Universidad de las Ciencias Informáticas

Facultad 2

SISTEMA INFORMÁTICO PARA LA GESTIÓN DE AMPLIOS VOLÚMENES DE GRABACIONES DE LLAMADAS EN EL CALL CENTER DEL ELASTIX.

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN CIENCIAS INFORMÁTICAS

Autores:

Irma Margarita Rodríguez Ruiz

Boris Orniel Pita Barrientos

Tutores:

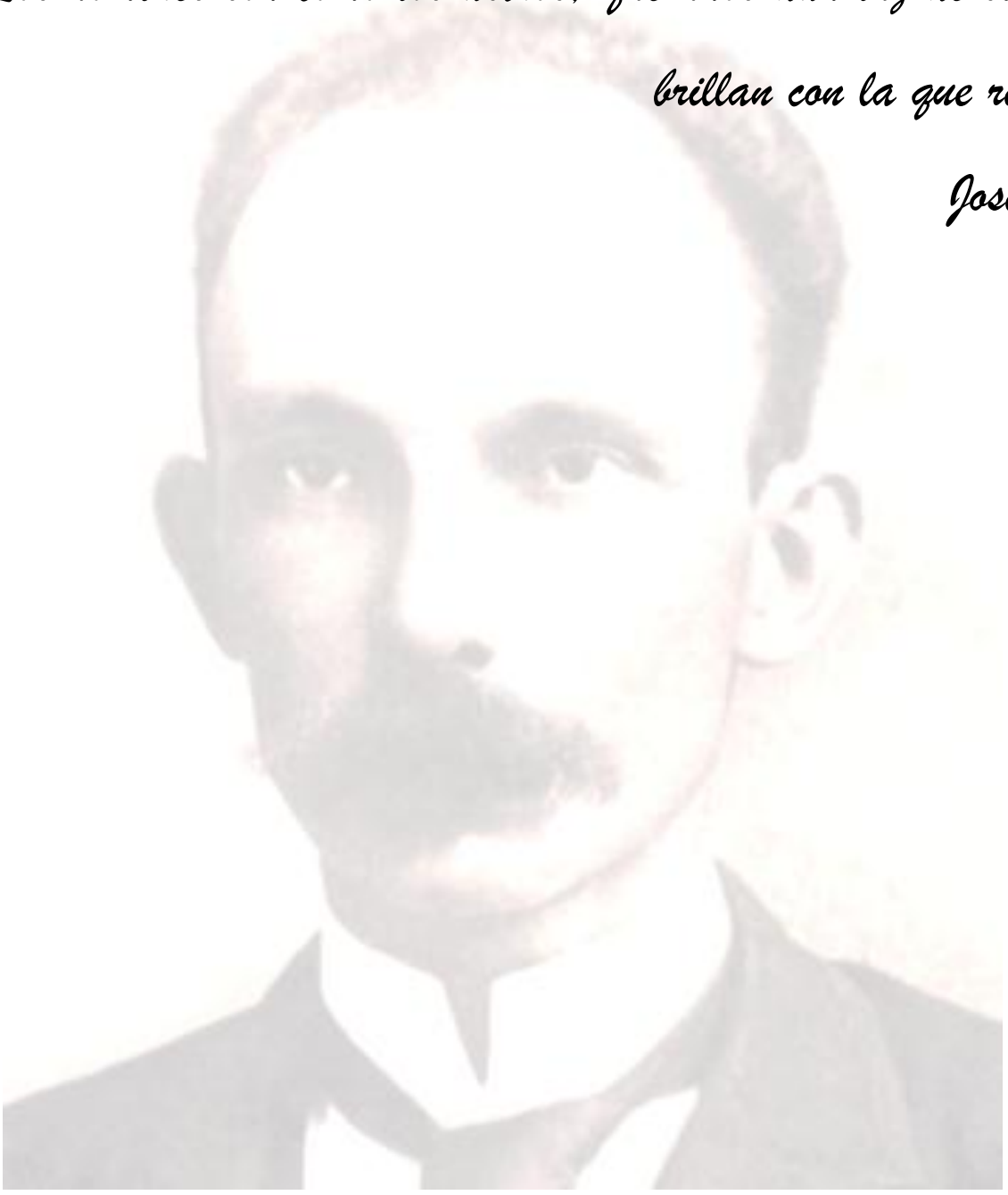
Ing. Arianna Pérez Carmenates

Ing. Alejandro Rodríguez Reyes

Universidad de las Ciencias Informáticas, La Habana, 2013.

"Los hombres son como los astros, que unos dan luz de sí y otros brillan con la que reciben."

José Martí



DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor del presente trabajo y autorizo a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

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

_____ _____

Irma Margarita Rodríguez Ruiz
Firma Autor

Boris Orniel Pita Barrientos
Firma Autor

_____ _____

Arianna Pérez Carmenates
Firma Tutor

Alejandro Rodríguez Reyes
Firma Tutor

DATOS DE CONTACTO

acarmenates@uci.cu Ing. Arianna Pérez Carmenates, graduada de Ingeniera en Ciencias Informáticas en el año 2008, Pertenece al centro de TLM, Dpto. Telecomunicaciones y se desempeña como Analista de la Línea de desarrollo de soluciones VoIP y *Call Center*.

arodriguezr@uci.cu Ing. Alejandro Rodríguez Reyes, graduado de Ingeniero en Ciencias Informáticas en el año 2007. Pertenece al centro de TLM, Dpto. Telecomunicaciones y se desempeña como Arquitecto de la Línea de desarrollo de soluciones VoIP y *Call Center*.

A la persona que ha sido capaz de darlo todo sin recibir nada. A ti que has mantenido la confianza en mí .A ti mami.

A mi hermana Mirelis para que se inspire en mí.

A mis abuelos que también se merecen este triunfo porque han sido como mis padres. A mi abuela Dignora que no se encuentra entre nosotros pero donde quiera que estés, siéntete orgullosa de tu nieto.

A tía Lesvia, mi segunda madre, también muchas gracias. Contigo he aprendido mucho y he vivido suficiente como para saber que eres de oro.

Gracias a todos los que aportaron ideas y opiniones para el desarrollo de este trabajo.

A mis compañeras de escuela Lidiagnis y Dayna por confiar en mí desde el primer momento y ser capaces de soportar mis malcriadeces, por auxiliarme en los momentos que se los he pedido y en los que se han brindado voluntarias, por preocuparse por mi docencia y por los ratos que pasamos juntos.

A todas las personas que en el transcurso de mi vida han hecho por mí y que saben que las estimo, un millón de gracias.

Boris

A aquellos gigantes que me enseñaron con su ejemplo a ser mejor cada día.

A mi mamá por su apoyo, su amor incondicional y por ser además mi mejor amiga. A mi papá que debajo de todo ese carácter serio esconde toneladas de amor y preocupación.

A mis hermanitos: Ivania e Izael por los abrazos cada vez que llego a mi casa, las malcriadeces y los clásicos pleitos entre hermanos.

A una personita especial que ha sabido conjugar paciencia, amor, risa y llanto en un solo cuerpo: mi chitico Yadir. Gracias por todos los momentos que me has ofrecido y por ser mi apoyo fundamental. Te quiere mucho, tu peque.

A toda mi familia, en especial mis abuelos Irma y Zoilo, a mis tíos y todos mis primos que saben que los quiero muchísimo.

A Noel por todo el tiempo que pasamos juntos y porque mi primer enfrentamiento con una tesis fue gracias a ti. No obstante descubres que es mucho más complicado cuando debes hacer la tuya propia.

Agradecimientos

A Yurisbel y Yunior por ser la familia que he podido escoger en estos últimos años, definitivamente esta universidad necesita más gente como ustedes.

A las amistades ajenas que se han vuelto propias en cuestiones de segundos: Reanna, Ruberlandy, Asiel, Arita.

A mis compañeros de aula, tanto los viejos de la facultad 1 como los nuevos de la 2.

A los chicos del proyecto: Luis Miguel, Yaisel y Luis Esteban pues no saben cuánto ha significado para mí un saludo y un abrazo de su parte en los momentos de estrés y preocupación.

A mis amistades de la facultad por los momentos vividos tanto en los festivales como con las chicas del softball y el kikimboll.

A todos los profes que han ayudado a formarme como profesional a lo largo de estos cinco años.

A mis tutores Alejandro y Arianna, así como los demás profes del proyecto por la paciencia y la dedicación que emplearon con nosotros.

A los muchachos del 95 por aguantarme durante muchos meses allá con ellos y tratarme como a uno más.

A mi compañero de tesis, Boris, que ha compartido conmigo todo este tiempo de trabajo.

A la universidad por regalarme los mejores recuerdos de mi vida en estos cinco años. Aquí he reído, llorado, he disfrutado del amor, he aprendido a ser actriz, deportista y por sobre todas las cosas que todo esfuerzo trae asociado una recompensa.

Irma.

A esas dos personitas especiales en mi vida que me han enseñado a esforzarme y a luchar por alcanzar mis sueños: mi mamá y mi papá.

Irma.

A mis padres con todo mi amor, por haber dedicado toda su vida a mi formación y no haber escatimado esfuerzos por ver realizado este que es nuestro gran sueño.

Boris.

RESUMEN

Elastix es un ejemplo de distribución orientada a las Comunicaciones Unificadas que posee funcionalidades propias y permite la creación e incorporación de módulos desarrollados por terceros. Esta plataforma cuenta con el módulo *Call Center* (o centro de atención de llamadas), encargado de gestionar los centros de llamadas masivas en los que interactúan agentes y abonados. Elastix posee un subsistema encargado de gestionar las grabaciones de las llamadas, sin embargo el mismo no cumple con una serie de requisitos relacionados con el filtrado de las grabaciones y el manejo de grandes volúmenes de las mismas. El presente trabajo se desarrolló con el objetivo de implementar un nuevo Sistema de Gestión de Grabaciones de llamadas encargado de mostrar información que anteriormente no era manejada, así como brindar facilidades para la obtención de la misma incorporando filtros de búsquedas. La solución resuelve además, el manejo adecuado de grandes volúmenes de llamadas; resultado que agrega un conjunto de funcionalidades que ayudan en la consolidación del sistema Elastix como un paquete de software disponible para la telefonía de código abierto.

PALABRAS CLAVE

Call Center, Elastix, Grabaciones, Llamadas.

TABLA DE CONTENIDOS

ÍNDICE DE TABLAS:

ÍNDICE DE IMÁGENES

INTRODUCCIÓN

Con la necesidad de las empresas de telefonía de agilizar el proceso de gestión de las llamadas surgen las centrales telefónicas. Tradicionalmente eran utilizadas para el servicio telefónico, sin embargo el desarrollo alcanzado ha permitido que estas se empleen además en la transmisión de datos. Como una solución ideal a este proceso de transmisión e integración de datos se emplean las Comunicaciones Unificadas. Estas llevan la convergencia más allá de la telefonía IP, integrando presencia, mensajería instantánea, mensajería unificada, conferencia *Web* y videoconferencia en una plataforma común, accesible desde cualquier medio (1).

Elastix, como distribución libre de Comunicaciones Unificadas, es una de las soluciones que presenta un alto desarrollo dentro del ámbito empresarial, que cuenta con tecnología de punta en cuanto a la integración de los principales medios de comunicación, tales como: video conferencias, mensajería unificada, *email*, *fax,* mensajería instantánea. Sus funcionalidades están implementadas sobre cuatro programas de software: HylaFAX[1], OpenFire[2], PostFix[3] y Asterisk. Estos brindan las funciones de fax, mensajería instantánea, correo electrónico y PBX[4] respectivamente. Esta última no es más que una central telefónica con el objetivo de gestionar las llamadas internas, entrantes y salientes atendidas en la misma. Elastix añade además su propio *kit* de desarrollo que permite la creación de módulos de terceros, dando lugar a un excelente paquete de software disponible de código abierto para la telefonía. Entre los ya implementados se encuentra el módulo de *Call Center* que no es más que un paquete de software diseñado para asistir en la implementación de centros de llamadas masivas con interacción entre los operadores de la central telefónica y los clientes, estos últimos contactados mediante las llamadas telefónicas (2).Asterisk, por su parte, posee una serie de funcionalidades ya implementadas, entre las que se encuentra un módulo para grabar y almacenar, todas las llamadas que son atendidas en el *Call Center*.

La Universidad de las Ciencias Informáticas, institución creada con el fin de convertirse en una entidad productora de software está desplegando el *Call Center* del Elastix en la República Bolivariana de Venezuela a través de una solución para centros de atención de emergencias. Este sistema está instalado en dos estados y se pretende instalar en al menos seis más, pues provee todas las funcionalidades necesarias para estos centros, en cada uno de los cuales se atienden de 1000 a 2000 llamadas diarias.

[1] **HylaFAX** - Software de clase empresarial para enviar y recibir fax, así como para el envío de páginas alfa-numéricas.
[2] **OpenFire** - Sistema de mensajería instantánea.
[3] **PostFix** – Servidor de correo electrónico.
[4] **PBX- (Private Branch Exchange)**: Central Privada de Conmutación Automática.

No obstante el *Call Center* no posee un módulo que se encargue de gestionar la información relacionada con las grabaciones; por lo cual se utiliza el módulo de gestión de grabaciones del Elastix. Sin embargo este posee algunos inconvenientes producto a que no está preparado para manejar un amplio volumen de grabaciones, lo que ocasiona que el sistema colapse por la sobrecarga de información y no se muestre la misma en la interfaz gráfica. Cuando Asterisk graba las llamadas y las almacena, los datos referentes a las mismas son guardados en el nombre del archivo. El sistema de gestión por su parte se encarga de listar cada una de estas grabaciones y leer sus respectivos nombres para poder extraer los datos a mostrar. Este proceso se complica cuando el sistema posee más de 70 000 grabaciones. Además el sistema no brinda las facilidades necesarias para el filtrado de la información, pues la misma sólo se visualiza seleccionando un intervalo de fecha. Tampoco muestra todos los datos referentes a una grabación, entre los que se encuentran: nombre del agente, cola de entrada de la llamada, transferencia, tiempo de espera y estado de la llamada.

Para lograr que el sistema continuara funcionando y no se sobrecargara se implementó un *script*[5] que se encarga de mover las grabaciones con más de 15 días hacia otra carpeta. Esta solución permite que el sistema de gestión de grabaciones pueda seguir trabajando aunque su rendimiento no sea el óptimo; pero constituye una solución parcial, ya que en el mismo sólo se visualizan las grabaciones de los últimos 15 días, cuando constituye una necesidad real tener conocimiento de todos los registros. Además en caso de que se necesitara buscar una grabación de una llamada de más de 15 días, hay que dirigirse a la carpeta donde se encuentran almacenadas miles de grabaciones y revisar todos los archivos hasta encontrar el deseado.

A partir de esta situación problemática se plantea como **problema a resolver**: La imposibilidad del *Call Center* del Elastix para gestionar grandes volúmenes de grabaciones de llamadas en los centros de atención a emergencia, teniendo como **objeto de estudio:** el proceso de gestión de grabaciones de llamadas. Delimitando como **objetivo general:** desarrollar un módulo que permita gestionar los grandes volúmenes de llamadas generados por los centros de atención a emergencias en Venezuela. El mismo estará enmarcado en el **campo de acción**: proceso de gestión de amplios volúmenes de grabaciones de llamadas para el *Call Center* del Elastix en los centros de atención de emergencias en Venezuela. Se establece como **idea a defender**: el módulo a desarrollar para el *Call Center* del Elastix permitirá gestionar amplios volúmenes de grabaciones de llamadas generados por los centros de atención a emergencias en Venezuela. Para dar cumplimiento al objetivo propuesto se definen las siguientes **Tareas:**

[5] ***script*** - Programa interpretado utilizado para combinar componentes e interactuar con el sistema operativo o con el usuario.

