MAGIC SMTP

Memoria del proyecto
de Ingeniera Técnica en
Informática de Sistemas
realizado por

Pablo Pedrosa Manchón

i dirigido por

Yolanda Benítez Fernández

Escola d'Enginyeria
Sabadell, Junio de 2013

El abajo firmante, Yolanda Benítez Fernández,
Profesor de la Escola d'Enginyeria de la UAB,

CERTIFICA:

Que el trabajo al que corresponde esta memoria ha sido realizado bajo su dirección por **Pablo Pedrosa Manchón**

Y para que conste firma la presente.
Sabadell, Junio de 2013

Firmado: **Yolanda Benítez Fernández**

Universitat Autònoma de Barcelona

HOJA DEL RESUMEN - PROYECTO FINAL DE CARRERA DE L'ESCOLA D'ENGINYERIA

Titulo del proyecto:

Magic SMTP

Autor: PABLO PEDROSA MANCHÓN **Fecha:** Mayo 2013

Tutor: YOLANDA BENÍTEZ FERNÁNDEZ

Titulación: INGENIERÍA TÉCNICA EN INFORMÁTICA DE SISTEMAS

Palabras clave:

- Catalán: SMTP, email, email marketing, API
- Castellano: SMTP, email, email marketing, API
- Inglés: SMTP, email, email marketing, API

Resumen del proyecto:

- Catalán: Desenvolupament de una API per el protocol SMTP. A través d'aquesta API els clients podrán conectar els seus programes clients de correu electrònic. Connectant els seus clients podrán enviar emails y veure estadístiques dels mateixos enviaments a través de l'nterficie web Eolica. Es tracta de donar un valor afegit a clients que fan les seves campanyes o enviaments de email a través dels seus clients de correu electrònic.

- Castellano: Desarrollo de un API por el protocolo SMTP. A través de este API los clientes podrán conectar sus programas cliente de correo electrónico. Conectando los clientes podrán enviar emails y ver estadísticas de los propios envíos a través de la interfaz web Eolica. Se trata de dar un valor añadido a clientes que hacen sus campañas o envíos de email a través de sus clientes de correo electrónico.

- Inglés: Development of an API for the SMTP protocol. Through this API customers can connect their email client programs. Connecting customers can send emails and view statistics of their own shipments through Eolica web interface. It's about giving value to customers who make their campaigns or sending emails through email clients.

AGRADECIMIENTOS

A Yolanda Benítez por sus indicaciones
en la realización del proyecto,

a infoAvisos por proporcionarme todos los recursos

y, a todas aquellas personas que confiaron
en que acabaría con este proyecto

AGRADECIMIENTOS 4
1. INTRODUCCIÓN 10
1.1. Descripción 11
1.2. Objetivos 12
1.3. Motivaciones personales 13
1.4. Estructura de la memoria 14
2. ESTUDIO DE VIABILIDAD 16
2.1. Introducción 16
2.2. Descripción 16
2.2.1. Tipología y palabras clave 16
2.2.2. Objetivos del proyecto 16
2.2.3. Definiciones, acrónimos y abreviaciones 17
2.3.4. Partes interesadas 18
2.3. Estudio de la situación actual del marketing online 19
2.3.1. Lógica del sistema 20
2.3.2. Usuarios y/o personal 21
2.3.3. Diagnóstico del sistema 21
2.3.4. Normativas y legislación 22
2.4. Requisitos del sistema 22
2.4.1. Requisitos funcionales 23
2.4.2. Requisitos no funcionales 23
2.4.3. Restricciones del sistema 23
2.4.4. Prioridades de los requisitos funcionales 23
2.4.5. Prioridades de los requisitos no funcionales 24
2.4.6. Relación objetivos con requisitos 24
2.5. Modelo de negocio 24
2.6. Alternativas y selección de la solución 26
2.6.1. Alternativa 1: MailChimp 26
2.6.2. Alternativa 2: Amazon SES 27

2.6.3. Alternativa 3: Canalmail 28
2.7. Solución propuesta 28
2.8. Conclusiones 29
3. PLAN DE PROYECTO 30
3.1. Introducción 30
3.1.1. Definiciones, acrónimos y abreviaciones 30
3.2. WBS (Work Breakdown Structure) 30
3.2.1. Fases y actividades del proyecto 30
3.2.2. Diagrama WBS 31
3.2.3. Milestones 32
3.3. Recursos del proyecto 32
3.3.1. Descripción de los recursos 32
3.3.2. Calendario de los recursos 33
3.4. Calendario del proyecto 33
3.4.1. Dependencias 33
3.4.2. Cuadro de tareas del proyecto 34
3.4.3. Calendario temporal (MS Project) 35
3.5. Evaluación de riesgos 36
3.5.1. Lista de riesgos 36
3.5.2. Catalogación de riesgos 37
3.5.3. Plan de contingencia 38
3.6. Presupuesto 38
3.6.1. Estimación coste personal 38
3.6.2. Estimación costes de los recursos 39
3.6.3. Estimación costes de las actividades 40
3.6.4. Estimación costes de desarrollo 40
3.6.5. Resumen y análisis de los costes 40
4. MARCO TEORICO 42
4.1. Entornos de desarrollo 42

4.1.1. Eclipse 42
4.2. Herramientas 44
4.2.1. MySQL 44
4.2.2. MySQL Workbench 44
4.2.3. Sequel Pro 45
4.2.4. Java Virtual Machine (JVM) 46
4.2.5. Git 47
4.3. Lenguajes de programación 47
4.3.1. Java 47
4.3.2. HTML 49
4.4. Librerías utilizadas 50
4.4.1. SubEtha SMTP 50
4.4.2. Javamail 51
4.4.3. HTML Parser 52
4.5. Software de apoyo 53
4.5.1. Eolica Services 53
4.6. Comparativa con otras tecnologías 53
4.6.1. Comparativa con otros lenguajes 53
4.6.2. Comparativa con otros servidores SMTP 54
4.6.3. Comparativa con otros parsers HTML 55
5. DISEÑO 56
5.1. Comparativa con otras tecnologías 56
5.2. Actores 57
5.2.1. Usuario Administrador 57
5.2.2. Usuario Empresa 57
5.3. Casos de uso 58
5.3.1. Casos de uso del actor usuario administrador 58
5.3.2. Casos de uso del actor usuario empresa 60
5.4. Arquitectura básica del sistema 61

5.5. Componentes 61
5.5.1. SMTP Entrada (Magic) 61
5.5.2. Eolica Services 62
5.6. Base de datos 64
5.6.1. Tabla tenvio 64
5.6.2. Tabla tempresa 65
5.6.3. Tabla sms_outbox 66
5.6.4. Tabla tcontactos_envios 67
5.7. Desarrollo del sistema 67
5.7.1. Escucha activa puerto 25 TCP 68
5.7.2. Verificación de usuario en el MAIL FROM 68
5.7.3. Parser de email 70
5.7.4 Parser de HTML 72
5.7.5. Modificación links y inserción de píxel para estadísticas 72
5.7.6. Integración Eolica para envío de la campaña y seguimiento de estadísticas 73
6. CODIFICACIÓN Y PRUEBAS 74
6.1. Estilo de codificación 74
7. CONCLUSIONES 76
7.1. Desviaciones 76
7.2 Ampliaciones 77
8. BIBLIOGRAFIA 78
9. GLOSARIO 80

1. INTRODUCCIÓN

Durante cinco años he trabajado para una empresa llamada Infoavisos S.L. la cual se dedica al envío de comunicaciones.

