comprobar que es modificada correctamente en la aplicación SACT.	
Condiciones de Ejecución: La aplicación debe de estar disponible y el servidor Asterisk debe de estar activo**.**	

Entrada / Pasos de ejecución: El usuario accede a Gestionar SACT, presiona la opción "Modificar" y escoge modificar el número de teléfono, la hora de ejecución, la fecha de ejecución o la planificación (semanal, mensual, para una hora específica o con una frecuencia de ejecución determinada), el mensaje, el cual puede ser seleccionado o grabado y una breve descripción del compromiso, en el caso de la planificación escogida sea "Semanal", se puede modificar los días de la semana a los cuales se desea ser avisado.
Resultado Esperado: El sistema verifica que la entrada de modificación es válida, inserta los datos en la base de datos.
Evaluación de la Prueba: Satisfactoria

Tabla 52 Prueba de aceptación: HU Modificar Solicitud de compromiso

Caso de Prueba de Aceptación	
Código Caso de Prueba: SACT-02-02	**Nombre Historia de Usuario:** Modificar Solicitud de Compromiso
Nombre de la persona que realiza la prueba: Yeisell G. Medina	
Descripción de la Prueba: Modificar una solicitud de compromiso para comprobar que es modificada correctamente en la aplicación SACT.	
Condiciones de Ejecución: La aplicación debe de estar disponible y el servidor Asterisk debe de estar activo**.**	
Entrada / Pasos de ejecución: El usuario accede a Gestionar SACT, selecciona sobre Gridpanel el compromiso deseado y el presiona la opción "Modificar" donde puede modificar cualquier parámetro de la solicitud de compromiso, ya sea: el número de teléfono, la hora de ejecución, la fecha de ejecución o la planificación (semanal, mensual, para una hora específica o con una frecuencia de ejecución determinada), el mensaje, el cual puede ser seleccionado o grabado y una breve descripción del compromiso, en el caso de la planificación escogida sea "Mensual", se puede modificar los meses a los cuales se desea ser avisado.	
Resultado Esperado: El sistema verifica que la entrada de modificación es válida, inserta los datos en la base de datos.	
Evaluación de la Prueba: Satisfactoria	

Tabla 53 Prueba de aceptación: HU Modificar Solicitud de compromiso

Caso de Prueba de Aceptación	
Código Caso de Prueba: SACT-02-03	**Nombre Historia de Usuario:** Modificar Solicitud de Compromiso
Nombre de la persona que realiza la prueba: Yeisell G. Medina	
Descripción de la Prueba: Introducir datos que no sean correctos con el objetivo de comprobar que no son modificados los ya existentes.	
Condiciones de Ejecución: La aplicación debe de estar disponible y el servidor Asterisk debe de estar activo**.**	
Entrada / Pasos de ejecución: El usuario accede a Gestionar SACT, selecciona sobre Gridpanel el compromiso deseado y el presiona la opción "Modificar" donde introduce datos incorrectos en cualquier parámetro de la solicitud de compromiso, ya sea: el número de teléfono, la hora de ejecución, la fecha de ejecución o la planificación, la cual puede ser semanal, mensual, frecuencia o para una hora específica, el mensaje, el cual puede ser seleccionado o grabado y una breve descripción del compromiso, en el caso de la planificación escogida sea "Semanal", se puede modificar los días de la semana a los cuales se desea volver a recibir el mensaje.	
Resultado Esperado: El sistema para el número de teléfono no permite la entrada de ningún valor que no sea un número y muestra una raya roja cuando la longitud de este no es la correcta. En el caso de la fecha de ejecución solo se podrá escoger una mayor o igual que la del día en que se está realizando la solicitud de compromiso; para la hora de ejecución en el caso de que la fecha se la del día en que se realiza la solicitud solo se podrán escoger iguales o superiores a la hora del momento, además se muestra un mensaje indicando que se debe seleccionar la fecha primero que la hora, en el caso de la semana es de obligatoria selección los nombres de los nuevos días de la semana. Al no ser correctos los datos no son insertados en la Base de Datos.	
Evaluación de la Prueba: Satisfactoria	

Tabla 54 Prueba de aceptación: HU Cancelar Solicitud de compromiso

Caso de Prueba de Aceptación	
Código Caso de Prueba: SACT-03-01	**Nombre Historia de Usuario:** Cancelar Solicitud de Compromiso
Nombre de la persona que realiza la prueba: Yeisell G. Medina	