1. Análisis del funcionamiento y las características principales del módulo *Call Center*, la planta telefónica Asterisk y la plataforma web Elastix para lograr un mejor entendimiento de los mismos.
2. Análisis de sistemas de gestión de grabaciones existentes en el mundo observando ventajas y desventajas de los mismos para lograr un producto robusto y flexible que elimine las deficiencias encontradas.
3. Valoración de la metodología de desarrollo de software a utilizar, así como plataforma, lenguaje y conjunto de herramientas de desarrollo para la elaboración de la aplicación.
4. Diseño de una propuesta de solución que detalle las características y funcionalidades principales del sistema informático a desarrollar.
5. Selección de los patrones de arquitectura y diseño más apropiados para la elaboración del producto de software.
6. Implementación de las funcionalidades principales correspondientes al Sistema de Gestión de Grabaciones para el *Call Center* del Elastix.
7. Implementación de una aplicación de escritorio que permita trabajar con la información almacenada en la base de datos para garantizar que el supervisor pueda consultar la misma sin necesidad de estar conectado directamente al Elastix.
8. Realización de pruebas a cada una de las funcionalidades correspondientes al Sistema de Gestión de grabaciones de llamadas para verificar que la solución desarrollada cumpla con el objetivo propuesto y elimine las deficiencias encontradas.

El trabajo consta de introducción, cuatro capítulos, conclusiones, recomendaciones, bibliografía, anexos y glosario de términos que complementan el contenido de los diferentes capítulos.

Capítulo 1 "Fundamentación Teórica": Contiene una descripción del proceso que será automatizado y un breve estudio de los principales conceptos tratados en el trabajo. Se exponen las principales características de las tecnologías, herramientas y metodología seleccionadas para la solución del problema actual.

Capítulo 2 "Planificación y Características del Sistema": En este capítulo se hace una propuesta del sistema que consta de una modelación de los procesos. Se exponen las listas de reserva del producto, se plantean de forma general las HU[6] y por último se lleva a cabo una estimación de planificación basada en dichas HU.

Capítulo 3. "Análisis y Diseño del Sistema": Contiene la arquitectura del sistema y con ello los patrones arquitectónicos empleados. Se abordan las tarjetas CRC[7] y se especifica el modelo físico de las Bases de Datos empleadas para una mayor comprensión del sistema creado.

Capítulo 4. "Implementación y Pruebas": En este capítulo se describen las tareas de ingeniería haciendo un énfasis detallado de cada una de ellas, además de un resumen de las pruebas unitarias y de aceptación que fueron realizadas al sistema.

[6] **HU** – Historias de usuario
[7] **CRC** – Clase – Responsabilidad – Colaborador

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

En el desarrollo de este capítulo se profundiza en algunos conceptos fundamentales para el desarrollo de un módulo que permita gestionar las grabaciones de las llamadas para el *Call Center* del Elastix. También se definen un conjunto de características pertenecientes a las metodologías, herramientas y lenguajes que constituyen la base para la solución del problema existente, así como la justificación de la selección de las mismas.

1.1 Conceptos fundamentales

1.1.1 Asterisk

Dentro de la documentación general ofrecida en el sitio web Asterisk Colombia se define como:

Software para múltiples plataformas bajo los sistemas operativos Linux, BSD, Apple OSX, donde las llamadas en el sistema disparan funciones a través de patrones de dígitos (mejor conocidos como extensiones), ofreciendo un completo control sobre el enrutamiento de las mismas con relativa facilidad. Permite crear sistemas IP PBX IP, *gateways* VoIP, servidores de conferencia, entre otros (3).

Según Flavio E. Gonçalves[8]: *"Software que provee todas las características que se esperan de una PBX, utiliza el concepto de software libre y es promovido por la empresa Digium que se encarga entre otros aspectos del desenvolvimiento del código fuente y del hardware de telefonía de bajo costo que funciona con Asterisk. Entre los servicios que proporciona Asterisk se encuentran: correo de voz, conferencias, respuesta de voz interactiva, llamada en espera y grabado de llamadas"* (4).

A manera de resumen, Asterisk no es más que:

Programa de software libre bajo licencia GPL que proporciona todas las funcionalidades de una planta telefónica. Como cualquier central telefónica, se puede conectar un número determinado de teléfonos para hacer llamadas entre sí e incluso conectar a un proveedor de VoIP. Tiene varias características como son:

- Contestador automático de llamadas.
- Transferencia de llamadas.
- Monitoreo y grabación de llamadas.

[8] Director general de V.Office y Sippulse en Brasil, la primera es una empresa de consultoría dedicada al área de *Networking*, Seguridad, telecomunicaciones y la segunda está dedicada al desarrollo de un sistema descentralizado y robusto de telefonía. Se ha desempeñado por 15 años como instructor, para ayudar a la gente a aprender sobre el software de telefonía de código abierto.

- Correo de voz.
- Colas de atención.
- Llamada en espera.
- Respuesta de voz interactiva (5).

1.1.2 *Call Center*

En el libro "Plataforma de *Call Center* y servicios informativos de valor agregado" se define como: *"Conjunto de herramientas de Informática y de Telecomunicaciones que, puestas a disposición de un grupo de operadores encargados de atender llamadas telefónicas masivas, eleva la productividad de los recursos tecnológicos y de los recursos humanos"* (6).

Margaret Rouse[9] explica:

Un centro de llamadas es un lugar físico donde las llamadas telefónicas de los clientes y de otras personas son manejadas por una organización, generalmente con cierta automatización informática. Típicamente, un centro de llamadas tiene la capacidad de manejar un volumen considerable de llamadas al mismo tiempo, filtrarlas, enviarlas a alguien cualificado para su manejo y finalmente registrarlas. Son utilizados por las organizaciones del catálogo de venta por correo, empresas de *telemarketing*, así como cualquier organización que utiliza el teléfono para ventas o servicios (7).

Edgar Landívar[10] en Comunicaciones Unificadas con Elastix expone que no es más que*"(…) un paquete de software diseñado para asistir en la implementación de centros de llamadas masivas con interacción entre operadores contratados por el administrador de la central telefónica llamados agentes y personas contactadas a través de llamadas telefónicas llamados abonados (…)"* (2).

A manera de resumen un *Call Center* es un centro de atención de llamadas diseñado para manejar grandes volúmenes de llamadas telefónicas entrantes y salientes, con el propósito de brindar servicios de diversos tipos a los clientes. El *Call Center* tiene como objetivos principales generar llamadas de manera automática a números de teléfonos que previamente han sido cargados en el sistema y también permite controlar las llamadas que se reciben a través de una cola creada en la central (8).

[9] Margaret Rouse escribe y gestiona WhatIs.com. Es responsable de construir el contenido del sitio que se dedica a ayudar a los profesionales de la Informática y las Telecomunicaciones a entender conceptos y definiciones de la forma más simple y directa posible.
[10] Fundador y líder del proyecto Elastix.

1.1.3 Elastix

Conocer qué es Elastix es de vital importancia para comprender el funcionamiento y las características fundamentales de esta plataforma para la cual se está desarrollando el presente trabajo de diploma. En el sitio web oficial de Elastix se define como:

Software empresarial de código abierto distribuido bajo la licencia GPL[11] para el establecimiento de comunicaciones unificadas cuyo objetivo es el de incorporar en una única solución todos los medios y alternativas de comunicación existentes en el ámbito empresarial (9).

En el sitio web de *PaloSanto Solutions*, compañía creadora del software se describe como:

Producto ecuatoriano distribuido a nivel mundial bajo los términos de la licencia GPL. Permite a la empresa no solamente contar con una central telefónica de grandes prestaciones, también le ofrece un completo servidor unificado de comunicaciones que integra tecnologías de fax, telefonía, mensajería instantánea, correo electrónico y de colaboración en un solo equipo (10).

De manera general puede resumirse que Elastix es un software aplicativo creado como una distribución libre de Servidor de Comunicaciones Unificadas que integra las principales tecnologías y medios de comunicación dentro del sector empresarial, tal y como se puede observar en la Figura1:

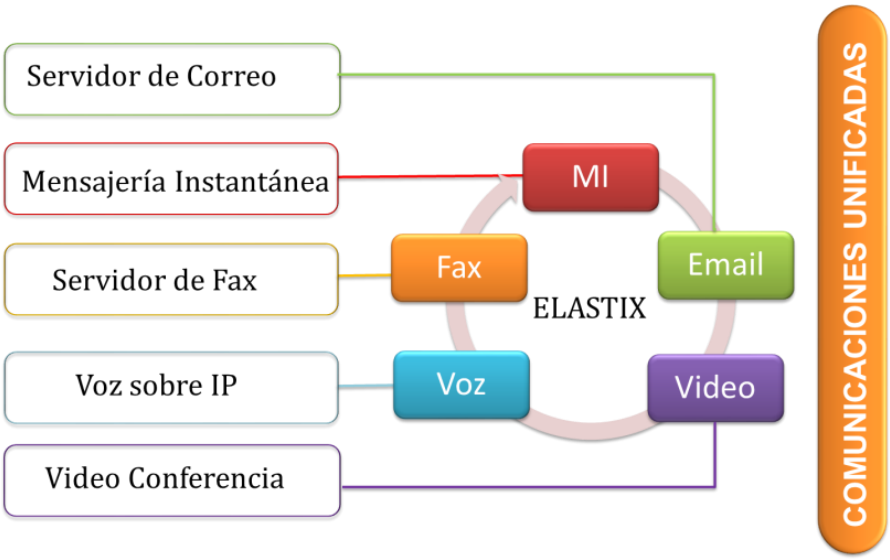


Figura 1: Principales componentes de la plataforma Elastix.

[11]**GPL – (General Public License)**: Licencia Pública General.

Algunas de las características básicas de Elastix incluyen además:

- Correo de Voz.
- Envío de *fax* a *email*.
- Interfaz de configuración Web.
- Sala de conferencias virtuales.
- Grabación de llamadas.
- Interconexión entre PBX's
- Identificación del llamante.
- Reportación avanzada (9).

1.2 Estado del arte

En el desarrollo de las TIC[12] se evidencia una tendencia al aumento de los *Call Center*; en la esfera internacional se destacan Avaya IP Office, Avanvox y Recall CTI, soluciones provistas de características únicas que los identifican y los convierten en su conjunto en una alternativa tecnológica a emplear en aquellas empresas que se dedican a la gestión de llamadas. De los mismos se expone un breve resumen a continuación.

Avaya IP Office es un sistema de comunicaciones empresariales para pequeñas y medianas empresas. Dispone de una amplia gama de aplicaciones que permiten a las empresas ofrecer una imagen profesional a los clientes. Entre estas se encuentra *ContactStore*[13]. Este complementa las capacidades de grabación de voz de *VoiceMail Pro*[14]. Almacena y clasifica las grabaciones de manera que se pueda acceder a ellas más fácilmente. Permite recuperar llamadas basándose en filtros de búsqueda que incluyen la fecha y hora, duración de la llamada, nombre y número del llamante. En la reproducción de las grabaciones, se puede observar la conversación en forma de onda para identificar fácilmente los puntos interesantes como silencios largos o elevación del volumen y se pueden exportar a formato WAV[15] (11).

A pesar de que este sistema cuenta con varias ventajas analizadas anteriormente no se recomienda el uso de este como solución al problema de la investigación ya que presenta una serie de inconvenientes

[12] **TIC**- Tecnologías de la Información y las Comunicaciones.
[13] **ContactStore** - Herramienta para la gestión de calidad de servicio y grabaciones de llamadas.
[14] **VoiceMail Pro** - Solución integral de mensajería y operadora automática con grabación de conversaciones, respuesta de voz interactiva, acceso a bases de datos y función de síntesis de voz.
[15] **WAV** - Formato estándar para archivos de sonido de Microsoft Windows.

pues es un software propietario protegido por leyes de derecho de autor estadounidense e internacionales, leyes de patentes y tratado de propiedad intelectual. Ciertos programas incluidos en esta solución contienen software distribuido bajo acuerdos de terceros, lo que agrega términos que amplían o limitan los derechos de utilizar ciertas partes del mismo. El propietario de la llamada es el número de la extensión que grabó la llamada por lo cual los permisos para acceder a la información son restringidos.

Avanvox es una solución avanzada para centrales IP basada en Asterisk, FreePBX, Hylafax y Queuemetrics. Incluye una plataforma de alto rendimiento que incorpora las funcionalidades principales del sistema *VoIP*[16]. Posee un potente sistema de gestión web para la planta telefónica Asterisk y permite configurar un 90% de las opciones del mismo. Además acepta la mayoría de las interfaces de telefonía del mercado, permitiendo la conectividad e integración entre centrales telefónicas tradicionales y otras centrales IP. Entre sus funcionalidades se destaca un sistema de gestión de grabaciones de llamadas (12).

Esta solución no es apropiada para resolver el problema planteado inicialmente, pues el sistema de grabaciones no se encuentra incorporado a Avanvox en todas sus versiones y sólo es recomendable para pequeñas o medianas empresas producto a que no puede manejar un amplio volumen de grabaciones y esto constituye uno de los problemas fundamentales a los que se le debe dar solución.

Recall CTI es una solución diseñada para poder grabar todas las llamadas entrantes y salientes en las extensiones propietarias de una central telefónica pequeña haciendo uso del software *Recording Manager*. Las grabaciones pueden ser realizadas bajo la demanda de un supervisor o el propio usuario, de forma permanente o de forma selectiva, son almacenadas en formato WAV en la PC[17] servidora. Esta agrega además las funcionalidades de gestión de grabaciones de llamadas que permite la búsqueda de grabaciones por tipo de llamada (entrante, saliente), duración, fecha y hora, extensión, número llamado y número llamante (13).

Después de analizar varias características detalladas anteriormente no se recomienda el uso de este software como solución al problema de la investigación ya que el mismo es una solución europea

[16] **VoIP** – Voz sobre IP.
[17] **PC** – (Personal Computer): Computadora Personal.

propietaria bajo licencia CTI que solamente guarda y gestiona las llamadas que se realizan a través de extensiones propietarias; cuando en realidad se necesitan todas las grabaciones.
Aunque estas soluciones resuelven aspectos importantes como son la gestión de las grabaciones a partir de filtros, cuentan con una serie de inconvenientes ya detallados con anterioridad. Por ende el siguiente trabajo de diploma pretende realizar un sistema de gestión de grabaciones para el *Call Center* del Elastix que brinde toda la información de una manera más detallada y útil acorde con las exigencias del cliente.

1.3 Metodología, Lenguajes y Herramientas de desarrollo

1.3.1 Metodología de desarrollo seleccionada

Existen numerosas propuestas de metodologías que de una forma u otra inciden en el proceso de desarrollo de software. Estas surgen ante la necesidad de utilizar una serie de procedimientos, técnicas, herramientas y soporte documental a la hora de desarrollar un producto determinado. Dichas metodologías pretenden guiar a los desarrolladores indicando ¿qué personas deben participar en el desempeño de las actividades? y ¿qué papel deben jugar en las mismas?

Las metodologías detallan la información que debe producirse como resultado de las actividades, garantizando así la calidad del trabajo a realizar. Estas se clasifican en dos grandes grupos: las orientadas al control de los procesos, denominadas Metodologías Pesadas y las orientadas a la interacción con el cliente y el desarrollo incremental del software, denominadas Metodologías ligeras o ágiles. Estas últimas muestran versiones parcialmente funcionales del software al cliente en intervalos cortos de tiempo, para que este pueda evaluar y sugerir cambios en el producto según se vaya desarrollando. Están especialmente orientadas para proyectos pequeños aportando una elevada simplificación que a pesar de ello, no renuncia a las prácticas esenciales para asegurar la calidad del producto.

Dadas estas características se decide utilizar XP para el desarrollo del presente trabajo de diploma. Resaltar además que los requisitos tienden a cambiar frecuentemente, conforme avanza el trabajo, el cliente puede agregar nuevas HU (Historias de Usuario), dividirlas o simplemente eliminarlas. El cliente forma parte del equipo de desarrollo, logrando una mejor retroalimentación y corrección de errores; permitiendo obtener un producto que satisfaga todas las necesidades del mismo. Además no se necesita la generación de tanta documentación y el trabajo está centrado en ser desarrollado en el menor tiempo posible.