Nos hemos planteado muchas veces, saber cual es nuestro papel dentro del mundo de las comunicaciones. Después de muchos quebraderos de cabeza, creemos que somos “carteros electrónicos”.

Dicho esto, cómo estudiante de Ingeniería Técnica Informática de Sistemas y viendo las necesidades que tiene nuestra empresa para abarcar todo lo relacionado con esa definición “carteros online”, decidí desarrollar lo que llamaré Magic SMTP.

Dentro de nuestro negocio, tenemos una herramienta de marketing online. Esta herramienta, llamada Eolica, permite hacer el envío masivo de emails y te permite evaluar el impacto que ha tenido la misma, tasa de llegada, de no llegada, de apertura, clics en links... De esta herramienta nos ha surgido la necesidad de llegar a clientes, los cuales no tienen un conocimiento elevado de internet, pequeñas empresas las cuales quieren enviar campañas de email, pero no saben cómo. Magic SMTP quiere acercar a estos usuarios de pequeña y mediana empresa, a las campañas de marketing por email.

Las pequeñas y medianas empresas con poco conocimiento de estos servicios, suelen enviar sus emails con los típicos clientes de correo, Outlook, Apple mail ... Estos clientes de correo, suelen ser clientes de escritorio y bastante rudimentarios para el envío de estas campañas. Nuestra experiencia nos dice que por mucho que intentes convencer a las empresas de migrar su cliente de correo a un software para hacer email marketing, estas prefieren utilizar su sistema.

Magic SMTP, es un servidor de email, el cual permitirá a estas empresas que no quieran migrar sus bases de datos de contactos, ni quieran cambiar su forma de hacer los envíos, enviar sus campañas de email, con tasas de llegada, de apertura, de lectura, clics en links...

Existen servicios de este tipo, pero Infoavisos decidió por incluirlo en su cartera de productos ya que tenemos varios posibles clientes interesados en el mismo.

1.1. Descripción

Esta aplicación está enfocada a esos usuarios que no quieren cambiar la forma de actuar en internet con sus clientes. Es decir, para pequeña y mediana empresa que quiere seguir gestionando sus comunicaciones por email de la misma forma solo configurando unos parámetros en su cliente de correo.

La principal característica de este software, Magic SMTP, será la de poder hacer un seguimiento activo de todos los envíos de email de estas empresas, sin tener que hacer grandes cambios en su forma de actuar. Si un cliente tiene una base de datos en Outlook de cien clientes y envía sus comunicaciones vía Outlook, solo cambiando el servidor smtp por el cual envía sus emails, podrá tener estadísticas de los mismos.

El cliente recibirá un pequeño manual de como cambiar sus parámetros SMTP de su cliente de correo y así podrá empezar a usar el servicio.

El servicio, Magic SMTP, validará la cuenta desde donde envía el usuario para poder verificar que es una cuenta que puede usar el servicio. Esta validación la hará mediante el protocolo SMTP, en el MAIL FROM más concretamente.

Una vez validado el usuario, el envío será realizado y el mismo usuario podrá entrar en su panel Eolica para hacer un seguimiento del envío realizado.

El sistema está pensado para enviar un email en texto plano, con una plantilla personalizada, o, un mail en HTML.

El envío se beneficiará del sistema Eolica, ya que tiene un historial de envío y las Direcciones IP que utilizamos están bien consideradas por los principales servidores de correo, Gmail, HOTMAIL, Yahoo...

1.2. Objetivos

Hay varios objetivos en la realización de este proyecto, el principal es adquirir conocimientos sobre el protocolo SMTP y así poder ser “más experto” como “cartero electrónico”. Por otra parte, está la consecución del proyecto final de carrera y así la consecuente titulación como Ingeniero Técnico en Informática de Sistemas.

Tener un buen conocimiento sobre el protocolo SMTP hace que podamos saber que acciones intervienen en el proceso de envío de un email y que servidores actúan hasta llegar a destino.

Si todo va bien, este producto saldrá a la cartera comercial en Septiembre-Octubre de 2013 y así también conseguiremos abarcar estos clientes que nos han pedido la herramienta.

Cómo necesidades que cubre mi proyecto, podemos destacar.

- Facilitar al usuario la comunicación online con su cartera de clientes.
- Proporcionar una herramienta para la fácil integración de las empresas.
- Generar estadísticas de envíos a pequeña y mediana empresa.
- Hacer más fácil la evolución de la pequeña y mediana empresa al mundo de las comunicaciones online.
- Ahorrar procesos de migración de servicios.
- Dar a conocer el mundo del marketing online a aquellas empresas que aún no están sumergidas en el mismo.
- Ser cartero electrónico de mediana y pequeña empresa.

También destacar que todo el tráfico que entre por Magic SMTP hará crecer nuestras Direcciones IP y así estás serán más populares y fiables para los principales servidores SMTP.

Por supuesto, está el objetivo de aumentar la cartera de clientes de Infoavisos.

1.3. Motivaciones personales

Las principales motivaciones personales son, crear un producto nuevo para la cartera de clientes de mi empresa y, por otro lado, la obtención de la titulación de Ingeniero Técnico en Informática de Sistemas.

En el tiempo que llevo trabajando en Infoavisos, hemos creado desde aplicaciones web, hasta aplicaciones para móviles. Este proyecto es otra idea de Infoavisos surgida de la necesidad de varios posibles clientes que no quieren integrarse con sistemas de email marketing y necesitan hacer envíos de campañas de email.

Hace unos años que creamos Eolica, una herramienta para hacer marketing online. Tuvo buena acogida entre nuestros clientes, pero hubo otros que necesitaban mi proyecto, Magic SMTP.

Gracias al mismo he podido aprender como se implementa una conversación SMTP, qué información incluye una cabecera de email, qué tipos de email existen, qué interviene en una o varias conversaciones SMTP durante el envío de un email.

La consecución de estos conocimientos, servirán para aumentar mi currículum y para ser mejor “cartero electrónico”.