Descripción de la Prueba: Comprobar que una solicitud de compromiso es eliminado de manera satisfactoria.
Condiciones de Ejecución: La aplicación debe de estar disponible y el servidor Asterisk debe de estar activo**.**
Entrada / Pasos de ejecución: El usuario accede a Gestionar SACT, selecciona sobre Gridpanel el compromiso que desea eliminar y el presiona la opción “Eliminar“, luego confirma la eliminación al presionar el botón “Aceptar”.
Resultado Esperado: El sistema muestra un mensaje con la opción de “Aceptar” y “Cancelar”, al ser seleccionada la opción “Aceptar” elimina la solicitud seleccionada de la base de datos.
Evaluación de la Prueba: Satisfactoria

Tabla 55 Prueba de aceptación: HU Cancelar Solicitud de compromiso

Caso de Prueba de Aceptación	
Código Caso de Prueba: SACT-03-02	**Nombre Historia de Usuario:** Cancelar Solicitud de Compromiso
Nombre de la persona que realiza la prueba: Yeisell G. Medina	
Descripción de la Prueba: Comprobar que una solicitud de compromiso cuando es cancelada no es eliminada.	
Condiciones de Ejecución: La aplicación debe de estar disponible y el servidor Asterisk debe de estar activo**.**	
Entrada / Pasos de ejecución: El usuario accede a Gestionar SACT, selecciona sobre Gridpanel el compromiso que desea eliminar y el presiona la opción “Eliminar“, luego confirma la eliminación al presionar el botón “Cancelar”.	
Resultado Esperado: El sistema muestra un mensaje con la opción de “Aceptar” y “Cancelar”, al ser seleccionada la opción “Cancelar” el sistema cancela la eliminación de forma satisfactoria.	
Evaluación de la Prueba: Satisfactoria	

Tabla 56 Prueba de aceptación: HU Garbar Mensaje de compromiso

Caso de Prueba de Aceptación	
Código Caso de Prueba: SACT-04-01	**Nombre Historia de Usuario:** Grabar Mensaje de Compromiso

Nombre de la persona que realiza la prueba: Yeisell G. Medina
Descripción de la Prueba: Comprobar que el mensaje de solicitud de compromiso es grabado de forma satisfactoria.
Condiciones de Ejecución: La aplicación debe de estar disponible, el servidor Asterisk debe de estar activo y debe de existir un teléfono disponible.
Entrada / Pasos de ejecución: El usuario accede a Gestionar SACT, presiona la opción "Adicionar" y llena todos los campos, luego escoge la opción "Grabar Mensaje", el teléfono perteneciente al número introducido timbra, al ser levantado el megáfono son escuchadas las instrucciones para grabar el mensaje, al terminar estas, se procederá a grabar el mensaje.
Resultado Esperado: El sistema graba el mensaje de forma satisfactoria.
Evaluación de la Prueba: Satisfactoria

Tabla 57 Prueba de aceptación: HU Grabar Mensaje de compromiso

Caso de Prueba de Aceptación	
Código Caso de Prueba: SACT-04-02	**Nombre Historia de Usuario:** Grabar Mensaje de Compromiso
Nombre de la persona que realiza la prueba: Yeisell G. Medina	
Descripción de la Prueba: Comprobar que el mensaje de solicitud de compromiso no es grabado de forma satisfactoria.	
Condiciones de Ejecución: La aplicación debe de estar disponible, el servidor Asterisk debe de estar activo y debe de existir un teléfono disponible.	
Entrada / Pasos de ejecución: El usuario accede a Gestionar SACT, presiona la opción "Adicionar" y llena todos los campos, luego escoge la opción "Grabar Mensaje", el teléfono perteneciente al número introducido timbra, al ser levantado el megáfono son escuchadas las instrucciones para grabar el mensaje, el usuario no cumple las instrucciones mencionadas.	
Resultado Esperado: El sistema no graba el mensaje de forma satisfactoria.	
Evaluación de la Prueba: Satisfactoria	

Tabla 58 Prueba de aceptación: HU Enviar Mensaje de compromiso

Caso de Prueba de Aceptación	
Código Caso de Prueba:	**Nombre Historia de Usuario:** Enviar Mensaje de Compromiso

SACT-05-01	
Nombre de la persona que realiza la prueba: Yeisell G. Medina	
Descripción de la Prueba: Comprobar que el mensaje de solicitud de compromiso es enviado de forma satisfactoria.	
Condiciones de Ejecución: La aplicación debe de estar disponible, el servidor Asterisk debe de estar activo y debe de existir un teléfono disponible.	
Entrada / Pasos de ejecución: El Proceso accede a la base de datos leyendo los compromisos y ejecutando el aviso a la hora y fecha especificada.	
Resultado Esperado: El sistema envía de forma satisfactoria el mensaje de compromiso.	
Evaluación de la Prueba: Satisfactoria	