1.3.6.1 Programación Extrema (XP)

XP es una de las metodologías de desarrollo de software caracterizada por ser flexible y de riesgo mínimo. La misma está centrada en potenciar las relaciones interpersonales como clave para el éxito en el desarrollo de software; promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores y propiciando un buen clima de trabajo. XP se basa en la retroalimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y versatilidad para enfrentar los cambios. Se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, donde existe un alto riesgo técnico (14). XP abarca un conjunto de reglas y prácticas que ocurren en el contexto de cuatro actividades del marco de trabajo: planificación, diseño, codificación y pruebas, las cuales puede observarse en la siguiente figura (15):

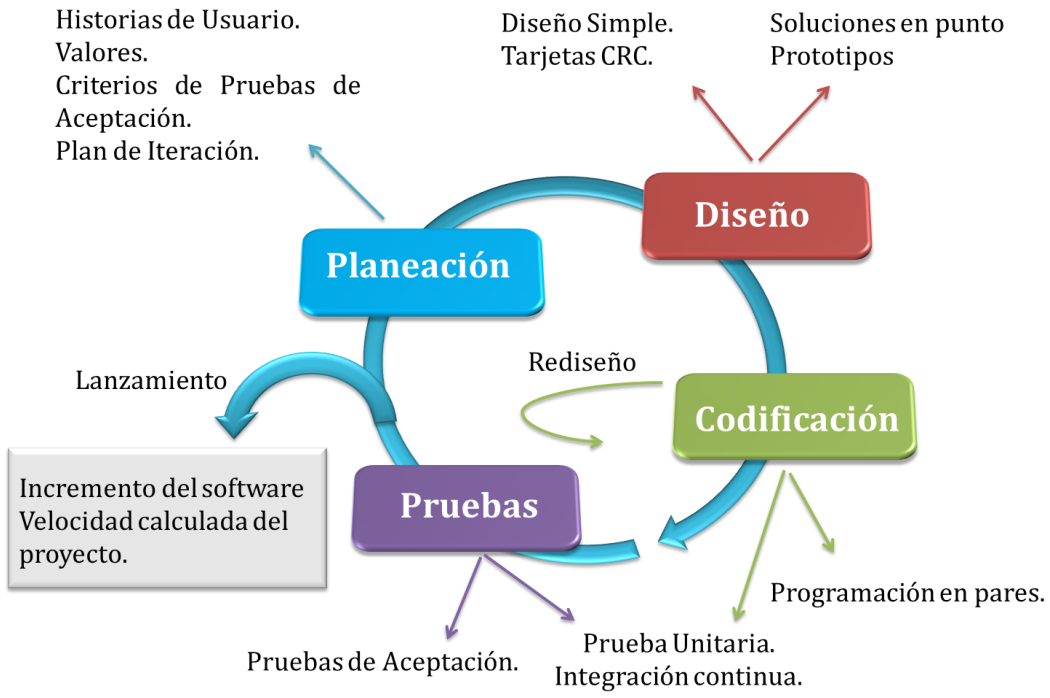


Figura 2: Actividades del marco de trabajo de XP.

1.3.2 Herramienta de Modelado

Las herramientas CASE[18] conforman un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores durante todos los pasos del ciclo de vida de desarrollo de un software. Existen tres tipos de herramientas CASE: las de Alto Nivel, que son las que apoyan las fases iniciales del ciclo de vida del desarrollo del software, las de Bajo Nivel, que apoyan solamente las fases finales y finalmente las de Cruzado de Ciclo de vida, que tienen lugar a lo largo de todo el proceso de desarrollo. Ésta última clasificación es la más recomendada, ya que asegura guiar y entender el proceso en todo momento, en este caso entran Visual Paradigm y Rational Rose (16).

En la realización de este trabajo de diploma se emplea esta última clasificación y para determinar la herramienta a utilizar se midieron características tales como: código abierto, tipo de licencia, lenguaje de programación utilizado, integración con entornos de desarrollo, coste, versión de UML, diagramas que soporta, capacidades de ingeniería inversa y requisitos de instalación. Aunque Visual Paradigm y Rational Rose satisfacen la calidad global deseada, se decide utilizar Visual Paradigm para UML por la estabilidad de ejecución en diferentes sistemas operativos, la facilidad de abrir y trabajar con un modelo UML utilizando el mismo programa sin importar el sistema operativo y sin afectar en absoluto el trabajo realizado. Destacar además que esta herramienta guarda todo el modelo en un solo fichero y con copiar el mismo se garantiza tener todo el trabajo encapsulado en él.

Visual Paradigm se empleó en la modelación de la base de datos y en la transformación de los diagramas de Entidad-Relación a tablas de base de datos, posibilitando la creación y diseño del modelo de datos.

1.3.3 Gestor de Base de Datos

1.3.3.1 MySQL

Para este trabajo de diploma se emplea como gestor de base de datos, MySQL en su versión 5.0.77, debido a que este es utilizado por el módulo de *Call Center* para la cual se desarrollará el sistema de gestión de grabaciones de llamadas.

MySQL es un sistema de gestión de bases de datos multiusuario y relacional. Es muy utilizado en aplicaciones web, sobre plataformas Linux/ Windows/ FreeBSD. Cuenta con *APIs*[19] para C, C++, Java,

[18] **CASE**– (Computer Aided Software Engineering): Ingeniería de Software Asistida por Computadoras.
[19] **API** – (Aplication Program Interface): Interfaz de Programación de Aplicaciones.

Perl, PHP[20], Python, Ruby, entre otros. Entre sus características resaltan: Soporte completo para las cláusulas y operadores SQL[21], un sistema de privilegios y contraseñas flexible y seguro, soporte para amplios volúmenes de información dentro de la BD, el servidor brinda mensajes de error a los clientes en varios idiomas y posee un sistema de reserva de memoria muy rápido (17).

1.3.3.2 SQLite

Para salvar los datos con los que estará trabajando el supervisor en la aplicación de escritorio se emplea SQLite como sistema gestor de BD. Este, a diferencia de los sistemas de gestión de bases de datos cliente-servidor, no es un proceso independiente con el que el programa principal se comunica si no que su biblioteca se enlaza con el programa pasando a ser parte integral del mismo. Entre sus principales características se destacan:

- Varios procesos o hilos pueden acceder a la misma base de datos sin problemas.
- No posee configuración, esto significa que de la forma en que fue creado y diseñado SQLite, no necesita ser instalado.
- Es portable, lo que significa que puede ser ejecutado en diferentes sistemas operativos, como Windows, Linux, BSD, Mac OS X, Solaris y sus bases de datos pueden ser fácilmente portadas sin ninguna configuración o administración.
- Se ejecuta en muchas plataformas, ofrece buen rendimiento pues realiza operaciones de manera eficiente y es más rápido que MySQL y PostgreSQL.
- Es de dominio público, y por tanto, es libre de utilizar para cualquier propósito sin costo y se puede redistribuir libremente (18).

1.3.4 Servidor web

En el presente trabajo se hace uso del servidor web Apache producto a que Elastix emplea el mismo como servidor de aplicaciones web. Además es flexible, rápido, continuamente actualizado y adaptado a los nuevos protocolos HTTP. Funciona en una multitud de sistemas operativos, lo que lo hace prácticamente universal. Es una tecnología gratuita y de código fuente abierto, es un servidor altamente configurable, de diseño modular, trabaja con gran cantidad de lenguajes como Perl, PHP, posee soporte

[20] **PHP** – (Hypertext Preprocessor): Pre-procesador de Hipertexto.
[21] **SQL** – (Structured Query Language): Lenguaje de Consulta Estructurado.

para la creación de páginas web dinámicas, permite personalizar la respuesta ante los posibles errores que puedan ocurrir en el servidor y es altamente configurable en la creación y gestión de *logs* (19).

1.3.5 Lenguajes de desarrollo

C++

Se decide implementar una aplicación de escritorio para garantizar que el supervisor pueda trabajar con la información almacenada en el Elastix sin necesidad de estar conectado directamente a la central. Para ello se selecciona como lenguaje de desarrollo C++. Esta selección se basa en características del lenguaje tales como:

- Lenguaje de programación orientado a objetos.
- Es muy potente y robusto en lo que se refiere a creación de sistemas complejos.
- Permite elaborar desde aplicaciones sencillas hasta sistemas operativos en dependencia del manejo del lenguaje.
- Existe gran cantidad de documentación que facilita su empleo y entendimiento, tales como: tutoriales en línea, libros y códigos fuentes abiertos.
- Existen muchos algoritmos cuyo pseudocódigo se encuentra ya desarrollado en C++, de manera que pueden reutilizarse y amoldarse a la solución.

Otras de las características a resaltar son el hecho de que abarca tres paradigmas de la programación: la programación estructurada, la programación genérica y la programación orientada a objetos. Proporciona facilidades para la programación orientada a objetos y para el uso de plantillas o programación genérica (20).

1.3.5.1 Lenguaje de desarrollo del lado del servidor

PHP v5.1.6

PHP es un lenguaje de programación interpretado, diseñado originalmente para la creación de páginas web dinámicas. Se usa principalmente para para ser interpretado del lado del servidor, pero puede ser utilizado desde una interfaz de línea de comandos o en la creación de otros tipos de programas incluyendo aplicaciones con interfaz gráfica (21). Es un lenguaje de código abierto y de alto nivel, muy popular y especialmente adecuado para el desarrollo web. Por ello, está dotado de un gran número de funciones que simplifican enormemente tareas habituales como descargar documentos, enviar correos, el

trabajo con cookies y sesiones. Permite la integración con múltiples BD tales como MySQL, PostgreSQL, Oracle, dbm, filePro, interbasem, entre otras (19).

El *framework* de Elastix, NEO, utiliza en su programación el lenguaje PHP aprovechando así todas y cada una de las propiedades que este brinda. Elastix establece una estándar de codificación para PHP permitiendo mejor legibilidad en el código, de esta forma los desarrolladores no tendrán problema en entender el código de otros desarrolladores que quieran colaborar.

1.3.5.2 Lenguaje de desarrollo del lado del cliente

JavaScript

JavaScript es un lenguaje de programación interpretado. Se utiliza principalmente del lado del cliente, es interpretado por el navegador web y brinda mejoras tanto en la interfaz de usuario como en las páginas web. Es soportado por la mayoría de los navegadores modernos como Internet Explorer, Netscape Navigator, Opera, Mozilla Firefox (22).

En el presente trabajo se utiliza este lenguaje principalmente para aumentar el nivel de interactividad en los entornos cliente de la aplicación, realizar modificaciones a la GUI[22] desde el entorno cliente sin tener que otorgarle esa responsabilidad al servidor y controlar los recursos del navegador web que son necesarios para el desarrollo del sistema, tales como AJAX y la librería encargada de reproducir multimedia.

CSS[23]

Lenguaje que describe la presentación de los documentos estructurados en hojas de estilo para diferentes métodos de interpretación, es decir, describe cómo se va a mostrar un documento en pantalla. Permite la separación de los contenidos de los documentos escritos en HTML, XML, XHTML de la presentación del documento, incluyendo elementos tales como: colores, fondo, márgenes, bordes, tipos de letra, etc. De esta forma es más sencillo modificar la apariencia de una página web, permitiendo a los desarrolladores controlar el estilo y formato de los documentos (23). Estas hojas de estilo permiten en este trabajo seguir las pautas del diseño de la interfaz web de Elastix.

[22] GUI – (Graphic User Interface): Interfaz Gráfica de Usuario.
[23] **CSS** – (Cascading Style Sheets): Hojas de Estilo en Cascada.

1.3.6 *Framework* de Desarrollo

Framework Neo

El *framework* usado para el desarrollo de la aplicación es el Neo y se caracteriza por:

- Una arquitectura web basada en Modelo Vista Controlador.
- Desarrollo de la programación orientada a objetos.
- Concepto e implementación modular.
- Mantenimiento de lenguajes.
- Mantenimiento de la ayuda embebida.
- Patrones de desarrollo (24).

Existen otras características por las cuales se hizo la selección del mismo para la construcción del sistema tales como: la posesión de tipos de datos especiales, validaciones automáticas, recursos de autorización y autenticación, informes y otras herramientas utilitarias que facilitan y optimizan las operaciones. Además la interfaz web del Elastix está construida con este marco de trabajo y como la aplicación se va a desarrollar para este, se opta por su uso para estandarizar el producto.

Framework QT

Luego de haberse seleccionado el lenguaje C++ para la implementación de la aplicación de escritorio, se realizó un pequeño estudio sobre los *frameworks* existentes que potencian dicho lenguaje, encontrándose: C++Builder, wxWidget y Qt. El primero posee licencia comercial por lo cual no es recomendable para el desarrollo de la herramienta, debido a que no se cuenta con los recursos económicos necesarios para su adquisición. Los dos últimos poseen gran similitud en el logro de los objetivos planteados, ambos son de código abierto y de libre adquisición, poseen soporte para: Windows, Linux, Unix, OS X. No obstante wxWidget no posee el mismo nivel de aceptación que Qt, esto viene dado por el hecho de que QT posee un amplio apoyo, tanto de empresas como de usuarios, lo cual garantiza la existencia de gran cantidad de documentación y soporte para el mismo. QT no solo permite crear la interfaz gráfica sino que ofrece toda una API de la biblioteca, con métodos para acceder a bases de datos mediante SQL, así como uso de XML, gestión de hilos, soporte de red, manipulación de archivos, etc. Estas constituyen algunas de las razones fundamentales que se tuvieron en cuenta para la selección del mismo. Cabe resaltar además que este es la base del entorno de escritorio KDE y que las librerías Qt ofrecen soluciones para utilizarse con otros lenguajes tales como:

- Java (QJambi).
- Python (PyQt).
- Ruby.
- JavaScript (módulo QtScript).
- PHP (25).

Doctrine

Es un ORM[24] que proporciona una capa de persistencia para objetos PHP. Es una capa de abstracción que se encarga de interactuar con el SGBD. Se basa en el modelo de registro activo para trabajar con datos, en los que una clase se corresponde con una tabla de base de datos. Una característica a destacar es el bajo nivel de configuración que necesita para empezar un proyecto, producto a que puede generar clases a partir de una base de datos existente y después se pueden especificar relaciones entre las clases de la BD, así como añadir funcionalidades extras a las clases autogeneradas. Resaltar además:

- Posibilita escribir consultas de BD utilizando un dialecto de SQL denominado DQL[25] que está inspirado en Hibernate (Java).
- Posee soporte para datos jerárquicos.
- Soporte para *hooks*[26] y eventos para manejar la lógica de negocio relacionada.
- Diversos comportamientos del modelo (conjuntos anidados, internacionalización, *log*, índice de búsqueda).
- Una función "compilar" que combina varios archivos PHP del *framework* en uno solo para evitar el descenso de rendimiento que provoca incluir varios archivos PHP (26).

1.3.7 Herramientas de Desarrollo

Netbeans v7.0.1

NetBeans es un IDE[27] de código abierto, multiplataforma y que en esta versión ya integra los lenguajes de programación: PHP, HTML y JavaScript para poder editar archivos de este tipo dentro del mismo. Posee un sistema para examinar todo los directorios de cada proyecto, haciendo reconocimiento y carga de clases, métodos y objetos para acelerar la programación. El editor de PHP es ágil y a la vez robusto,

[24] **ORM** – (Object-Relational Mapping): mapeador de objetos-relacional.
[25] **DQL** – (Doctrine Query Language): lenguaje de consulta Doctrine.
[26] **Hooks:** Métodos que pueden validar o modificar las escrituras y lecturas de la base de datos.
[27] **IDE**– (Integrate Development Environment): Entorno Integrado de Desarrollo.

contiene ayuda en línea y reconocimiento de sintaxis. El IDE de NetBeans para PHP también ofrece la línea de comandos de depuración: La salida del programa PHP aparece en una pantalla de línea de comandos en el IDE y se puede inspeccionar el código HTML generado sin tener que cambiar a un navegador. NetBeans brinda la posibilidad de contar con la integración de sistemas de control de versiones, tales como SVN[28], CVS[29], Mercurial y Git (27).

Qt Creator: Es un IDE multiplataforma para C++ que forma parte de Qt SDK. Incluye un *debugger* visual y un editor de Interfaces Gráficas de Usuario y de formularios. Contiene varias facilidades entre las que incluyen el sobresaltado de sintaxis y el auto completamiento. Qt Creator usa el compilador de C++ de la colección de compiladores de GNU en Linux y BSD. En Windows puede usar los compiladores MinGW o MSVC que forman parte de la propia instalación del producto. Qt Creator provee soporte para construir y ejecutar aplicaciones de Qt para entornos de escritorio (Windows, Linux, FreeBSD y MacOS) y para dispositivos móviles (Symbian, Maemo y Meego). Las opciones de ensamblado permiten cambiar los objetivos del mismo. Cuando se construye una aplicación para un dispositivo móvil conectado a una computadora, Qt Creator genera un paquete de instalación, lo instala en el dispositivo y lo ejecuta (28).