La aplicación está hecha en Java, y he aprovechado varias librerías para poder hacer el sistema. Java es una herramienta que se usa mucho en la actualidad, pero que yo nunca había usado. Había utilizado OBJECTIVE-C, HTML, CSS, Javascript, JQuery, PHP y sus Frameworks, Coldfusion, PERL. Actualmente con este proyecto, ya puedo decir que he tocado algo de Java.

Esto es algo que siempre sufriremos o admiraremos, nos toca actualizarnos día a día y aprender el máximo de tecnologías ya que en la informática se crea una nueva herramienta cada minuto.

También he adquirido algún conocimiento en como actúan los principales servidores de correo contra el llamado spam. Cómo hacer crecer la popularidad de tu IP, o IP's de salida de tus envíos.

1.4. Estructura de la memoria

A partir de la introducción del proyecto, aquí dejo detallados los puntos que permitirán tener una visión más técnica de cada uno de los apartados del proyecto Magic SMTP:

- El **estudio de viabilidad** del proyecto, donde se detallará el plan de negocio, se estudiará la situación actual analizando la posible competencia, ofreciendo una comparativa entre la misma y Magic SMTP, y la justificación del proyecto.

- El **plan del proyecto**, donde se analizarán las fases del proyecto, especificando las tareas y los recursos necesarios. Veremos cuales son los riesgos del mismo y el plan de contingencia en base a sus riesgos.

- El **presupuesto**, calculando todos los costes del proyecto. En este presupuesto se especificará una estimación del coste total del proyecto.

- El **marco teórico** incluirá una descripción de los aspectos teóricos del proyecto, por ejemplo toda la tecnología usada en el mismo.

- El **diseño** tendrá los diagramas de casos de uso y de las tablas de la base de datos utilizada. También aparecerá la arquitectura del proyecto y una explicación de la misma.

- Las **conclusiones**, será el apartado donde se explicarán toda la serie de conocimientos adquiridos. También se verán todas las posibles evoluciones del software y cómo ha sido la evolución del software, proyecto, actual.

- La **bibliografía y glosario**, mostrará toda la información recopilada para el desarrollo del proyecto y las diferentes palabras técnicas utilizadas.

2. ESTUDIO DE VIABILIDAD

2.1. Introducción

El estudio de viabilidad pretende evaluar la viabilidad del proyecto, estudiando su rentabilidad económica así como el plan de desarrollo.

Veremos todos los costes del proyecto y la posible salida económica que se le puede dar a nivel comercial. Se analizarán los objetivos y las estrategias a realizar. Se incluirán todas las actividades a desarrollar y su planificación.

2.2. Descripción

Magic SMTP pretende ser una herramienta que permita a los usuarios poder enviar campañas de email de forma práctica y sencilla. Con la herramienta podrán conectar sus actuales clientes de correo (Outlook, Apple mail...) a un sistema de envía el cual extraerá toda la información posible del envío. Esta información se verá en un panel web. Es una herramienta enfocada a ser un API para terceros.

2.2.1. Tipología y palabras clave

Es una herramienta para facilitar el Email Marketing a pequeña y mediana empresa.
Sus palabras clave pueden ser: ENVÍO DE EMAILS MASIVO, EMAIL MARKETING, MARKETING ONLINE, API ENVÍO EMAIL, API EMAIL.

2.2.2. Objetivos del proyecto

Objetivo	Descripción	Tipo
Objetivo 1	Envío de emails de forma masiva	Crítico
Objetivo 2	Asegurar la recepción de los emails a terceros	Crítico
Objetivo 3	Explotar los datos estadísticos de los envíos	Crítico
Objetivo 4	Garantizar la seguridad de estos envíos	Crítico

Objetivo	Descripción	Tipo
Objetivo 5	Facilitar al cliente la gestión de sus campañas de email	Crítico
Objetivo 6	Mejorar la captación de clientes a empresas	Prioritario
Objetivo 7	Proporcionar una nueva herramienta de comunicación a las empresas	Secundario
Objetivo 8	Hacer que sea “Magic” la experiencia de envío de una campaña a toda la base de datos de un cliente	Prioritario
Objetivo 9	Facilitar el trabajo a los técnicos que intervienen en envíos masivos de campañas	Secundario

Tabla 1: Objetivos del proyecto

2.2.3. Definiciones, acrónimos y abreviaciones.

Email: Comunicación que se realizará entre la empresa y sus clientes. El emisor será la empresa y el receptor será su cliente.

Marketing: Hacer marca, la empresa que adquiera el producto, podrá generar marca con ella.

Campañas: Envío masivos de una idea o producto de una empresa.

SMTP: Protocolo el cual interviene en la comunicación.

Spam: Posible bloqueo de los emails enviados.

Listas negras: Listas las cuales tienen un registro de IP’s consideradas responsables del envío masivo de emails sin control.

Listas blancas: Listas las cuales tienen un registro de IP’s con buena reputación para poder hacer el envío de emails.

Explotación de estadísticas: Toda la información interesante sobre el envío de una campaña a través de Magic SMTP.

2.2.4. Partes interesadas

Stakeholders

Nombre	Descripción	Responsabilidad
Pablo Pedrosa	Responsable de la entidad	Se encarga de supervisar el proyecto y de que todo vaya sobre el tiempo previsto. Participa en los requisitos y en la funcionalidad.
Pablo Pedrosa	Responsable técnico	Se encarga de proponer ideas sobre el desarrollo del proyecto.
Yolanda Benítez	Director del proyecto	Supervisa el trabajo del alumno y evalúa el proyecto.

Tabla 2: Stakeholders

Perfiles de usuario

Nombre	Descripción	Responsabilidad
Pablo Pedrosa	Administrador del sistema	Gestión y control del sistema. Control de usuarios dados de alta y responsable de que todo funcione bien.
Técnico empresa	Usuario experto	Gestión de las campañas, tanto en el envío cómo en el control de estadísticas. Uso a modo de API
Comercial empresa	Usuario no experto	Gestión de las campañas, tanto en el envío cómo en el control de estadísticas. Uso desde un cliente de correo.

Tabla 3: Perfiles de usuario

Project Team

Nombre	Descripción	Responsabilidad
Pablo Pedrosa	Gestor del proyecto (GP)	Define, gestiona, planifica y controla el proyecto.
Pablo Pedrosa	Analista (A)	Creación del estudio de viabilidad y la planificación prevista. Analiza la aplicación: arquitectura, metodología, especificaciones...
Pablo Pedrosa	Programador (P)	Desarrolla la aplicación de acuerdo con el análisis y la planificación prevista. Participa en el proceso de validación.
Pablo Pedrosa	Técnico de pruebas (TP)	Participa en el diseño de las pruebas internas y externas. Realiza las pruebas y participa en el proceso de control de calidad
Pablo Pedrosa	Asistente de Marketing (AM)	Proporciona documentación necesaria para poder usar el software
Yolanda Benítez	Directora del proyecto (DP)	Supervisa el trabajo del alumno. Participa en la planificación y validación.