Tabla 59 Prueba de aceptación: HU Enviar Mensaje de compromiso

Caso de Prueba de Aceptación	
Código Caso de Prueba: SACT-05-01	**Nombre Historia de Usuario:** Enviar Mensaje de Compromiso
Nombre de la persona que realiza la prueba: Yeisell G. Medina	
Descripción de la Prueba: Comprobar que el mensaje de solicitud de compromiso no es enviado de forma satisfactoria.	
Condiciones de Ejecución: La aplicación debe de estar disponible, el servidor Asterisk debe de estar activo y debe de existir un teléfono disponible.	
Entrada / Pasos de ejecución: El Proceso accede y se encuentra con las siguientes situaciones: el servidor de Asterisk o base de datos están caídos o el teléfono no está registrado.	
Resultado Esperado: El sistema no envía el mensaje de compromiso y guarda en un log la información del error.	
Evaluación de la Prueba: Satisfactoria	

Tabla 60 Prueba de aceptación: HU Repetir Mensaje de compromiso

Caso de Prueba de Aceptación	
Código Caso de Prueba: SACT-06-01	**Nombre Historia de Usuario:** Repetir Mensaje de Compromiso
Nombre de la persona que realiza la prueba: Yeisell G. Medina	
Descripción de la Prueba: Comprobar que el mensaje de solicitud de compromiso es repetido de	

forma satisfactoria.
Condiciones de Ejecución: La aplicación debe de estar disponible, el servidor Asterisk debe de estar activo y debe de existir un teléfono disponible.
Entrada / Pasos de ejecución: El usuario no levantó el manófono en los sesenta segundos que tiene después del primer timbrazo, el sistema procede a repetir el mensaje de compromiso cada sesenta segundos.
Resultado Esperado: El sistema repite el mensaje de compromiso.
Evaluación de la Prueba: Satisfactoria

Tabla 61 Prueba de aceptación: HU Terminar Mensaje de compromiso

Caso de Prueba de Aceptación	
Código Caso de Prueba: SACT-07-01	**Nombre Historia de Usuario:** Terminar Mensaje de Compromiso
Nombre de la persona que realiza la prueba: Yeisell G. Medina	
Descripción de la Prueba: Comprobar que el mensaje de solicitud de compromiso es terminado de forma satisfactoria.	
Condiciones de Ejecución: La aplicación debe de estar disponible, el servidor Asterisk debe de estar activo y debe de existir un teléfono disponible.	
Entrada / Pasos de ejecución: El usuario levantó el manofóno en los sesenta segundos que tiene después del primer timbrazo y el sistema deja de enviar el mensaje o el usuario no levantó el manofóno en los sesenta segundos que tiene después del primer timbrazo por tanto, el sistema procedió a repetir el mensaje de compromiso, pero el teléfono nunca fue levantado, entonces el sistema deja de enviar el compromiso.	
Resultado Esperado: El sistema termina el mensaje de compromiso.	
Evaluación de la Prueba: Satisfactoria	

Glosario de Términos

Asterisk: Es una aplicación de software libre que proporciona funcionalidades de una central telefónica (PBX). Como cualquier PBX, se puede conectar un número determinado de teléfonos para hacer llamadas entre sí e incluso conectar a un proveedor de VoIP o bien a una tarjeta RDSI tanto básicos como primarios.

Release: Se refiere a un producto final, preparado para lanzarse como versión definitiva a menos que aparezcan errores que lo impidan.

DOM: Es esencialmente una interfaz de programación de aplicaciones que proporciona un conjunto estándar de objetos para representar documentos HTML y XML, un modelo estándar sobre cómo pueden combinarse dichos objetos, y una interfaz estándar para acceder a ellos y manipularlos.

Framework: es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado.

IDE: Es un programa compuesto por un conjunto de herramientas para un programador desde el que se pueden editar programas, compilarlos y depurarlos.

GPL: La GNU General Public License (inglés: Licencia Pública General) es una licencia creada por la Free Software Foundation y orientada principalmente a los términos de distribución, modificación y uso de software. Su propósito es declarar que el software cubierto por esta licencia es Software Libre.

RAM: Memoria de acceso aleatorio. Tipo de memoria donde la computadora guarda información para que pueda ser procesada más rápidamente. En la memoria RAM se almacena toda información que está siendo usada en el momento.

Protocolo: Conjunto de normas que rigen un determinado proceso de comunicación.

MVC: Modelo Vista Controlador.

HU: Historia de Usuario