1.4 Conclusiones del capítulo

El análisis de las tendencias en cuanto a la implementación de los sistemas de gestión de grabaciones en el campo de la telefonía permitió a los autores determinar como tecnologías adecuadas a emplear aquellas basadas en software libre, a partir de lo cual se garantiza y se dispone de las herramientas para gestionar grandes volúmenes de grabaciones de llamadas en los centros de atención de emergencias.

[28] **SVN – S**istema de revisión de versiones Subversion.
[29] **CVS** – (Concurrent Versions System): Aplicación informática que implementa un sistema de control de versiones.

CAPÍTULO 2: PLANIFICACIÓN Y CARACTERÍSTICAS DEL SISTEMA

En el presente capítulo se elabora una propuesta del sistema a implementar detallando sus principales características para lograr un mejor entendimiento del mismo. Se identifican y describen los principales procesos y artefactos generados, teniendo en cuenta la planificación del tiempo y el esfuerzo en las fases posteriores. Con vista a documentar los procedimientos y técnicas empleados, se detallan las HU para cada iteración definida, en correspondencia con la metodología de desarrollo seleccionada, proporcionando así una mejor visión sobre lo que el cliente desea.

2.1 Propuesta del sistema informático

La solución propuesta está diseñada para los centros de atención de emergencias en Venezuela, cuyos servicios telefónicos se gestionan mediante la interfaz web de Elastix y utilizan Asterisk como software que provee las funcionalidades de una central telefónica. El presente módulo de gestión de amplios volúmenes de grabaciones de llamadas propone desarrollar un sistema cuyo objetivo principal es lograr una mejor manipulación y control de las llamadas recibidas en los centros de atención de emergencia en Venezuela.
La presente solución propone desarrollar un sistema integrado por tres subsistemas que automatizarán varios procesos por separado, tal y como se muestra en la Figura 3.

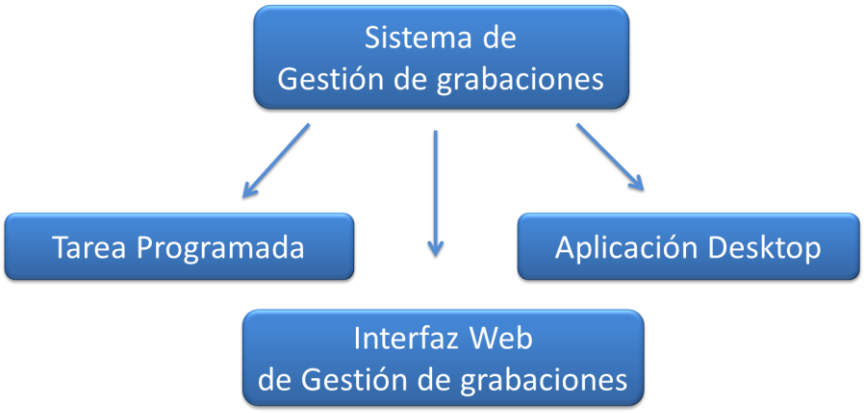


Figura 3: Propuesta del Sistema.

2.1.1 Tarea Programada

Está centrada en inspeccionar los registros de grabaciones generados por la PBX Asterisk y moverlos hacia una nueva estructura de directorios basados en fecha (Ej. Año/Mes/Día), con el objetivo de agilizar

el proceso de búsqueda de las grabaciones. La tarea programada actuará de forma periódica cada un minuto y una vez modificada la dirección asociada a una grabación se actualizarán las tablas que guardan la información de las llamadas (record, *call* y *call_entry*).

2.1.2 Interfaz Web

La interfaz web del módulo de gestión se encarga de la gestión de las grabaciones, permitiéndole al usuario realizar varias operaciones tales como: filtrar, reproducir y descargar registros de grabaciones. De manera general este subsistema gestiona datos que no eran manejados anteriormente, entre los que se encuentran: nombre del agente, cola de entrada de la llamada, transferencia, tiempo de espera y estado de la llamada. El supervisor podrá generar además reportes en varios formatos (excel, pdf y csv) y crear un archivo compactado con los datos almacenados en el Elastix (información existente en la base de datos y los archivos de audio) para trabajar con estos en una aplicación de escritorio.

2.1.3 Aplicación de escritorio

Este subsistema permitirá a los supervisores importar la información que fue filtrada y compactada con anterioridad. Los datos serán mostrados en formato de tabla en la ventana principal de la aplicación y el supervisor podrá filtrar la información, agregarla a la base de datos, visualizar todos los datos que están almacenados en la misma y reproducir la grabación asociada a una llamada.

2.2 Funcionalidades del Sistema de Gestión de Grabaciones

Los requerimientos funcionales son las capacidades o condiciones que el sistema debe cumplir (29). A continuación se muestran las funcionalidades que desarrolla el presente trabajo:

- Organizar las grabaciones.
- Filtrar y mostrar los datos de las grabaciones.
- Cargar dinámicamente los valores de los filtros.
- Reproducir grabación.
- Descargar grabación.
- Exportar datos de grabaciones.
- Crear archivo compactado.
- Importar grabaciones.

2.3 Personas relacionadas con el sistema de Gestión de Grabaciones

Se define como persona relacionada con el sistema aquel que administra todo el Sistema de Gestión de Grabaciones y obtiene un resultado de todos los procesos que se ejecutan en el sistema. En este caso, el Supervisor es la persona encargada de trabajar con el Sistema de Gestión de Grabaciones a través de la interfaz web del Elastix. También podrá hacer uso de los datos exportados para trabajar con ellos en la aplicación de escritorio.

2.4 Listas de Reserva del producto

Las listas de reserva del producto en una aplicación constituyen las cualidades que todo sistema debe poseer para su correcto funcionamiento. Para la realización de la solución propuesta se deben tener en cuenta los siguientes requerimientos:

2.4.1 Usabilidad

Se necesita una preparación previa, aunque no extensa para operar con el sistema. Se requiere un nivel medio de conocimientos de computación, aunque el manejo de la aplicación es sencillo y permite la fácil comprensión por el usuario.

2.4.2 Portabilidad

Producto a que el sistema forma parte del módulo de *Call Center* del Elastix, este podrá ser usado solamente en una distribución funcional de dicho servidor de comunicaciones unificadas con el módulo de *Call Center* instalado.

2.4.3 Seguridad

Disponibilidad: Producto a que los dos primeros subsistemas están integrados al módulo *Call Center* del Elastix su disponibilidad depende de que la central telefónica esté operando de forma permanente y sin interrupciones. Esta característica es fundamental debido a que continuamente se obtendrán datos correspondientes a las grabaciones de las llamadas que son realizadas y registradas dentro del centro de atención a emergencias. La disponibilidad de la información a mostrar se garantiza con la ejecución periódica de la tarea programada que se encarga de actualizar los datos en la BD. Por su parte la aplicación de escritorio estará disponible en todo momento.

Integridad: la información manejada por el sistema será objeto de cuidadosa protección, siendo almacenada en una BD protegida por contraseña y para acceder a la interfaz web del sistema de gestión de grabaciones para el *Call Center* el usuario debe autenticarse previamente en el Elastix.

2.4.4 Interfaz de Usuario

La aplicación propuesta poseerá una interfaz sencilla dirigida a las personas que se relacionen con el sistema. El diseño se realiza siguiendo las pautas de la interfaz web del Elastix.

2.4.5 Hardware

Para la instalación de la aplicación se debe disponer de una computadora con tarjeta de red, 2GB de RAM o superior, 500 GB de disco duro o superior y como mínimo un microprocesador Intel Pentium 4 a 2.5 GHz o un microprocesador AMD Athlom 1500+.

2.4.6 Software

Para las PC servidoras se requiere la instalación del PHP v5.1.6, Elastix v2.2 como distribución de software libre de Comunicaciones Unificadas que integra una interfaz web compatible con el sistema operativo CentOS v5.6. Se requiere además la comunicación entre Elastix y Asterisk v1.8, así como el módulo del *Call Center* v2.3 instalado en el Elastix.

En las PC clientes como navegador se recomienda Mozilla Firefox v19.0.2, no obstante la solución está diseñada para trabajar en otros navegadores como Internet Explorer, Google Chrome, Opera y Safari. Para la reproducción de las grabaciones de las llamadas deben estar previamente instalados los códec de audio en las PC clientes.

2.4.7 Restricciones de Diseño e Implementación

Se hace uso del estándar de codificación que propone el *framework* utilizado, así como del estándar de codificación *Camel Case*.

2.5 Fase de Exploración

La fase de Exploración es la primera fase definida por la metodología XP. Es aquí donde se define el alcance real del sistema. Los clientes plantean a grandes rasgos las HU que son de interés para la primera entrega del producto. Al mismo tiempo el equipo de desarrollo se familiariza con las herramientas, tecnologías y procesos que se utilizan en el proyecto. Se prueba la tecnología y se exploran las

posibilidades de la arquitectura del sistema construyendo un prototipo. La fase de exploración toma de pocas semanas a pocos meses, dependiendo del tamaño y familiaridad que tengan los programadores con la tecnología (30).

Historias de usuario

Las HU son la forma en que se especifican en XP los requisitos funcionales del sistema. Son escritas por el cliente en un lenguaje no técnico sin hacer mucho énfasis en los detalles. En estas no se debe hablar de posibles algoritmos para su implementación, ni diseños de BD adecuados; sino que son usadas para estimar el riesgo y el tiempo de desarrollo de la parte de la aplicación que describen. El cliente es el encargado de asignarle una prioridad a cada HU y el equipo de desarrollo de asignarle un costo. Durante todo el progreso del proyecto existe una estrecha comunicación entre los desarrolladores y el cliente para obtener todos los detalles necesarios. Si las historias, según lo planificado, demoran en desarrollarse se sugiere dividirlas en historias más pequeñas. También, es importante destacar, que las HU nuevas pueden describirse en cualquier momento, con esto se comprueba la flexibilidad de la metodología. Son utilizadas además en la fase de pruebas para verificar si el programa cumple con lo que especifica la HU, siendo el tiempo de desarrollo ideal para dicha historia de 1 a 3 semanas.

La prioridad en el negocio:

Alta: Se le otorga a las HU que resultan funcionalidades fundamentales en el desarrollo del sistema, a las que el cliente define como principales para el control integral del sistema.

Media: Se le otorga a las HU que resultan para el cliente como funcionalidades a tener en cuenta, sin que estas tengan una afectación sobre el sistema que se esté desarrollando.

Baja: Se le otorga a las HU que constituyen funcionalidades que sirven de ayuda al control de elementos asociados al equipo de desarrollo, a la estructura y no tienen nada que ver con el sistema en desarrollo.

El riesgo en el desarrollo:

Alta: Cuando en la implementación de las HU se considera la posible existencia de errores que lleven a la inoperatividad del sistema.

Media: Cuando pueden aparecer errores en la implementación de la HU que puedan retrasar la entrega de la versión.

Baja: Cuando pueden aparecer errores que serán tratados con relativa facilidad sin que traigan perjuicios para el desarrollo del proyecto. El cliente y el equipo de desarrollo trabajan en conjunto para definir como agrupar las HU para su lanzamiento.

Las HU son representadas mediante tablas divididas por las siguientes secciones:

Tabla 1: Maqueta de una Historia de usuario.

Historia de Usuario	
Número: (Número de la historia de usuario incremental en el tiempo)	**Nombre de Historia de Usuario:** (El nombre de la historia de usuario sería para identificarlas mejor entre los desarrolladores y el cliente)
Modificación de Historia de Usuario Número: (Si sufrió alguna modificación anterior)	
Usuario: (Se citan los desarrolladores responsables de la implementación de la HU)	**Iteración Asignada:** (Número de la iteración)
Prioridad en negocio: (Alta / Media / Baja)	**Puntos estimados**: (El tiempo estimado en semanas que se demorará el desarrollo de la HU)
Riesgo en Desarrollo: (Alta / Media / Baja)	**Puntos Reales:** (El tiempo que se demoró en realidad el desarrollo de la HU)
Descripción: (Breve descripción de la HU)	
Observaciones: (Señalamiento o advertencia del sistema)	
Prototipo de interfaz: (Prototipo de interfaz si aplica)	

A continuación se expone una muestra de las HU definidas por el equipo de desarrollo en conjunto con el cliente:

Tabla 2: Historia de Usuario #1 Organizar las grabaciones.

Historia de Usuario	
Número: 1	**Nombre de Historia de Usuario:** Organizar las grabaciones.
Modificación de Historia de Usuario Número: Ninguna.	
Usuario: Irma Rodríguez Ruiz Boris Pita Barrientos	**Iteración Asignada:** 1
Prioridad en negocio: Media.	**Puntos estimados**: 2
Riesgo en Desarrollo: Medio.	**Puntos Reales:** 2
Descripción: El sistema se encarga de acceder a las grabaciones y organizarlas de acuerdo a una nueva estructura de directorios basada en fechas con el formato año/mes/día/nombre de la grabación, así como de actualizar en la base de datos las tablas *call*, *call_entry* y *record*.	
Observaciones:	
Prototipo de interfaz: No Aplica.	

Tabla 3: Historia de Usuario #2 Filtrar y mostrar los datos de las grabaciones.

Historia de Usuario	
Número: 2	**Nombre de Historia de Usuario:** Filtrar y mostrar los datos de las grabaciones.
Modificación de Historia de Usuario Número: Ninguna.	

Usuario: Irma Rodríguez Ruiz Boris Pita Barrientos	**Iteración Asignada:** 2
Prioridad en negocio: Alta.	**Puntos estimados**: 2
Riesgo en Desarrollo: Alto.	**Puntos Reales:** 2

Descripción: El supervisor del sistema selecciona los parámetros por los cuales quiere filtrar las grabaciones y el sistema se encarga de obtener y mostrar los datos correspondientes a las grabaciones tales como:

- número de agente
- nombre del agente
- hora de inicio
- hora de fin
- fecha inicio
- fecha fin
- duración
- tiempo de espera
- cola
- tipo de llamada
- teléfono
- transferencia

- estado

Observaciones: El supervisor del sistema debe estar previamente autenticado.

Prototipo de interfaz:

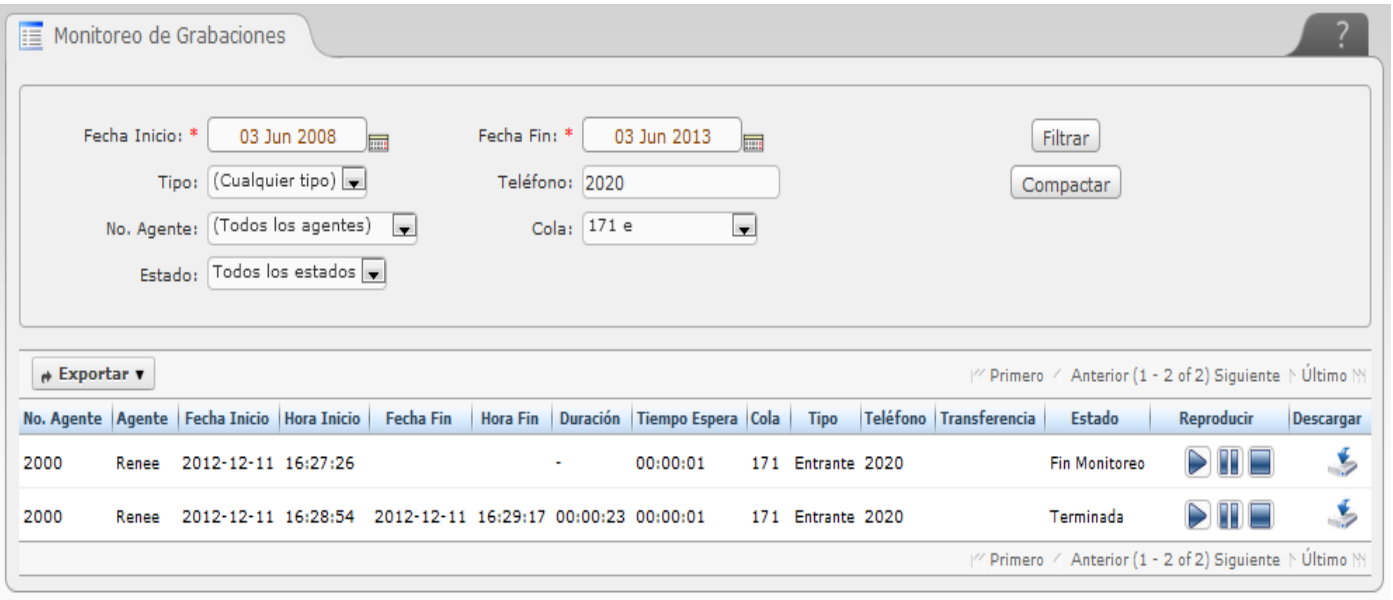


Figura 4: Interfaz de filtrar y mostrar los datos de una grabación.