Tabla 4: Project Team

2.3. Estudio de la situación actual del marketing online

Ahora mismo en el mundo del email marketing tenemos soluciones las cuales hacen que tengas que migrar todo lo que tengas en tu sistema habitual de envío de emails, al software de email marketing.

Queremos que los usuarios se conecten vía SMTP y no tengan que migrar las bases de datos de sus clientes de correo.

Las empresas quieren hacer campañas de marketing pero para hacerlo tienen que adaptarse a las nuevas herramientas.

La idea es proporcionar una pequeña documentación sobre como conectarse a Magic SMTP y poder tener todas la funcionalidad

y características que te proporciona cualquier software de email marketing. La pequeña y mediana empresa se puede beneficiar de esto ya que ha un precio muy bajo puede tener todo lo que le proporcionan las grandes empresas de email marketing.

Muchas de estas pequeñas y grandes empresas, ni siquiera saben del potencial y impacto que puede tener una campaña de email, ya que todos tenemos como mínimo una cuenta de email.

Tampoco son conscientes de lo importante que es hacer una buena base de datos de clientes para poder explotarla con herramientas así.

Magic SMTP quiere proporcionar esa canal mágico de comunicación con muy poco trabajo por parte del cliente que quiera enviar sus campañas.

2.3.1. Lógica del sistema

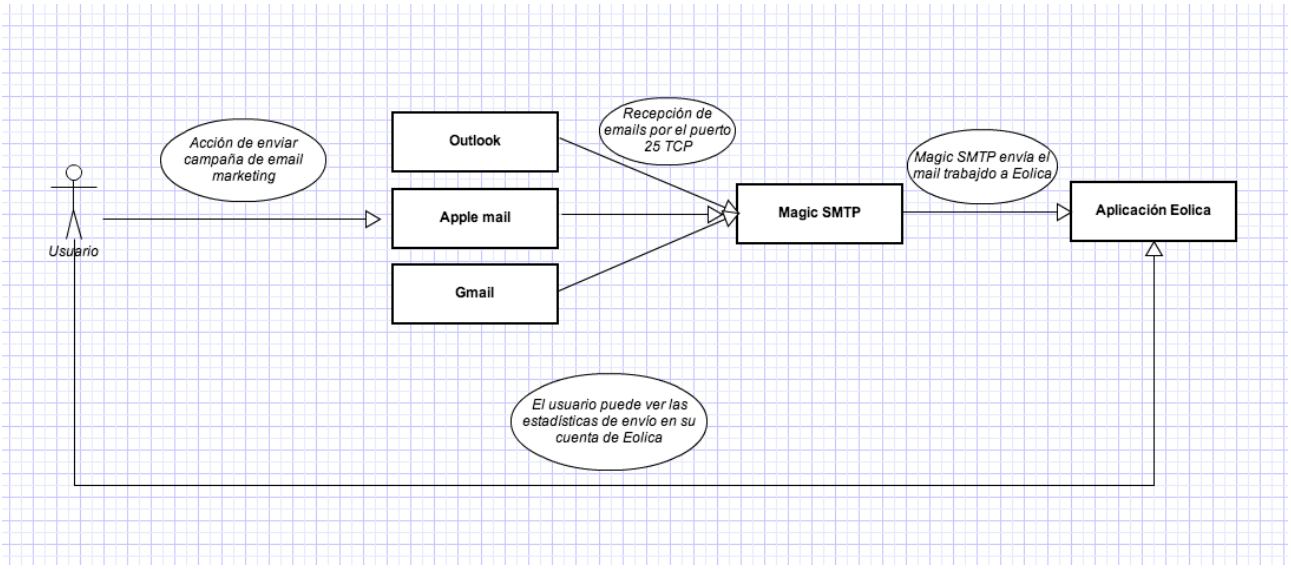

Ilustración 1: Lógica de como funciona el servicio Magic SMTP

2.3.2. Usuarios y/o personal del sistema

Nombre	Descripción	Responsabilidad
Responsable campaña de marketing, comercial	Usuario que genera campañas	Genera la campaña de marketing y utiliza el sistema para notificarla a sus clientes.
Responsable campaña de marketing, técnico	Usuario que genera campañas conectándose a la API	Genera la campaña de marketing y utiliza el sistema para notificarla a sus clientes. Lo hace mediante el sistema de envío de su lenguaje de programación.
Técnico de aplicación	Usuario que se encarga de que todo vaya bien	Se encarga de comprobar que las Direcciones IP que envían estén siempre en listas autorizadas por los grandes dominios de recepción de correo electrónico.

Tabla 5: Usuarios y personal del sistema

2.3.3. Diagnóstico del sistema

Se analiza la situación actual en el email marketing:

Deficiencias:

- El usuario generador de campañas tiene que exportar su base de datos de contactos al software de Email Marketing.
- Envíos puntuales, muy segmentados, sin posibilidad de tener estadísticas desde un cliente de correo habitual.
- Aprender a manejar un software nuevo de Email Marketing.

Mejoras:

- Posibilidad de realizar Email Marketing con el software de envío de email habitual.
- No aprender a manejar un software nuevo
- Posibilidad de conexión vía API para desarrolladores, con estadísticas de envío
- Estadísticas en un envío puntual a un usuario

2.3.4. Normativas y legislación

Actualmente existe la LOPD que nos obliga a tener un cierto cuidado con los datos que tenemos para hacer envíos masivos de email. Todos los emails enviados a través de la plataforma tienen que tener un enlace para poder darse de baja de esa suscripción, o de ese envío puntual, para notificar al emisor de que no quiere recibir mas emails suyos.
También tenemos el problema del Spam, hay que cumplir una serie de normas dentro del mundo del email marketing para que no te confundan con un Spammer, Algunas de ellas son:

- Definir un registro SPF (Sender Policy Framework) dentro de tu dominio. Los servidores de correo consultaran el domino del emisor para ver si este puede enviar en nombre del servicio. El cliente x que utilice Magic SMTP tiene que incluir el SPF de Magic SMTP ya que este enviará el correo y el servidor al que ha sido enviado consultará el FROM del envío para saber si está autorizado a enviar.

- Monitorizar tus Direcciones IP de envío en listas negras para saber si estas están en las listas.

- Ganar prestigio de tus direcciones IP progresivamente, ya que si un servidor de correo comprueba que eres una dirección IP nueva intentando enviar muchos emails a la vez, te considerará Spammer.

2.4. Requisitos del sistema

Aquí veremos la lista de funciones de la aplicación. Los requisitos funcionales son aquellas características requeridas del sistema que expresan una capacidad de acción del mismo, expresadas en una declaración verbal.

La lista es la siguiente:

2.4.1. Requsitos funcionales

1.- Acceso a un API sencillo.
2.- Envío de emails de forma segura.

3.- Tratamiento de emails para una explotación estadística.
4.- Generación de estadísticas de apertura, clicks, etc...
5.- Generar una respuesta para el cliente cuando realiza un envío.
6.- Documentar el API para una conexión sencilla.
7.- Utilizar los sistemas de envío habituales.

2.4.2. Requisitos no funcionales

1.- Cumplimiento de todas las normas anti Spam en internet.
2.- Cumplimiento de las normas de la LOPD.
3.- Seguridad de la conexión.
4.- Filtrar posibles ataques.

2.4.3. Restricciones del sistema

1.- Aplicación pasiva. Tiene que escuchar un puerto y actuar cuando los clientes quieran enviar.
2.- Entorno Java
3.- El proyecto tiene que finalizar antes del 1 de Septiembre de 2013

2.4.4. Prioridades de los requisitos funcionales

	RF1	RF2	RF3	RF4	RF5	RF6	RF7
Esencial	X	X	X	X	X		X
Condicional						X	
Opcional							

Tabla 6: Prioridades requisitos funcionales

2.4.5. Prioridades de los requisitos no funcionales

	RNF1	RNF2	RNF3	RNF4
Esencial	X	X	X	X
Condicional				
Opcional				

Tabla 7: Prioridades requisitos no funcionales

2.4.6. Relación objetivos con requisitos

	RF1	RF2	RF3	RF4	RF5	RF6	RF7	RNF1	RNF2	RNF3	RNF4
Objetivo 1	X	X					X	X	X	X	
Objetivo 2	X	X			X	X	X	X	X	X	X
Objetivo 3			X	X			X		X	X	
Objetivo 4	X	X					X			X	
Objetivo 5	X	X					X				
Objetivo 6	X	X	X	X	X	X	X	X	X	X	X
Objetivo 7	X	X				X		X	X	X	
Objetivo 8	X	X	X	X	X	X	X	X	X	X	X
Objetivo 9	X					X				X	X

Tabla 8: Relación objetivos con requisitos

2.5. Modelo de negocio

Magic SMTP sigue el modelo de tarifa plana de envío, es decir, los usuarios para poder utilizar los servicios contratarán un paquete de envíos. Este contrato les proporcionará acceso a la cuenta y podrán enviar, una vez pasado el límite de envío establecido en el paquete, los envíos sobrantes se le cobrarán a una cuota superior que la del contrato.
Se ofrecerían servicios premium, incluyendo plantillas especificas para envíos determinados.

También se ofrecería un servicio específico para el cliente que lo requiera. Este servicio se basaría en cualquier desarrollo que necesite la empresa para la fácil integración con Magic SMTP.

Dentro de los paquetes descritos más arriba podríamos ofrecer tres tIpos de paquete.

Tipo básico (Pack Básico):
- Hasta mil envíos de email al mes
- Plantilla única para envíos de texto plano.
- Estadísticas de envío.
- Precio 10 Euros/mes

Tipo Premium (Pack Premium):
- Hasta diez mil envíos de email al mes.
- Plantillas predefinidas para envíos de texto plano.
- Estadísticas de envío
- Precio 60 Euros/mes

Tipo Profesional (Pack Profesional):
- Hasta cien mil envíos de email al mes.
- Opción para desarrollo de plantilla personalizada para envíos de texto plano.
- Estadísticas de envío.
- Posibilidad de limpieza de base de datos.
- Precio 500 Euros/mes.

Defino 3 paquetes que podrían ser a nivel usuario, pequeña empresa y mediana empresa. Magic SMTP se puede adaptar a cualquier tipo de caso, al ser un API, permite flexibilidad a la hora de definir un paquete personalizado para un cliente.

2.6. Alternativas y selección de la solución

2.6.1. Alternativa 1: MailChimp

MailChimp es una herramienta muy conocida en el mundo del Marketing online. Es una herramienta muy potente y se podría decir que es la número uno. Una vez dicho esto, esta es la herramienta la cual va a buscar una gran empresa ya que su servicio es muy potente, con Magic SMTP pretendo llegar a la pequeña y mediana empresa que se inicia en el mundo del email marketing, con un servicio al cliente más personalizado.

Ilustración 2:Logo de MailChimp

El modelo de negocio de MailChimp consiste en envío gratuito pero a un número limitado de suscriptores(emails). Magic SMTP se centrará en el envío de emails en las diferentes campañas.

Monthly price	$	$10	$15	$30	$50	$75	$150	$240
Subscribers		0-500	501-1,000	1,001-2,500	2,501-5,000	5,001-10,000	10,001-25,000	25,001-50,000
Send limit		Unlimited	Unlimited	Unlimited	Unlimited	Unlimited	Unlimited	Unlimited

Ilustración 3:Precios MailChimp

MailChimp ofrece todo tIpo de formularios para conseguir suscrIptores a sus clientes, es decir, hacer una buena base de datos para tus envíos. Magic SMTP parte de que el cliente ya tiene una base de datos y quiere hacer el envío desde su herramienta habitual, cliente Outlook por ejemplo. A este cliente ofrecerle unas estadísticas de envío.

2.6.2. Alternativa 2: Amazon SES

Quién no conoce Amazon, pues también tienen un servicio de envío de emails. Este servicio se puede considerar más API que MailChimp, ya que lo único que hace es enviar emails, no permite personalizar tus envíos en su plataforma. Amazon presume de sus años en internet y su experiencia hace que sea fiable a la hora de hacer los envíos. Amazon también está enfocado a la gran empresa, Magic SMTP sería más a nivel pequeña y mediana empresa.

Ilustración 4: Amazon SES

El modelo de negocio de Amazon SES es como el de Magic SMTP, tendrá beneficio por email lanzado. Al ser enfocado como API, el modelo es muy sencillo.

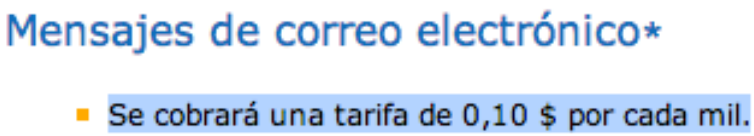

Ilustración 5: Precios Amazon

La alternativa Amazon requiere de un conocimiento previo informático, Magic SMTP quiere conseguir que el único conocimiento que necesitemos sea el de configurar una cuenta de SMTP en un cliente de correo.
También está enfocado a gran empresa o a grandes desarrollos que necesiten de grandes volúmenes de envío.

2.6.3. Alternativa 3: Canalmail

Canalmail ofrece unos servicios de email marketing orientados a negocios que quieren ampliar sus bases de datos, es decir, ellos te proporcionan una base de datos conocida, segmentada y según tu negocio puedes hacer un envío a ese base de datos segmentada.