Tabla 4: Historia de Usuario #3 Cargar dinámicamente los valores de los filtros.

Historia de Usuario	
Número: 3	**Nombre de Historia de Usuario:** Cargar dinámicamente los valores de los filtros.
Modificación de Historia de Usuario Número: Ninguna.	
Usuario: Irma Rodríguez Ruiz Boris Pita Barrientos	**Iteración Asignada:** 1

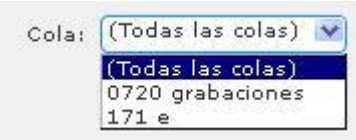
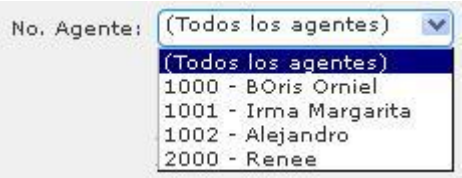
Prioridad en negocio: Baja.	**Puntos estimados**: 1
Riesgo en Desarrollo: Bajo.	**Puntos Reales:** 1
Descripción: El sistema se encarga de obtener los agentes y las colas activas de la base de datos del *Call Center* y agrega los nombres de los mismos como los valores a seleccionar por parte del usuario dentro de los filtros.	
Observaciones: El supervisor del sistema debe estar previamente autenticado.	
Prototipo de interfaz:  **Figura 5:** Filtrar por Cola.  **Figura 6:** Filtrar por Agente.	

Tabla 5: Historia de Usuario #7 Crear archivo compactado.

Historia de Usuario	
Número: 7	**Nombre de Historia de Usuario:** Crear archivo compactado.
Modificación de Historia de Usuario Número: Ninguna.	
Usuario: Irma Rodríguez Ruiz Boris Pita Barrientos	**Iteración Asignada:** 3
Prioridad en negocio: Alta.	**Puntos estimados**: 2

Riesgo en Desarrollo: Alto.	**Puntos Reales:** 2
Descripción: Luego de obtener la(s) grabación(es) deseadas el supervisor del sistema selecciona la opción de exportar y el sistema se encarga de guardar los datos y las grabaciones seleccionadas en un archivo compactado.	
Observaciones: El supervisor del sistema debe estar previamente autenticado.	
Prototipo de interfaz: 	

Figura 7: Compactar grabaciones.

2.6 Fase de Planificación

En la fase de planificación de la metodología XP el cliente y los desarrolladores acuerdan el orden en que deberán implementarse las HU, se realiza la estimación del esfuerzo que costará la implementación de cada HU y asociadas a estas: las entregas. En esta metodología las métricas son libres, por lo que puede utilizarse cualquier criterio para medir el desempeño del proyecto en cuestión, aunque las más usada es la medida de puntos.

Un punto se considera como una semana ideal de trabajo, donde los miembros de los equipos de desarrollo trabajan el tiempo planeado sin ningún tipo de interrupción. Esta estimación incluye todo el esfuerzo asociado a la implementación de la HU. En esta fase se realizan una o varias reuniones grupales de planificación y el resultado de esta fase es un Plan de Entregas.

2.6.1 Estimación del esfuerzo por historias de usuarios

Para desarrollar correctamente la aplicación propuesta se realiza la estimación de esfuerzo para cada HU identificada, llegando a los resultados que se muestran a continuación:

Tabla 6: Estimación del esfuerzo por HU.

Historia de Usuario	Puntos de estimación
Organizar las grabaciones.	2
Mostrar en formato tabla los datos de las grabaciones.	2
Cargar dinámicamente los valores de los filtros.	1
Reproducir grabación.	1
Descargar grabación.	1
Exportar datos de grabaciones.	1
Crear archivo compactado.	2
Importar grabaciones.	3

2.6.2 Plan de iteraciones

Después de identificar, describir las HU y estimar el esfuerzo dedicado a la realización de cada una de ellas, se procede a la planificación de la fase de implementación. Para ello se seleccionan las HU a desarrollar durante cada iteración. Para lograr un mejor desempeño del equipo de desarrollo se establece una división de la implementación en 3 iteraciones.

2.6.3 Plan de duración de las iteraciones

El plan de duración de las iteraciones se encarga de mostrar las HU en el orden en que se implementarán en cada iteración, así como la duración estimada de las mismas. Las iteraciones duran un total de 13 semanas y se dividen en 3. El tiempo de duración de las HU en cada iteración puede observarse en la siguiente tabla:

Tabla 7: Plan de duración de iteraciones.

Iteración	Orden de la HU a implementar.	Duración total

1	HU 1, HU 2	4 semanas
2	HU 7, HU 8	5 semanas
3	HU 3, HU 4, HU 5, HU 6	4 semanas

2.6.4 Plan de entregas

Se presenta el Plan de Entregas estimado para la fase de implementación, detallando la fecha de fin de cada iteración, los productos obtenidos, así como el módulo sobre el cual se está implementando.

SGAVGLL: Sistema de Gestión de Amplios Volúmenes de Grabaciones de Llamadas.

Tabla 8: Plan de entregas.

Módulo	Final de la Iteración 1 (15 de marzo del 2013)	Final de la Iteración 2 (19 de abril del 2013)	Final de la Iteración 3 (17 de mayo del 2013)
SGAVGLL	**SGAVGLL** v0.1	**SGAVGLL** v0.2	**SGAVGLL** v1.0

2.7 Conclusiones del capítulo

Los fundamentos para diseñar el sistema informático propuesto justifican la necesidad de agilizar el proceso de búsqueda de las grabaciones. Por ello se decidió crear una nueva estructura de ficheros basada en fecha, así como minimizar el procesamiento de las peticiones realizadas por el usuario a través del empleo de la información almacenada en la BD. Otro de los elementos a tener en cuenta es la existencia de una aplicación de escritorio que posibilita al supervisor emplear otras vías para el trabajo con las grabaciones, en las que no está sujeto a la disponibilidad de la central telefónica. Además permite verificar las funcionalidades asociadas a cada uno de los subsistemas con la creación de un plan de entregas que facilita al cliente la notificación de cambios o no conformidades que influyen eventualmente en el correcto acabado del sistema.

CAPÍTULO 3: ANÁLISIS Y DISEÑO DEL SISTEMA

En este capítulo se generan los artefactos correspondientes a la fase de diseño de la metodología de desarrollo XP. En esta se plantea que la implementación de un producto debe realizarse de forma iterativa, obteniendo después del desarrollo de cada iteración, un producto funcional que debe ser mostrado al cliente y previamente probado para incrementar la visión de los desarrolladores y clientes de posibles cambios. Además se identifican y organizan las clases relevantes para las funcionalidades del sistema, así como los patrones arquitectónicos utilizados y los patrones GRASP[30] para la realización del módulo. Se detalla además el diseño de la base de datos y se definen las tarjetas CRC en cada una de las iteraciones como herramienta de reflexión en el diseño de software orientado a objetos.

3.1 Patrones Arquitectónicos

Los patrones arquitectónicos o patrones de arquitectura, son patrones de diseño de software que ofrecen soluciones a problemas arquitectónicos en la ingeniería de software. Un patrón arquitectónico expresa un esquema de organización estructural esencial para un sistema de software que consta de: subsistemas, las responsabilidades asociadas a estos e interrelaciones. Ejemplos de estos lo constituyen los siguientes: Programación por capas, *Pipeline*, Invocación implícita, Arquitectura en pizarra, Arquitectura dirigida por eventos, Presentación-abstracción-control, *Peer-to-peer*, SOA[31], Objetos desnudos, Cliente/Servidor y MVC[32] (31). Para el desarrollo del sistema de gestión se utilizaron los patrones de arquitectura MVC y Programación por Capas.

3.1.1 Modelo – Vista – Controlador

Elastix está basado en el patrón de diseño web conocido como arquitectura MVC, el cual es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos. El patrón MVC se ve frecuentemente en aplicaciones web, en las que la vista está conformada por la página HTML y el código que provee de datos dinámicos a la página; el modelo es el sistema de gestión de base de datos y la lógica de negocio, y el controlador es el

[30] **GRASP** – (General Responsibility Assignment Software Patterns): Patrones de Asignación de Responsabilidades.
[31] **SOA** – (Service Oriented Architecture): Arquitectura orientada a servicios.
[32] **MVC** – (Model View Controller): Modelo Vista Controlador.

responsable de recibir los eventos de entrada desde la vista, teniendo como finalidad mejorar la reusabilidad por medio del desacople entre la vista y el modelo (32).

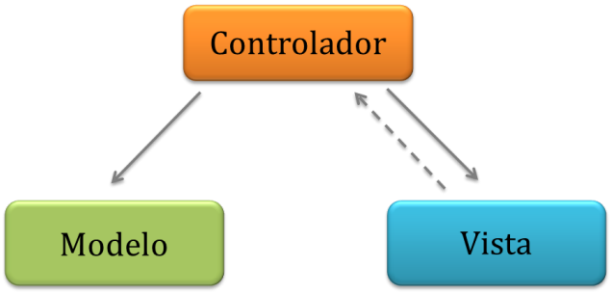


Figura 8: Patrón arquitectónico MVC

Es importante destacar que el *framework* NEO establece un estándar en su estructura que cumple con el patrón de arquitectura MVC. La carpeta *themes* define la capa vista y dentro de ella están las definiciones de las plantillas, el código JavaScript y CSS que proveen de datos dinámicos a la página. La capa de control se define en el archivo *index.php* ya que constituye el punto de entrada común para todas las pantallas de la interfaz de Elastix. Esta es quien decide qué pantalla mostrar en dependencia de los parámetros recibidos. En la carpeta *libs* donde se encuentran las librerías utilizadas es donde se define la capa del modelo (33).

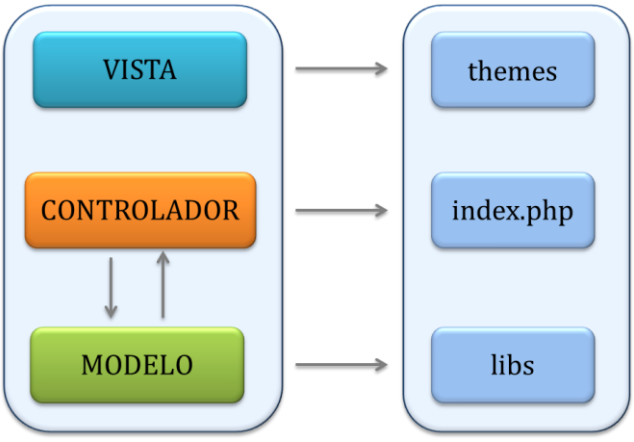


Figura 9: *Framework* NEO basado en arquitectura MVC.

A continuación se muestran las responsabilidades de las capas basadas en la arquitectura MVC empleada en el *framework* Neo:

El modelo es el responsable de:

- Acceder a la capa de almacenamiento de datos. Lo ideal es que el modelo sea independiente del sistema de almacenamiento.
- Definir las reglas de negocio.

El controlador es el responsable de:

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

Las vistas son responsables de:

- Recibir datos del controlador y mostrarlos al usuario.
- Tener un registro de su controlador asociado.
- Dar el servicio de “Actualización”, para que sea invocado por el controlador.

Ventajas del uso del patrón de arquitectura MVC

Dentro de las ventajas que este patrón posee se encuentran:

- La posibilidad de tener diferentes vistas para un mismo modelo.
- La construcción de nuevas vistas sin necesidad de modificar el modelo subyacente.
- Proporciona un mecanismo de configuración a componentes complejos muchos más tratables que el puramente basado en eventos, ya que el modelo puede verse como una representación estructurada del estado de la interacción.

3.1.2 Programación por Capas

En el diseño de sistemas informáticos se suelen emplear las arquitecturas multinivel o Programación por capas. Este patrón permite organizar el modelo de diseño en capas, de forma tal que los componentes de una capa sólo pueden hacer referencia a componentes en capas inmediatamente inferiores. Su empleo ayuda a diseñar arquitecturas escalables (que pueden ampliarse con facilidad en caso de que las necesidades aumenten) y simplifica la comprensión y organización del desarrollo de sistemas complejos; reduciendo las dependencias de forma que las capas más bajas no son conscientes de ningún detalle o interfaz de las superiores. Además permite identificar qué puede reutilizarse dentro de las mismas.

El término "capa" hace referencia a la forma en la que una solución es segmentada desde el punto de vista lógico. Una aplicación se divide en tres capas lógicas distintas, cada una de ellas con un grupo de interfaces perfectamente definidas, tal y como se explica a continuación:

Capa de presentación: Es la capa que presenta el sistema al usuario, le comunica la información y captura los datos del mismo en un mínimo de proceso. También es conocida como interfaz gráfica y debe tener la característica de ser entendible y fácil de usar para el usuario. Esta capa se comunica únicamente con la capa de negocio.

Capa de negocio: Es donde residen las funciones que se ejecutan, se procesa la información y se envían las respuestas tras el proceso. Se denomina capa de negocio o capa de lógica del negocio porque es aquí donde se establecen todas las reglas que deben cumplirse. Esta capa se encarga de recibir las solicitudes y presentar los resultados, y se comunica con la capa de acceso a datos para solicitar al gestor de base de datos almacenar o recuperar datos de él.

Capa de acceso a datos: Esta capa es la encargada de almacenar los datos del sistema y de los usuarios. Su función es almacenar y devolver datos a la capa de negocio, aunque para esto también es necesario en algunos casos, que tengan procedimientos almacenados y funciones dentro de la capa. Está formada por uno o varios sistemas gestores de bases de datos, localizados en un mismo servidor o en varios (34). Tal y como se muestra a continuación:

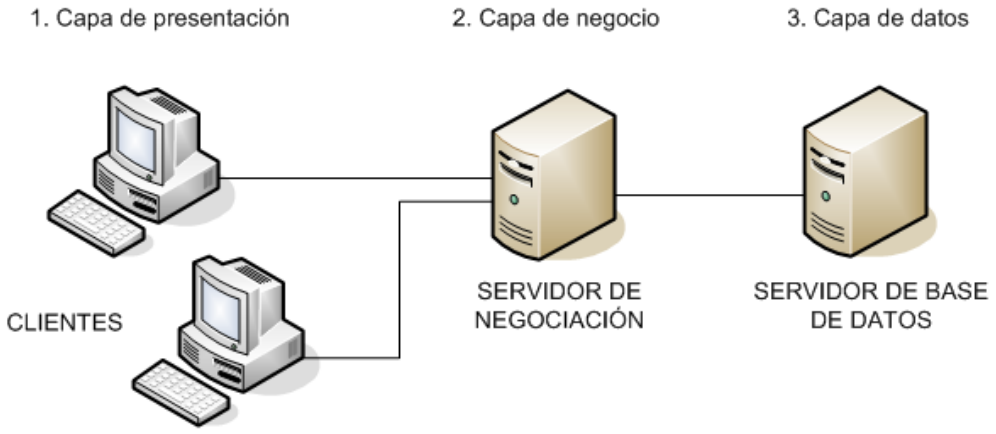


Figura 10: Representación de la programación por capas (Tres capas).

La aplicación de escritorio se implementó siguiendo esta estructura, sin embargo dentro de la tarea programada no existe una capa de presentación o una interfaz de usuario producto a que este no

interactúa con la aplicación, sino que es ejecutada periódicamente por el sistema. Para una mejor comprensión se muestra a continuación la distribución por capas y carpetas en la tarea programada:

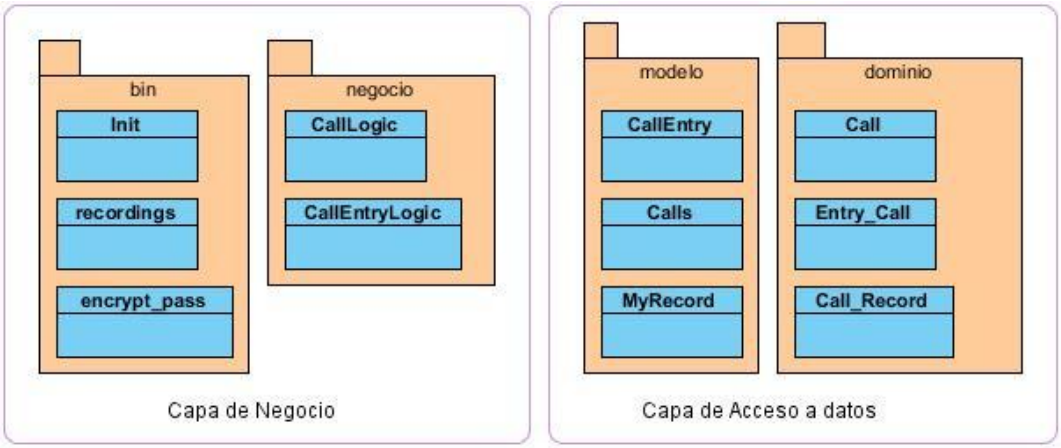


Figura 11: Distribución por capas y carpetas.