Ilustración 6: Logo Canalmail

El precio de Canalmail requiere de una alta en sus servicios para saberlo. Entiendo que el precio irá enfocado a envíos y resultado de estos envíos. Magic SMTP no ofrecerá bases de datos segmentadas, sino una herramienta de envío para aquellos que ya la tienen. Es la alternativa que menos se parece a Magic SMTP ya que no está enfocada a un API.

2.7. Solución propuesta

Las soluciones propuestas están enfocadas a la gran empresa. Es decir, están orientados a grandes volúmenes de envío. MailChimp y Canalmail están enfocados a usuarios que no requieran de un nivel técnico avanzado para hacer los envíos, Amazon en cambio si requiere de algún conocimiento para su implementación. Ninguno de los tres puede ofrecer la flexibilidad a la hora de hacer envíos que puede ofrecer Magic SMTP. Magic SMTP se puede adaptar a las necesidades de las pequeñas empresas y puede cubrir su Marketing online de forma muy personalizada.

Justificamos la elección de Magic SMTP para aquellas empresas cuales quieran un envío personalizado de sus bases de datos sin necesidad de una integración y desde la plataforma la cual ya hacían sus envíos (clientes de correo).

2.8. Conclusiones

Beneficios:

1.- API de conexión para hacer envío fácil de usar.

2.- Estadísticas para terceros.

3.- No migrar bases de datos, se integra en aplicaciones de terceros.

4.- Centralizar la lucha antispam de los clientes.

5.- Mejorar la llegada de emails a usuarios finales.

Inconvenientes:

1.- Lucha antispam continúa en el tiempo.

2.- Integración en aplicaciones de terceros.

Beneficios + Inconvenientes = **Proyecto Viable**

3. PLAN DE PROYECTO

3.1. Introducción

El proyecto es un sistema de Email Marketing en forma de API, para facilitar las campañas de Marketing a sus clientes.
Se intentará dar un punto de vista técnico, se verán los requisitos del mismo y especificaciones que necesita Magic SMTP.
Se mostrarán las herramientas de desarrollo y los métodos usados dentro de las mismas.
También se verá el calendario de las tareas desarrolladas.

3.1.1. Definiciones, acrónimos y abreviaciones

- Java: software de desarrollo.
- GIT: software de control de versiones.
- Microsoft Project: software para el control del desarrollo del proyecto.
- SMTP: Send Mail Transfer Protocol, Protocolo usado para gestionar los envíos de email
- Diagrama de Gantt: Cronología del proyecto.

3.2. WBS (Work Breakdown Structure)

Work Breakdown Structure o WBS, (estructura de descomposición del trabajo o EDT), es la lista de todas las fases, actividades y tareas del proyecto.

3.2.1. Fases y actividades del proyecto

Fases	Descripción
Iniciación	Fase de iniciación. Incluye todas las actividades de definición del proyecto, asignación y matriculación.
Planificación	Incluye el Estudio de viabilidad y el Plan del Proyecto.
Análisis	Análisis de los requisitos funcionales y los no funcionales. Arquitectura del sistema.

Fases	Descripción
Diseño	Incluye el diseño de la capa de datos, de control y la interfaz. Diseño de los test.
Desarrollo	Fase de desarrollo de la aplicación.
Test y pruebas	Fase de prueba del sistema. Incluye tests unitarios y de integración.
Implantación	La aplicación se instala en su entorno real. Incluye la formación a usuarios.
Generación de documentos	Fase de documentación del proyecto. Incluye manuales y memoria del proyecto.
Cierre del proyecto	Fase de cierre. El director del proyecto firma la aceptación y cierre de proyecto.
Defensa del proyecto	Defensa del proyecto delante de la comisión, tribunal.

Tabla 9: Fases y actividades del proyecto

3.2.2. Diagrama WBS

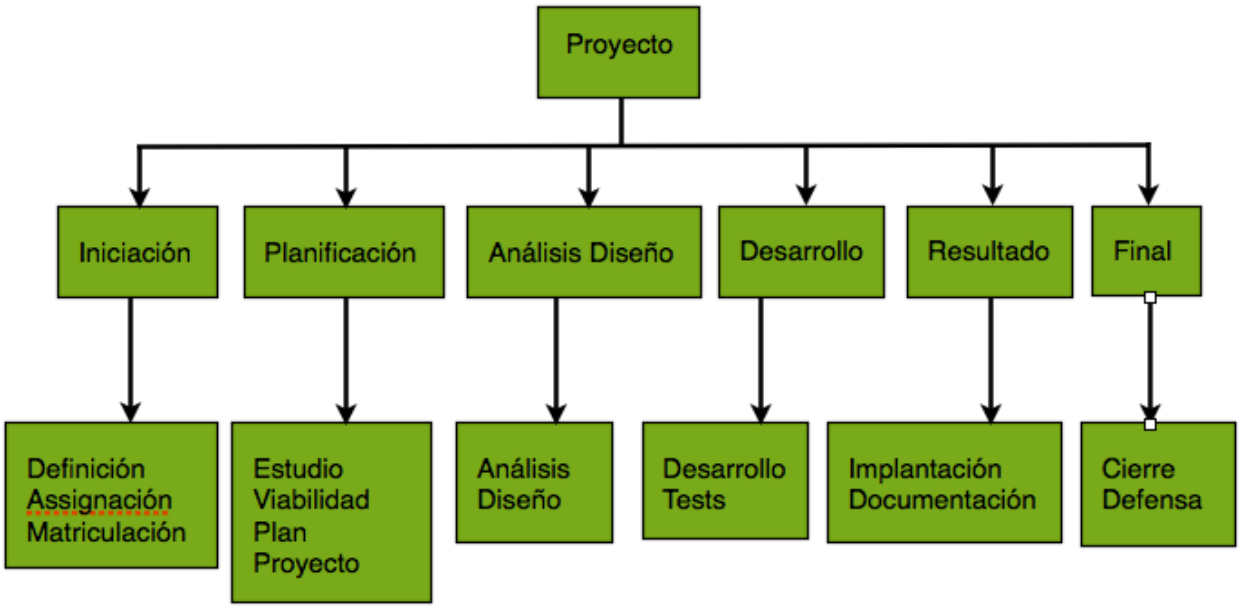

Ilustración 7: Diagrama WBS

3.2.3. Milestones

- Iniciación: 12/10/2012
- Planificación: 12/12/2012
- Análisis y diseño: 31/01/2013
- Desarrollo: 19/04/2013
- Resultado: 31/05/2013
- Final: 10/06/2013

3.3. Recursos del proyecto

Recursos humanos y materiales asignados a cada tarea del proyecto

3.3.1. Descripción de los recursos

Recursos materiales: Se utilizará el software de desarrollo Java, libre para el uso público.

Costes indirectos: amortización de los recursos de desarrollo.

A continuación una tabla donde se clasifica el valor de las horas realizadas en el desarrollo; recursos humanos:

Recursos humanos	Valoración
Jefe de proyecto	60 €/h
Analista	45 €/h
Asistente de marketing	20 €/h
Programador	30 €/h
Técnico de pruebas	15 €/h

Tabla 10: Horas de desarrollo

3.3.2. Calendario de los recursos

Los recursos humanos se utilizarán en el ciclo de vida del proyecto:

- **Jefe de proyecto:** Iniciación, Planificación, Generación de documentos, Cierre y Defensa.
- **Analista:** Análisis y diseño, Implantación y Puntos de control de análisis, diseño y desarrollo.
- **Asistente de marketing:** Análisis y diseño, Implantación y Puntos de control, diseño y desarrollo, fase de pruebas.
- **Programador:** Diseño, Desarrollo, Test e implantación.
- **Técnico de pruebas:** Fase de test.

3.4. Calendario del proyecto

El proyecto debe seguir un calendario para obtener los resultados deseados.

Calendario del proyecto: El proyecto se desarrolla de Octubre de 2012 a Junio de 2013. El total de horas dedicadas al proyecto será de 210 horas.

Fecha de inicio: 10 Octubre de 2012

Fecha de finalización: 10 Julio de 2013

Herramientas de planificación y control: Microsoft Project.

3.4.1. Dependencias

Las fases pueden sobre ponerse unas a otras ya que pueden surgir cambios en algunas fases.
Las tres primeras fases del proyecto tienen dependencia entre ellas, el desarrollo puede hacer que se modifiquen aspectos de diseño, pero el resto sería lineal. Se acaba una y empieza otra.

3.4.2. Cuadro de tareas del proyecto

Nombre de tarea	Predecesoras	Duración
Proyecto Magic SMTP (total horas)		412 horas
Asignación y matriculación del proyecto		2 horas
Inicio del proyecto: asignación y matriculación del proyecto (total horas tareas)		42 horas
Estudio de viabilidad	2	20 horas
Aprobación del estudio de viabilidad	4	1 hora
Plan de proyecto	5	20 horas
Aprobación del plan de proyecto	6	1 hora
Análisis de la aplicación (total horas)	3	44 horas
Análisis de los requisitos (casos de uso)	7	15 horas
Análisis de datos (modelo, base de datos)	9	15 horas
Análisis de la seguridad y la legalidad	10	10 horas
Documentación del análisis	11	3 horas
Aprobación del análisis	12	1 hora
Diseño de la aplicación (total horas)	8	46 horas
Diseño de la base de datos	13	20 horas
Diseño modular de la aplicación	15	25 horas
Aprobación del diseño	16	1 hora
Desarrollo de la aplicación (total horas)	3	224 horas
Instalación librería SubbEtha SMTP	17	20 horas
Codificación escucha de puerto TCP	19	4 horas
Codificación entrada de mail vía SMTP	20	40 horas
Codificación autenticación usuario sistema	21	30 horas
Instalación librería Javamail	22	5 horas
Codificación parseo de email	23	30 horas

Nombre de tarea	Predecesoras	Duración
Codificación integración campaña con sistema Eolica	24	20 horas
Instalación librería Html	25	5 horas
Codificación parseo de html de email	26	15 horas
Codificación sistema estadísticas	27	15 horas
Codificación integración envío de campaña con sistema Eolica	28	20 horas
Empaquetar sistema	29	20 horas
Implantación	18	3 horas
Instalación del servicio	30	2 horas
Integración en cliente de correo	32	1 hora
Generación de documentos (memoria proyecto, presentación proyecto)	3	50 horas
Defensa del proyecto	34	1 horas