La carpeta *bin* contiene las clases encargadas de ejecutar las funciones principales de la solución. Estas por su parte se encuentran implementadas en la carpeta de negocio (para las llamadas entrantes y salientes). La carpeta modelo contiene las clases encargadas de elaborar las consultas necesarias para obtener la información de la BD, mientras que la carpeta dominio contiene las clases encargadas de mapear las tablas de la BD como entidades; haciendo uso del *Doctrine.*

Ventajas del uso de la Programación por Capas

Dentro de las ventajas que este patrón posee se encuentran:

- Aislamiento: ya que se pueden realizar actualizaciones en el interior de las capas sin que esto afecte al resto del sistema.
- Mejor rendimiento: pues en caso que se distribuyan las capas en distintos niveles físicos se mejora la tolerancia a fallos, la escalabilidad y el rendimiento.
- Testeabilidad: producto a que cada capa tiene una interfaz bien definida sobre la cual realizar las pruebas.
- Independencia: ya que elimina la necesidad de considerar el hardware y el despliegue, así como las interfaces externas.

- Rapidez en la implementación: pues permite distribuir el trabajo de creación de una aplicación por niveles y cada grupo de trabajo está totalmente abstraído del resto de niveles (34).

3.2 Patrones de Diseño

Los patrones de diseño son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces. Un patrón es una solución a un problema de diseño no trivial. Para que la solución propuesta sea considerada un patrón se debe comprobar la efectividad, significa que la solución ha valido para resolver el problema en diseños pasados y la otra es que debe ser reusable, es decir, aplicable a diferentes problemas de diseño en distintas circunstancias. En general los patrones de diseño facilitan el aprendizaje y comunicación entre los diseñadores pues poseen una serie de características que definen las estructuras de diseño (o sus relaciones) para el desarrollo de software, enunciadas a continuación:

- Una solución estándar para un problema común de programación.
- Una técnica para flexibilizar el código haciéndolo satisfacer ciertos criterios.
- Un proyecto o estructura de implementación que logra una finalidad determinada.
- Un lenguaje de programación de alto nivel.
- Una manera práctica de describir los aspectos de la organización de un programa.
- Conexiones entre componentes de programas.
- La forma de un diagrama de objeto o de un modelo de objeto (35).

3.2.1 Patrones para Asignar Responsabilidades

Los patrones GRASP describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones (36). Dentro de los patrones GRASP utilizados en el desarrollo del sistema se encuentran los siguientes:

3.2.1.1 Experto

En información es el principio básico de asignación de responsabilidades que suele utilizarse en el diseño orientado a objetos; con él que se pretende designar una idea clara de la mejor forma posible; el patrón experto indica, que la responsabilidad de la creación de un objeto o la implementación de un método, debe recaer sobre la clase que conoce toda la información necesaria para crearlo, es decir , una clase contiene toda la información necesaria para realizarla labor que tiene encomendada (37).

Problema: ¿Cómo lograr que cada clase cumpla con la responsabilidad que le corresponde?
Solución: Asignar una responsabilidad a la clase que tiene la información necesaria para cumplirla.
La clase **CallEntryLogic** posee la información requerida para cumplir las responsabilidades que le corresponden.

3.2.1.2 Creador

Este patrón ayuda a identificar quién debe ser el responsable de la creación (o instanciación) de nuevos objetos o clases (37).
Problema: ¿Cómo lograr relacionar una clase con la clase responsable de realizar la conexión a la base de datos?
Solución: Asignar a la clase B la responsabilidad de crear una instancia de clase A, si se cumple una de las siguientes condiciones:

- B contiene A.
- B registra A.
- B agrega A.
- B utiliza A muy de cerca.
- B tiene los datos de inicialización de A.

La clase **call_dao** es responsable de crear una nueva instancia de la clase que realiza la conexión a la base de datos (conexion)**.**

3.2.1.3 Controlador

Es un patrón que sirve como intermediario entre la interfaz y el algoritmo que la implementa, de tal forma que es la que recibe los datos del usuario y la que los envía a las distintas clases según el método llamado. Sugiere que la lógica de negocios debe estar separada de la capa de presentación, esto posibilita la reutilización de código y a la vez tener un mayor control (37).
Problema: ¿Cómo lograr atender un evento del sistema?
Solución: Un Controlador es un objeto de interfaz no destinada al usuario que se encarga de manejar un evento del sistema. Define además el método de su operación.
La clase **index** es responsable de atender varios eventos del sistema, siendo esta la clase que controla todos los datos relacionados con las llamadas y se los muestra al usuario en la interfaz web.

3.2.1.4 Alta Cohesión

El patrón Alta Cohesión es un principio que se debe tener presente en todas las decisiones de diseño, es la meta principal que ha de buscarse en todo momento. La cohesión es una medida de cuan relacionadas y enfocadas están las responsabilidades de una clase, además de que una alta cohesión garantiza que clases con responsabilidades estrechamente relacionadas no realicen un trabajo enorme y que cada elemento del diseño debe realizar una labor única dentro del sistema (37).

Problema: ¿Cómo lograr que las clases trabajen en su misma área de aplicación?

Solución: Este patrón es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme.

A las clases **Calls** y **CallEntry** se le asignarán responsabilidades con el objetivo de que trabajen en la misma área de aplicación y que no tengan mucha complejidad.

3.2.1.5 Bajo Acoplamiento

El bajo acoplamiento fomenta el aumento de la reutilización y la eliminación de las redundancias, creando clases más independientes y con mayor resistencia al impacto de los cambios, que aumentan la productividad y la posibilidad de reutilización. Este patrón se basa en la idea de tener las clases lo menos entrelazadas que se pueda. De tal forma que en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto de clases, potenciando la reutilización, y disminuyendo la dependencia entre las clases (37).

Problema: ¿Cómo lograr que en caso de producirse una modificación en alguna de las clases, se tenga la mínima repercusión posible en el resto de ellas?

Solución: Este patrón es una medida de la fuerza con que una clase está conectada a otras clases. Acoplamiento bajo significa que una clase no depende de muchas clases.

Las clases se les asignan responsabilidades de forma tal que solo se comunican con la clase **paloSantoDB**. Ejemplo: La clase **paloSantoLlamada**.

3.3 Tarjetas Clase – Responsabilidad – Colaborador

Para el diseño de las aplicaciones, la metodología XP requiere el uso de las tarjetas CRC con el objetivo de desarrollar una representación organizada de las clases. Constituyen un modelo simple de organizar las clases más relevantes para las funcionalidades del sistema.

Un modelo CRC es en realidad una colección de tarjetas índices estándar que representan clases. Las tarjetas se dividen en tres secciones. Por lo general el nombre de la clase se coloca en el borde superior en forma de título, en la parte derecha los colaboradores (las clases que se implican en cada funcionalidad) y las responsabilidades (funcionalidades) de la clase en el extremo izquierdo (31), tal y como muestra la siguiente ilustración:

Tabla 9: Estructura de una Tarjeta CRC.

Nombre de la Clase	
Descripción	
Responsabilidades	Colaboradores

Nombre de la Clase: es cualquier evento, individuo, objeto, concepto, pantalla o reporte.
Descripción: Breve descripción del funcionamiento de la clase.
Responsabilidades: las responsabilidades de una clase, son las obligaciones que tiene un objeto con respecto a su comportamiento. Se asignan a los objetos en el diseño y son acciones que realizan atributos y métodos.
Colaboradores: los colaboradores de una clase son aquellas clases con las que trabaja en conjunto para llevar a cabo sus funcionalidades.
A continuación se definen las tarjetas CRC correspondientes a las clases más relevantes del sistema:

Tabla 10: Tarjeta CRC #1 Clase paloSantoLlamada.

Clase: paloSantoLlamada	
Descripción: clase encargada de obtener los datos a partir de la construcción de las consultas SQL.	
Responsabilidad	**Colaborador**
Construir consulta SQL según los parámetros seleccionados.	paloSantoDB
Obtener los datos de las grabaciones según los parámetros seleccionados.	
Obtener dirección de una o más grabaciones.	
Obtener datos de los agentes existentes en el sistema.	

Tabla 11: Tarjeta CRC #2 Clase index.

Clase: index	
Descripción: clase que procesa los filtros, muestra la información referente a las grabaciones de las llamadas del *Call Center* del Elastix y permite al usuario trabajar con las funcionalidades de descargar, reproducir, exportar y crear compactado.	
Responsabilidad	**Colaborador**
Cargar dinámicamente los valores de los filtros.	paloSantoDB paloSantoLlamada
Filtrar y mostrar los datos de las grabaciones.	
Reproducir grabación.	
Descargar grabación.	
Exportar datos de grabaciones.	
Crear archivo compactado.	

Tabla 12: Tarjeta CRC #4 Clase CallLogic.

Clase: CallLogic	
Descripción: clase encargada de crear la nueva estructura de directorios, organizar las grabaciones según la misma y actualizar en la base de datos para las llamadas salientes.	
Responsabilidad	**Colaborador**
Crear estructura de directorios.	Calls MyRecord
Mover grabaciones salientes a la nueva estructura de directorios.	
Actualizar las tablas *calls* y *record* en la base de datos.	

Tabla 13: Tarjeta CRC #5 Clase CallEntryLogic.

Clase: CallEntryLogic

Descripción: clase encargada de crear la nueva estructura de directorios, organizar las grabaciones según la misma y actualizar en la base de datos para las llamadas entrantes.	
Responsabilidad	**Colaborador**
Crear estructura de directorios.	CallEntry MyRecord
Mover grabaciones entrantes a la nueva estructura de directorios.	
Actualizar las tablas *call_entry* y *record* en la base de datos.	

Tabla 14: Tarjeta CRC #6 Clase CallEntry.

Clase: CallEntry	
Descripción: clase que obtiene los datos de las grabaciones entrantes que han sido guardadas por el Asterisk, pero que no han sido movidas a la nueva estructura de directorios.	
Responsabilidad	**Colaborador**
Obtener datos de grabaciones entrantes sin un *id_record*.	Entry_Call

Tabla 15: Tarjeta CRC #7 Clase Calls

Clase: Calls	
Descripción: clase que obtiene los datos de las grabaciones salientes que han sido guardadas por el Asterisk, pero que no han sido movidas a la nueva estructura de directorios.	
Responsabilidad	**Colaborador**
Obtener datos de grabaciones salientes sin un *id_record*.	Call

Tabla 16: Tarjeta CRC #8 Clase MyRecord.

Clase: MyRecord
Descripción: clase encargada de crear las relaciones de uno a mucho de la tabla *record* con *calls* y

call_entry respectivamente.	
Responsabilidad	**Colaborador**
Crear las relaciones de uno a mucho de la tabla *record* con *calls* y c*all_entry*	Call_Record

Tabla 17: Tarjeta CRC #10 Clase call_dao.

Clase: call_dao	
Descripción: clase encargada de realizar las consultas a la base de datos para obtener los datos de las llamadas.	
Responsabilidad	**Colaborador**
Obtener los datos de las llamadas de la base de datos.	conexion

Tabla 18: Tarjeta CRC # 13 Clase readzip.

Clase: readzip	
Descripción: clase encargada de leer el archivo **.zip**, reproducir el archivo de audio correspondiente a una grabación, filtrar y mostrar los datos de las grabaciones.	
Responsabilidad	**Colaborador**
Cargar archivo compactado.	call_dao
Filtrar y mostrar datos de grabaciones.	call
Reproducir grabación.	mainwindow
Guardar información en la base de datos.	

3.4 Modelo Físico de la Base de Datos

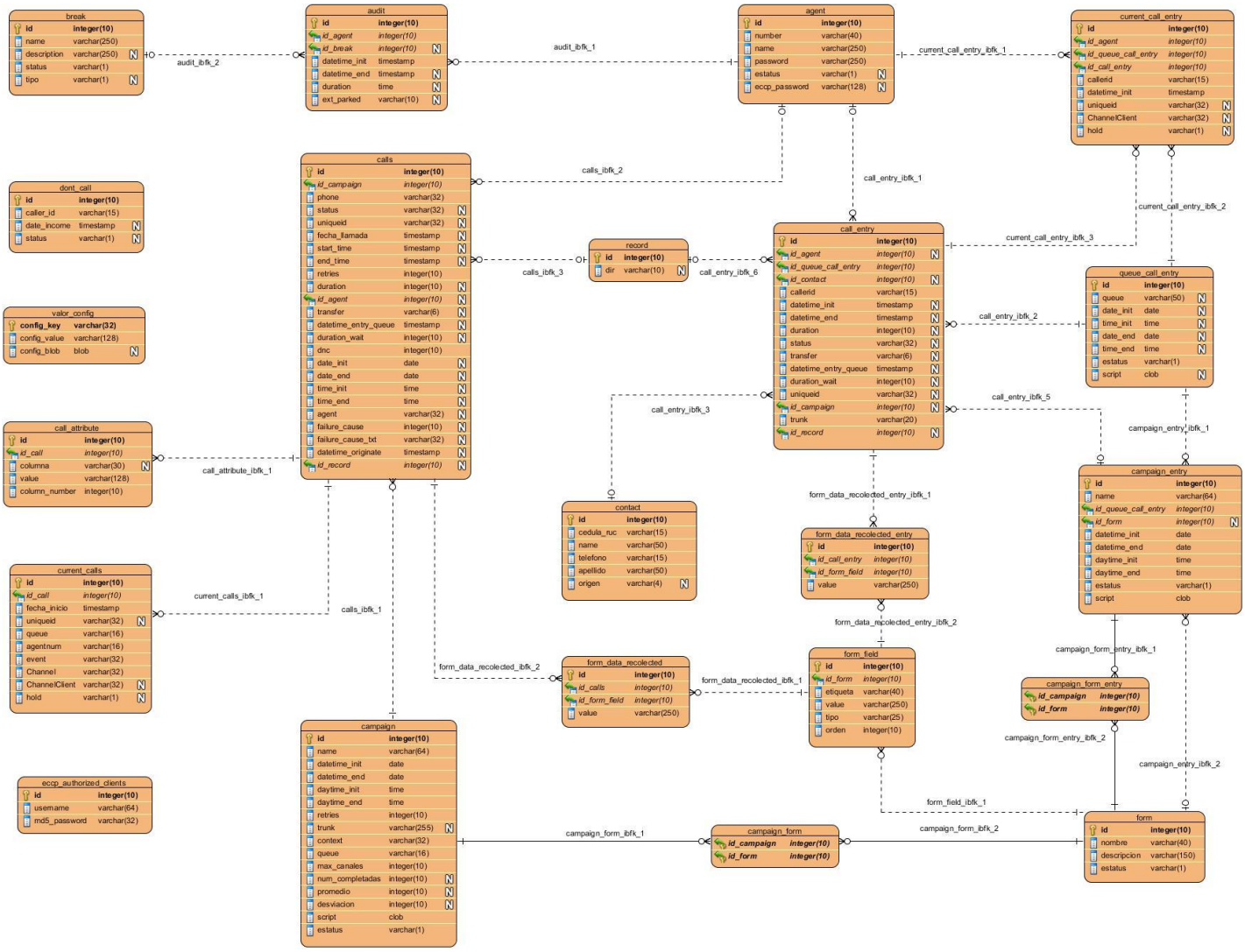


Figura 12: Modelo Físico de la Base de Datos del *Call Center*.

3.5 Especificaciones de la Base de Datos empleada

Inicialmente la base de datos del *Call Center* cuenta con 21 tablas, para la implementación del sistema se le agrega la tabla *record* que tendrá asociado un *id* y un atributo *dir* que almacena la dirección donde se encuentran guardadas cada una de las grabaciones correspondiente a cada llamada. Esta tabla posee una relación de uno a mucho con las tablas *calls* y *calls_entry*, por lo que su llave primaria (*id*) pasa como llave foránea para las dos tablas encargadas de guardar la información referente a las llamadas salientes y entrantes respectivamente. Tal y como se muestra en la siguiente imagen:

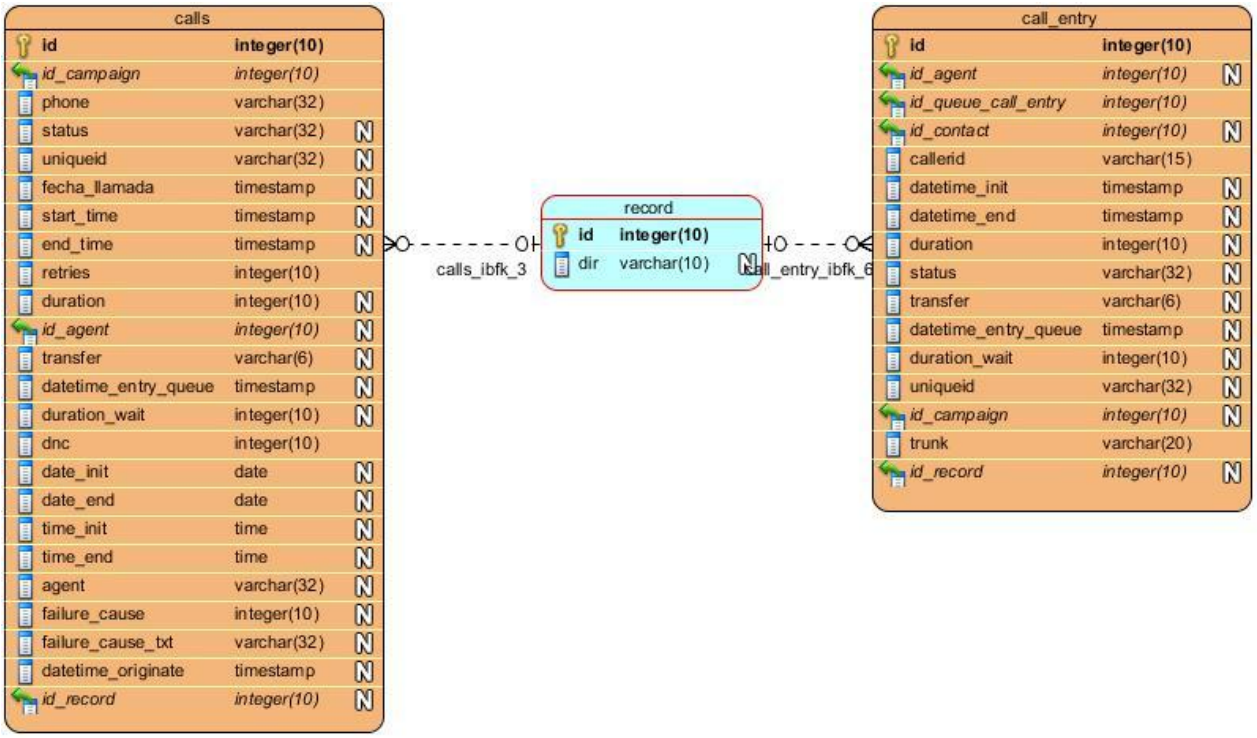


Figura 13: Especificaciones de la Base de Datos empleada.

3.6 Conclusiones del capítulo

En el presente capítulo se realiza un estudio detallado de los patrones de diseño en aras de fomentar la reutilización de código y disminuir el acoplamiento en las soluciones desarrolladas. Se tiene en cuenta además la estandarización del producto en la propuesta de solución planteada a través del empleo del patrón MVC según define el *framework* Neo. Por otra parte se agregó una nueva tabla a la Base de datos para garantizar el correcto funcionamiento del sistema informático para la gestión de grabaciones.

CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBAS

En este capítulo se describe la fase de implementación y prueba; como parte del mismo se detallan las tareas de la ingeniería generadas por cada HU, se realizan además las pruebas de aceptación al software con el propósito de garantizar una adecuada implementación del sistema informático.

4.1 Tareas de Ingeniería

Las tareas de la ingeniería son escritas por el equipo de desarrollo a partir de las HU elaboradas por el cliente. Cada tarea describe a cada HU, dando un detalle más profundo de las mismas para realizar la implementación, estimando un tiempo más cercano a la realidad para realizar cada una de ellas. Las tareas de la ingeniería serán representadas mediante tablas divididas por secciones de la siguiente forma:

Tabla 19: Estructura de una Tarea de Ingeniería.

Tarea de Ingeniería	
Número Tarea: [Los números deben ser consecutivos]	**Número Historia de Usuario:** [Número de la historia de usuario a la que pertenece la tarea]
Nombre Tarea: [Nombre que identifica a la tarea]	
Tipo de Tarea : [Las tareas pueden ser de: Desarrollo, Corrección, Mejora, Otra(Especificar)]	**Puntos Estimados:** [Tiempo estimado en días que se le asignará]
Fecha Inicio:[Fecha en que se inicia la tarea]	**Fecha Fin:** :[Fecha en que se termina la tarea]
Programador responsable: [Los programadores encargados]	
Descripción: [Breve descripción de la tarea]	

A continuación se muestran las tareas de ingeniería correspondientes a algunas de las HU.

Tabla 20: Tarea de Ingeniería #1 Crear nueva estructura de directorios.

Tarea de Ingeniería	
Número Tarea: 1	**Número Historia de Usuario:** 1
Nombre Tarea: Crear nueva estructura de directorios.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 4
Fecha Inicio: 16/2/2013	**Fecha Fin:** 19/2/2013
Programador responsable: Irma Rodríguez Ruiz Boris Pita Barrientos	
Descripción: Acceder al directorio */var/spool/asterisk/monitor/* donde se encuentran almacenadas las grabaciones generadas por la PBX Asterisk y crear la nueva estructura de directorios basada en fecha, de forma dinámica, a partir del nombre de las grabaciones. Esta estructura se encarga de crear la(s) carpeta(s) por año, dentro de estas por meses y en cada uno de ellos por días, siguiendo la estructura Año/Mes/Día.	

Tabla 211: Tarea de Ingeniería #2 Organizar y actualizar datos de grabaciones.

Tarea de Ingeniería	
Número Tarea: 2	**Número Historia de Usuario:** 1
Nombre Tarea: Organizar y actualizar datos de grabaciones.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 4
Fecha Inicio: 20/2/2013	**Fecha Fin:** 23/2/2013
Programador responsable: Irma Rodríguez Ruiz Boris Pita Barrientos	

Descripción: Mover las grabaciones hacia la nueva estructura de directorios y actualizar en la base de datos los datos correspondientes a la nueva dirección de estas en la tabla *record*.

Tabla 22: Tarea de Ingeniería #3 Ejecutar tarea programada.

Tarea de Ingeniería	
Número Tarea: 3	**Número Historia de Usuario:** 1
Nombre Tarea: Ejecutar tarea programada.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 3
Fecha Inicio: 24/2/2013	**Fecha Fin:** 26/2/2013
Programador responsable: Irma Rodríguez Ruiz Boris Pita Barrientos	
Descripción: Ejecutar el proceso de organización de grabaciones y actualización de sus datos dentro del Elastix de forma periódica.	

Tabla 23: Tarea de Ingeniería #11 Obtener dirección de una o más grabaciones.

Tarea de Ingeniería	
Número Tarea: 11	**Número Historia de Usuario:** 4,5
Nombre Tarea: Obtener dirección de una o más grabaciones.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 2
Fecha Inicio: 30/4/2013	**Fecha Fin:** 1/5/2013
Programador responsable: Irma Rodríguez Ruiz Boris Pita Barrientos	

Descripción: Obtener la dirección donde se encuentra almacenada una o más grabaciones a partir de una lista de *id* de grabaciones.

4.2 Pruebas

El instrumento adecuado para determinar el estado de la calidad de un producto de software es el proceso de pruebas. En este se ejecutan pruebas dirigidas a componentes de software o al sistema de software en su totalidad; con el objetivo de medir en qué grado este cumple con los requerimientos (38).

Una de las características principales de XP es su fuerte énfasis en las pruebas, que constituyen uno de los pilares básicos de la metodología. Los desarrolladores dirigen el trabajo a la búsqueda constante de errores y someten sistemáticamente a pruebas las funcionalidades del sistema. Esto posibilita el perfeccionamiento del sistema desarrollado y eventualmente el aumento de la calidad del mismo, reduciendo el número de errores no detectados. Las pruebas a desarrollar se dividen en dos grupos: pruebas unitarias o de integración y pruebas de aceptación.

4.2.1 Pruebas Unitarias

Uno de los métodos utilizados para realizar pruebas de software en la metodología XP son las pruebas unitarias. La base de este método es el hacer pruebas en pequeños fragmentos del código de la aplicación. Estos fragmentos deben ser unidades estructurales del programa encargados de una tarea específica, en programación procedural u orientada a objetos se puede afirmar que estas unidades son los métodos o las funciones que se tienen definidos (38). El objetivo de estas pruebas es el aislamiento de partes del código y la demostración de que no contienen errores. Estas no generan artefactos y no son directamente palpables para el cliente.

Las pruebas unitarias fueron desarrolladas constantemente cada vez que se terminaba de implementar alguna funcionalidad probándola directamente en el entorno real.

4.2.2 Pruebas de Aceptación

Por su parte las pruebas de aceptación son especificadas por el cliente y se enfocan en las características generales y las funcionalidades del sistema, están destinadas a evaluar si al final de una iteración se obtuvo la funcionalidad requerida; así como comprobar que dicha funcionalidad sea la esperada, es decir: la descrita por el cliente en las HU que se han implementado (38).

Para una mayor organización las pruebas de aceptación propuestas a realizarse se encuentran divididas en las siguientes secciones:

- Clases Válidas**:** describe cada uno de los pasos seguidos durante el desarrollo de la prueba, se tiene en cuenta cada una de las entradas válidas que hace el usuario con el objetivo de ver si se obtiene el resultado esperado.
- Clases Inválidas: describe cada uno de los pasos seguidos durante el desarrollo de la prueba, se tiene en cuenta cada una de las posibles entradas inválidas que hace el usuario con el objetivo de ver si se obtiene el resultado esperado y cómo responde el sistema.
- Resultado Esperado: se describe el resultado que se espera ya sea para entradas válidas o entradas inválidas.
- Resultado de la Prueba: se describe el resultado que se obtiene.
- Observaciones: algún señalamiento o advertencia que sea necesario hacerle a la sección que se está probando.

Las pruebas de aceptación se realizan en correspondencia con el diseño de los casos de prueba, para ello se tiene en cuenta el orden preestablecido de las HU y la prioridad asignada, en dependencia de la necesidad de desarrollo de la funcionalidad. A partir de un intercambio con el cliente se deciden las pruebas que serán llevadas a cabo por parte de algunos miembros seleccionados del proyecto, siendo registrados los resultados en cada una de las secciones de las tablas de las pruebas de aceptación.

Tabla 24: Prueba de Aceptación #2 Filtrar y mostrar los datos de las grabaciones.

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la prueba	Observaciones
El usuario accede a la interfaz “Detalles de Llamadas”, selecciona los valores por los cuales desea filtrar las		El sistema muestra los datos de las grabaciones que cumplen con los parámetros seleccionados por el usuario.	Satisfactorio.	El Servidor Elastix tiene que estar funcionando junto con el módulo *Call Center*.

grabaciones y presiona el botón “Filtrar”.				
	El usuario accede a la interfaz “Detalles de Llamadas”, introduce un rango inválido de fecha y presiona el botón “Filtrar”.	El sistema le muestra un mensaje informativo especificando que hay un error de validación en la fecha para que el usuario pueda rectificar la misma.	Satisfactorio.	El Servidor Elastix tiene que estar funcionando junto con el módulo *Call Center.*
	El usuario accede a la interfaz “Detalles de Llamadas”, introduce como número de teléfono una cadena o varios números separados por espacio y presiona el botón “Filtrar”.	El sistema le muestra un mensaje informativo especificando que hay un error de validación en el Teléfono para que el usuario pueda rectificar la misma.	Satisfactorio.	El Servidor Elastix tiene que estar funcionando junto con el módulo *Call Center*.

Tabla 25: Prueba de Aceptación #5 Descargar grabación.

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la prueba	Observaciones
El usuario accede a la interfaz “Detalles		El sistema descarga el archivo de audio de la grabación	Satisfactorio.	El Servidor Elastix tiene que estar funcionando junto

de Llamadas" y presiona la imagen asociada a la opción "Descargar" correspondiente a una grabación.		correspondiente.		con el módulo *Call Center*.

4.2.3 Pruebas de Rendimiento

Las pruebas de rendimiento son una práctica informática que se esfuerza por mejorar el rendimiento, englobándose en el diseño y la arquitectura de un sistema. Sirven para diferentes propósitos tales como: demostrar que un sistema cumple los criterios de rendimiento, comparar dos sistemas y encontrar cuál de ellos funciona mejor o medir que partes del sistema provocan un mal rendimiento. Para su diagnóstico, los ingenieros de software utilizan herramientas encargadas de medir qué partes de un dispositivo o software contribuyen más al mal rendimiento y establecer niveles que mantenga un tiempo de respuesta aceptable (39).

Estas pruebas fueron desarrolladas con el objetivo de observar el comportamiento de la aplicación bajo volumen establecido de grabaciones y determinar la solidez del trabajo desarrollado. La siguiente tabla muestra los resultados de la comparación entre el sistema de Monitoreo existente anteriormente y el nuevo sistema de Gestión:

Tabla 26: Resultados de las pruebas de rendimiento.

No.	Módulo de Monitoreo		Módulo de Gestión de Grabaciones	
	Cant. de grabaciones	Tiempo de respuesta	Cant. de grabaciones	Tiempo de respuesta
1	100	0.76 seg.	100	0.73 seg.
2	1000	1.88 seg.	1000	1.61 seg.
3	5000	4.07 seg.	5000	1.90 seg.

4	40000	20.58 seg.	40000	2.30 seg.
5	60000	No responde	60000	3.80 seg.

Las pruebas fueron realizadas con condiciones semejantes al ambiente real y con iguales prestaciones de hardware. Luego de este análisis se obtuvieron los siguientes resultados:

- Ante un bajo volumen de grabaciones ambos sistemas responden con características similares.
- A medida que se aumenta el volumen de las grabaciones el tiempo de respuesta del módulo existente se hace mayor que el del sistema desarrollado.
- Mientras el tiempo de respuesta del módulo existente aumenta paulatinamente el nuevo sistema de gestión no posee muy variaciones considerables.
- Cuando se generan 60 000 grabaciones el módulo de Monitoreo excede el tiempo de espera y no se muestran en la interfaz web los resultados correspondientes, mientras tanto el nuevo sistema continúa funcionando correctamente.

4.2.4 Resultados de las Pruebas

A continuación se exponen los resultados alcanzados después de realizar las pruebas de aceptación. Se realizaron 3 iteraciones y de las 15 no conformidades encontradas: 10 pertenecen a la primera iteración y 5 a la segunda, pues en la última no fueron detectados nuevos errores. Estas no conformidades fueron solucionadas en sus respectivas iteraciones y ninguna quedó pendiente o no procedió. Tal y como se muestra a continuación:

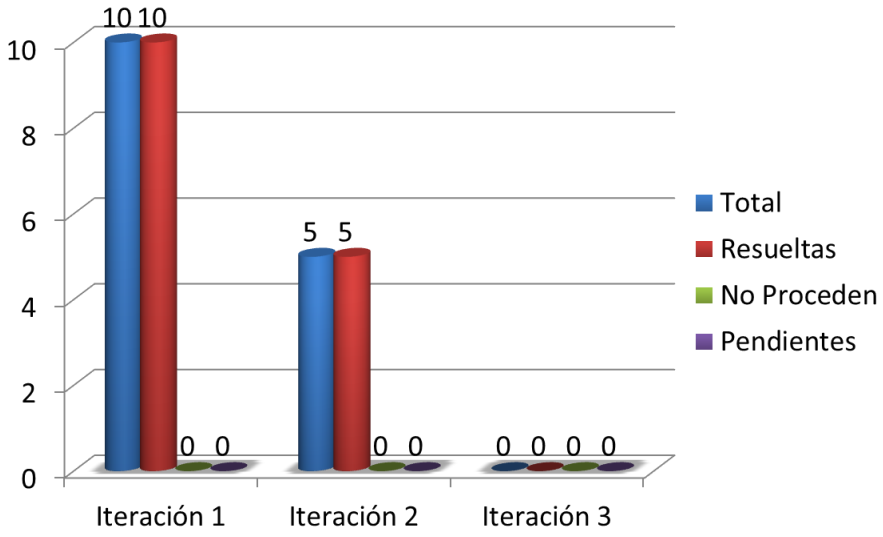


Figura 14: No conformidades.

De estas no conformidades: 2 fueron detectadas en la tarea programada, 8 pertenecen a la interfaz web y 5 a la aplicación de escritorio. Lo cual representa un 13%, 54% y 33% del total respectivamente.

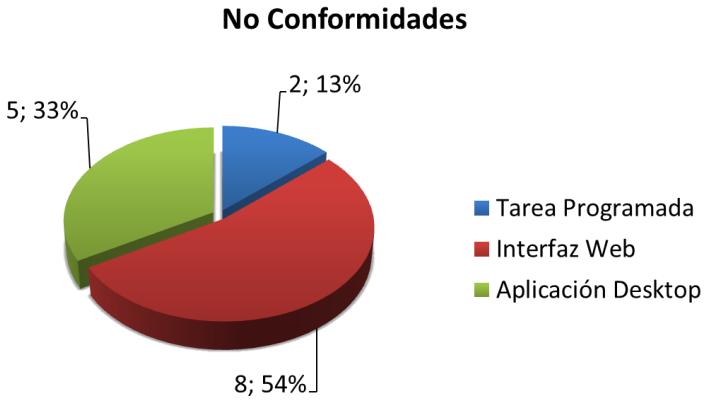


Figura 15: Distribución de no conformidades por cada aplicación.

4.3 Conclusiones del capítulo

En el capítulo se realizó la descripción de las tareas de la ingeniería asociadas a las Historias de Usuario para desglosar la implementación de la solución desarrollada, logrando explicar con mayor detalle todo el proceso de realización de las funcionalidades que componen el producto final y la estimación de las tareas

de ingeniería para establecer un tiempo más cercano a la realidad asociado al cumplimiento de cada una de las mismas.

Se desarrollaron las pruebas unitarias, de aceptación y de rendimiento. A partir de las mismas fueron resueltas deficiencias detectadas en cada una de las aplicaciones que componen el sistema de gestión de grabaciones brindando conformidad y seguridad al cliente.

CONCLUSIONES

- A partir del análisis de las tendencias en cuanto a la implementación de los sistemas de gestión de grabaciones en el campo de la telefonía se evidenció la necesidad de crear a través del software libre la propuesta de solución al problema planteado, la cual garantiza y dispone de las herramientas para gestionar amplios volúmenes de llamadas en el *Call Center* del Elastix.
- El sistema informático propuesto justifica la necesidad de agilizar el proceso de búsqueda de grabaciones y permite verificar las funcionalidades asociadas a cada uno de los subsistemas permitiendo al cliente estar al tanto del desarrollo de la solución.
- A partir del estudio de los patrones de diseño se tuvo en cuenta la reutilización del código y la disminución del acoplamiento en la solución desarrollada.
- La propuesta diseñada cumple con los requisitos para su funcionamiento, siendo demostrado a través de las pruebas unitarias, de aceptación y de rendimiento realizadas al sistema y validadas por el criterio del cliente.

RECOMENDACIONES

A partir de las conclusiones abordadas se proponen las siguientes recomendaciones:

- ✓ Crear un instalador que agilice el proceso de instalación y configuración del módulo integrado al Elastix.
- ✓ Publicar el presente trabajo para que sea utilizado en aquellas organizaciones o empresas en las que exista al menos una planta telefónica, con el fin de mejorar el proceso de gestión de las grabaciones de llamadas y ampliar el uso del software libre en las mismas.
- ✓ Agregar un sistema de salvas a la interfaz web del sistema de gestión de grabaciones.

REFERENCIAS BIBLIOGRÁFICAS

1. **Cabanillas., Marta.** El auténtico valor de la convergencia.Comunicaciones Unificadas. [En línea] [Citado el: 10 de Octubre de 2012.] http://www.networkworld.es/El-autentico-valor-de-la-convergencia_Comunicaciones-unifica/seccion-/articulo-180573.

2. **Landívar, Edgar.** *Comunicaciones Unificadas con Elastix , Volúmen II.* Bolivia : s.n., 2008-2009.

3. Asterisk Colombia. [En línea] [Citado el: 7 de Enero de 2013.] http://www.asteriskcolombia.org/documentacion/general/%C2%BFque-es-asterisk.

4. **Gonçalves, Flavio E.** *Asterisk PBX. Guía de la Configuración.* Janeiro : s.n., 2007. ISBN: 978-85-906904-3-6.

5. Soluciones en Tecnología de la Información y las Comunicaciones. [En línea] [Citado el: 8 de Enero de 2013.] http://www.solutecperu.com/spsac/asterisk-central-telefonica-pbx.

6. **CubaTel s.a. Sociedad Cubana para las Telecomunicaciones.** *Plataforma de Call Center y Servicios.*

7. **Rouse, Margaret.** SearchSoftwareQuality. [En línea] WhatIs.com. [Citado el: 6 de Diciembre de 2012.] http://searchsoftwarequality.techtarget.com/definition/.

8. **Solutions, PaloSanto.** *Elastix 0.9-alpha. Manual de usuario en Español.*

9. Sitio Oficial de Elastix. [En línea] [Citado el: 9 de Enero de 2013.] http://www.elastix.org.

10. Sitio Oficial de la compañia PaloSanto Solutions. [En línea] [Citado el: 9 de Enero de 2013.] http://www.palosanto.com.

11. **Avaya Inc.** Bussines Comunications Solutions from Avaya. [En línea] [Citado el: 9 de Enero de 2013.] http://www.avaya.com/es/resource/assets/factsheet/bp2576se.pdf.

12. Quarea. [En línea] http://www.quarea.com/es/centralita_IP_asterisk_avanvox.

13. Centrales telefonicas PANASONIC. [En línea] http://www.telec.com.ar/shop/otraspaginas.asp?paginanp=62&t=grabador-de-llamadas.htm.

14. **Amaro Calderón, Sarah Dámaris Valverde y Jorge Carlos Rebaza.** *Metodologías Ágiles.* Trujillo, Perú : s.n., 2007.

15. **Pressman, Roger S.** *Software Engineering: A practitioner's Approach.Chapter3: Agile Development.* 2005.

16. **Zapata., Luis Giraldo y Yuliana.** *Herramientas de desarrollo de ingeniería de software para Linux. .* s.l. : Monitoria de Ingesoft, 2005.

17. **MySQL Enterprise.** MySQL: Developer Zone. [En línea] http://dev.mysql.com/doc/refman/5.0/es/features.html.

18. SQLite Home Page. [En línea] [Citado el: 9 de Diciembre de 2012.] http://www.sqlite.org.

19. **Vázquez, José Antonio Gallego.** *Desarrollo Web con PHP y MySQL.* Madrid,España : ANAYA Multimedia, 2003.

20. Cplusplus. [En línea] [Citado el: 10 de Diciembre de 2012.] http://www.cplusplus.com/.

21. **The PHP Group.** Sitio Oficial de PHP. [En línea] www.php.net.

22. **Pérez, Javier Eguíluz.** Introducción a Javascript. [En línea] 7 de Junio de 2008. Disponible en: http://sunshine.prod.uci.cu.

23. Programación y Diseño web. [En línea] España. [Citado el: 7 de Diciembre de 2012.] http://www.masadelante.com.

24. Framework NEO. [En línea] [Citado el: 10 de Enero de 2013.] http://www.neoframework.org.

25. **Rojas, Alan D. Osorio.** *Qt 4 Manual Introductorio.* 16-Enero-2008.

26. **Jonathan H. Wage, Guilherme Blanco, Benjamin Eberlei, Bulat Shakirzyanov.** Doctrine Project. [En línea] [Citado el: 10 de Diciembre de 2012.] http://www.doctrine-project.org/.

27. **Oracle Corporation.** Sitio Oficial de Netbeans. [En línea] http://netbeans.org.

28. **Nokia.** Qt Creator IDE and tools. [En línea] [Citado el: 13 de Noviembre de 2011.] http://qt.nokia.com/products/developer-tools.

29. **Pantoja Záldivar, Yoenis, Sánchez Gutierrez, Alain y Paneque Corchete, Antonio.** Plataforma de servicios de Mapas para la Univerisidad de las Ciencias Informáticas. [En línea] 26 de enero de 2012. [Citado el: 2 de febrero de 2012.] Disponible en: http://vinculando.org/beta/plataforma-de-servicios-de-mapas.html.

30. **José H. Canós, Patricio Letelier y Mª Carmen Penadés.** *Métodologías Ágiles en el Desarrollo de Software.* Valencia : DSIC -Universidad Politécnica de Valencia.

31. **Pressman, Roger S.** *Software Engineering, a practitioner's approach. (7th edición).* s.l. : McGraw-Hill, 10 de Noviembre de 2011. ISBN 9780071267823.

32. **Comusoft.com.** Comunidad de Sftware y Tecnología. [En línea] [Citado el: 15 de Abril de 2013.] http://www.comusoft.com/modelo-vista-controlador-definicion-y-caracteristicas.

33. Extendiendo la interfaz web del Elastix. [En línea] [Citado el: 17 de Febrero de 2013.] http://es.scribd.com/doc/70479844/139/El-framework-Neo.

34. **César de la Torre Llorente, Unai Zorrilla Castro, Miguel Ángel Ramos Barros y Javier Calvarro.** *Guía de Arquitectura N-Capas orientada al Dominio con .NET 4.0.* España : Krasis Consulting, marzo del 2010. ISBN 978-84-936696-3-8.

35. **Larman, Craig.** *UML y Patrones. Introducción al análisis y diseño orientado a objetos.* México : s.n., 1999.

36. Prácticas de Software. *Experiencias sobre la Ingeniería y Management del Software.* [En línea] [Citado el: 26 de Abril de 2013.] http://www.practicasdesoftware.com/.

37. **SlideShare.** Ingeniería en Sistemas de Información. Diseño de sistemas. [En línea] [Citado el: 3 de febrero de 2013.] Disponible en: http://www.slideshare.net/jpbthames/patrones-para-asignar-responsabilidades-grasp.

38. Pruebas de Software. *Gestión de Calidad y Pruebas de Software.* [En línea] [Citado el: 30 de Abril de 2013.] http://pruebasdesoftware.com.

39. **O'Reilly Media.** *The Art of Application Performance Testing.* Enero de 2009. ISBN 978-0-596-52066-3.

BIBLIOGRAFÍAS CONSULTADAS

1. Buenas Tareas. [En línea] [Citado el: 15 de Mayo de 2013.] http://www.buenastareas.com/temas/pruebas-de-aceptaci%C3%B3n/20.

2. Call Center. [En línea] [Citado el: 17 de Enero de 2013.] http://www.estoesmarketing.com/Habilidades/CallCenter.pdf.

3. Gestión de Calidad y Pruebas de Software. [En línea] [Citado el: 9 de Mayo de 2013.] Disponible en: http://pruebasdesoftware.com/pruebadeaceptacion.htm.

4. La Revista Informática.com. [En línea] [Citado el: 21 de Febrero de 2013.] http://www.larevistainformatica.com/CENTRALES-TELEFONICA.html.

5. Proactiva-calidad. [En línea] [Citado el: 13 de Mayo de 2013.] http://www.proactiva-calidad.com/java/patrones/mvc.html.

6. Programación Extrema. [En línea] [Citado el: 27 de Abril de 2013.] http://programacionextrema.tripod.com/fases.htm.

7. Slideshare. [En línea] [Citado el: 28 de Marzo de 2013.] http://www.slideshare.net/alexmerono/sistemas-gestores-de-bases-de-datos.

8. Universidad de Mursia. [En línea] [Citado el: 4 de Mayo de 2013.] http://www.um.es/docencia/barzana/IAGP/IAGP2-Metodologias-de-desarrollo.html.

9. **Sitio Oficial de Visual Paradigm.** Visual Paradigm. [En línea] [Citado el: 10 de Diciembre de 2012.] Disponible en: http://www.visual-paradigm.com.

10. Webtutoriales. [En línea] [Citado el: 24 de Marzo de 2013.] http://www.webtutoriales.com/tutoriales/programacion/modelo-vista-controlador.54.html.

11. **Escribano, Gerardo Fernández.** Introducción a Extreme Programming. [En línea] [Citado el: 9 de Diciembre de 2012.] Disponible en: http://www.dsi.uclm.es/asignaturas/42551/trabajosAnteriores/Presentacion-XP.pdf.

12. **Grau Abalo, Ricardo, Correa Valdés, Cecilia y Rojas Betancourt, Mauricio.** *METODOLOGÍA DE LA INVESTIGACIÓN.* [Documento] Ibagué : s.n.

13. **Gutiérrez, J. J., y otros, y otros.** *PRUEBAS DEL SISTEMA EN PROGRAMACIÓN EXTREMA.* [Documento] s.l. : Universidad de Sevilla.

14. **José H. Canós, Patricio Letelier y Mª Carmen Penadés.** *Métodologías Ágiles en el Desarrollo de Software.* Valencia : Universidad Politécnica de Valencia. : s.n.

15. **Kniberg, Henrik.** *Scrum y XP desde las trincheras.* s.l. : C4Media Inc., 2007.

16. **Landívar, Edgar.** *Comunicaciones Unificadas con Elastix.* [Documento] 1999.

17. **Larman, Craig.** *UML y Patrones. Introducción al análisis y diseño orientado a objetos.* México : s.n., 1999.

18. **Pérez, Javier Eguíluz.** Introducción a XHTML. [En línea] [Citado el: 10 de Diciembre de 2012.] Disponible en: http:// www.librosweb.es/xhtml/index.html.

19. **Prieto, Félix.** *Patrones de Diseño.* [Documento] s.l. : Universidad de Valladolid, 2008.

20. **Solis, Manuel Calero.** Una explicación de la programación extrema (XP). [En línea] [Citado el: 9 de Diciembre de 2012.] Disponible en: http://www.willydev.net/descargas/prev/ExplicaXp.pdf.

21. **Solutions, PaloSanto.** *Elastix 0.9-alpha.Manual de usuario en Español.*

22. **Sommerville, Ian.** *Ingeniería de Software. Séptima Edición.* [Documento] Madrid : Pearson educación, 2005.

23. **Villar, Malay Rodríguez.** Introducción de procedimientos Ágiles en la producción de software en la Facultad 7 de la Universidad de las Ciencias Informáticas. [En línea] [Citado el: 9 de Diciembre de 2012.] Disponible en: http://bibliodoc.uci.cu/TD/TD_0693_07.pdf.

24. Desarrollo Web. [En línea] [Citado el: 27 de Abril de 2013.] http://www.desarrolloweb.com/articulos/que-es-html.html.

25. Extendiendo la interfaz web del Elastix. [En línea] [Citado el: 17 de Febrero de 2013.] http://es.scribd.com/doc/70479844/139/El-framework-Neo.

26. Webestilo.Mysql. [En línea] [Citado el: 4 de Marzo de 2013.] http://www.webestilo.com/mysql/intro.phtml.

27. Wigahluk. [En línea] [Citado el: 18 de Mayo de 2013.] http://wigahluk.wordpress.com/category/software-development/pruebas-unitarias.

28. Asterisk. [En línea] [Citado el: 3 de Mayo de 2013.] http://www.asterisk.org.

29. CAVSI. [En línea] [Citado el: 29 de Marzo de 2013.] http://www.cavsi.com/preguntasrespuestas/que-es-un-sistema-gestor-de-bases-de-datos-o-sgbd.

30. **Lapuente, María Jesús Lamarca.** Tesis Doctoral Hipertexto: El nuevo concepto de documento en la cultura de la imagen. [En línea] [Citado el: 25 de Enero de 2013.] Disponible en http://www.hipertexto.info.

31. Instituto Tecnológico de Veracruz. [En línea] [Citado el: 7 de Enero de 2013.] http://www.prograweb.com.mx/pweb/0203ladoServidor.html.