Tabla 11: Cuadro de tareas del proyecto

3.4.3. Calendario temporal (MS project)

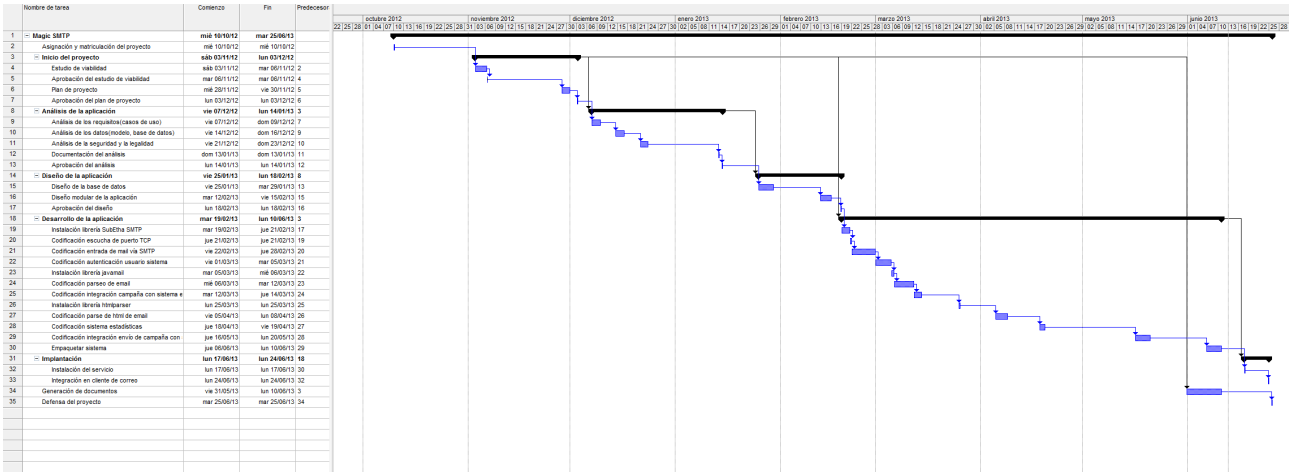

Ilustración 8: Diagrama de Gantt

	Nombre de tarea	Comienzo	Fin	Predecesor
1	**Magic SMTP**	**mié 10/10/12**	**mar 25/06/13**	
2	Asignación y matriculación del proyecto	mié 10/10/12	mié 10/10/12	
3	**Inicio del proyecto**	**sáb 03/11/12**	**lun 03/12/12**	
4	Estudio de viabilidad	sáb 03/11/12	mar 06/11/12	2
5	Aprobación del estudio de viabilidad	mar 06/11/12	mar 06/11/12	4
6	Plan de proyecto	mié 28/11/12	vie 30/11/12	5
7	Aprobación del plan de proyecto	lun 03/12/12	lun 03/12/12	6
8	**Análisis de la aplicación**	**vie 07/12/12**	**lun 14/01/13**	**3**
9	Análisis de los requisitos(casos de uso)	vie 07/12/12	dom 09/12/12	7
10	Análisis de los datos(modelo, base de datos)	vie 14/12/12	dom 16/12/12	9
11	Análisis de la seguridad y la legalidad	vie 21/12/12	dom 23/12/12	10
12	Documentación del análisis	dom 13/01/13	dom 13/01/13	11
13	Aprobación del análisis	lun 14/01/13	lun 14/01/13	12
14	**Diseño de la aplicación**	**vie 25/01/13**	**lun 18/02/13**	**8**
15	Diseño de la base de datos	vie 25/01/13	mar 29/01/13	13
16	Diseño modular de la aplicación	mar 12/02/13	vie 15/02/13	15
17	Aprobación del diseño	lun 18/02/13	lun 18/02/13	16
18	**Desarrollo de la aplicación**	**mar 19/02/13**	**lun 10/06/13**	**3**
19	Instalación librería SubEtha SMTP	mar 19/02/13	jue 21/02/13	17
20	Codificación escucha de puerto TCP	jue 21/02/13	jue 21/02/13	19
21	Codificación entrada de mail vía SMTP	vie 22/02/13	jue 28/02/13	20
22	Codificación autenticación usuario sistema	vie 01/03/13	mar 05/03/13	21
23	Instalación librería javamail	mar 05/03/13	mié 06/03/13	22
24	Codificación parseo de email	mié 06/03/13	mar 12/03/13	23
25	Codificación integración campaña con sistema e	mar 12/03/13	jue 14/03/13	24
26	Instalación librería htmlparser	lun 25/03/13	lun 25/03/13	25
27	Codificación parse de html de email	vie 05/04/13	lun 08/04/13	26
28	Codificación sistema estadísticas	jue 18/04/13	vie 19/04/13	27
29	Codificación integración envío de campaña con	jue 16/05/13	lun 20/05/13	28
30	Empaquetar sistema	jue 06/06/13	lun 10/06/13	29
31	**Implantación**	**lun 17/06/13**	**lun 24/06/13**	**18**
32	Instalación del servicio	lun 17/06/13	lun 17/06/13	30
33	Integración en cliente de correo	lun 24/06/13	lun 24/06/13	32
34	Generación de documentos	vie 31/05/13	lun 10/06/13	3
35	Defensa del proyecto	mar 25/06/13	mar 25/06/13	34

Ilustración 9: Zoom calendario diagrama de Gantt

3.5. Evaluación de riesgos

El proyecto está sujeto a ciertos riesgos para la no finalización del mismo, intentaré mostrar cuales son aquellos que hacen peligrar su realización.

3.5.1. Lista de riesgos

R1. Planificación temporal optimista: Plan del proyecto. Falta de tiempo en el proyecto, aumentan los recursos.

R2. Falta de alguna tarea necesaria: Plan del proyecto. No se cumplen los objetivos del proyecto.

R3. Presupuesto poco ajustado: Plan del proyecto. Falta de prestaciones del proyecto.

R4. Cambio de los requisitos: Estudio de viabilidad. Alarga el tiempo de desarrollo del proyecto.

R5. Herramientas de desarrollo inadecuadas: Desarrollo. Alarga el tiempo de desarrollo, falta de prestaciones del proyecto.

R6. Requisitos mal definidos: Estudio de viabilidad. Alarga extremadamente el proyecto.

R7. Fase de test errónea: Desarrollo. Prestaciones inadecuadas o mal acabadas.

R8. Incumplimiento seguridad: Todas las fases. Proyecto inviable.

R9. Escasez de envíos: Implantación. Económicamente no es rentable.

R10. Denegación de servicio: Implantación. Colapso de la aplicación por mala definición de seguridad.

R11. Abandono del proyecto antes de su finalización: Todas las fases. Pérdida de tiempo y de dinero. Frustración.

3.5.2. Catalogación de riesgos

	Probabilidad	Impacto
R1	**ALTA**	**CRÍTICO**
R2	**ALTA**	**CRÍTICO**
R3	**ALTA**	**CRÍTICO**
R4	**MEDIA**	**MARGINAL**
R5	**BAJA**	**MARGINAL**
R6	**ALTA**	**CRÍTICO**
R7	**MEDIA**	**CRÍTICO**
R8	**MEDIA**	**CRÍTICO**
R9	**ALTA**	**CRÍTICO**
R10	**ALTA**	**CRÍTICO**
R11	**MEDIA**	**CATASTRÓFICO**

Tabla 12: Catalogación de riesgos

3.5.3. Plan de contingencia

	Solución aplicable
R1	Prescindir de prestaciones del sistema. Posibles clientes descontentos.
R2	Modificación de tareas en el Plan de proyecto. Añadir o prescindir de tareas.
R3	Asumir sobrecosto o renegociar precio con el cliente final.
R4	Modificar requisitos, establecer requisitos deseados por todas las partes.
R5	Adquirir herramientas adecuadas, alarga el proyecto.
R6	Redefinir requisitos, rehacer proyecto, muy crítico.
R7	Reprogramar tests, pasar la aplicación a cuarentena. Uso supervisado.
R8	Revisar aplicación, pasar la aplicación a cuarentena. Uso supervisado.
R9	Incrementar la comunicación de la existencia de la aplicación. Mejorar marketing.
R10	Bloquear envíos, mejorar entrada de emails.
R11	No tiene solución.

Tabla 13: Plan de contingencia

3.6. Presupuesto

Presupuesto de la solución que se proporcionará de acuerdo con lo establecido en la planificación del proyecto.

3.6.1. Estimación coste personal

En la siguiente tabla se establecerá el coste total en base a las horas invertidas y el coste de las mismas.

Iniciales	Nombre del recurso	Tasa estándar	Trabajo	Costo
JP	Jefe de proyecto	60€/h	40h	2400 €
A	Analísta	45€/h	60h	2700 €
AM	Asistente de marketing	20€/h	15h	300 €
P	Programador	30€/h	250h	7500 €
TP	Técnico de pruebas	15€/h	8h	120 €
Coste de personal total = 13020 €				

Tabla 14: Estimación coste personal

3.6.2. Estimación costes de los recursos

La siguiente tabla muestra los costes de los recursos del proyecto. Se muestra la amortización de los mismos, en un periodo de 36 meses.

Nombre del recurso	Coste amortización	Coste unitario	Periodo de amortización
MacBook Pro	47€/mes aprox. (36 meses)	1700 €	564€ (12 meses)
iWork	1,5€/mes aprox. (36 meses)	54 €	18€ (12 meses)
MS Project	32,5€/mes aprox. (36 meses)	1169 €	390€ (12 meses)
Hosting	100€/mes aprox. (36 meses)	3600 €	1200€ (12 meses)
Total	181,2€/mes aprox. (36 meses)	6523 €	2174,3€ (12 meses)

Tabla 15: Estimación costes de los recursos

3.6.3. Estimación costes de las actividades

La siguiente tabla muestra los costes generados de todas las actividades realizadas durante el proyecto.

Se notificará a clientes potenciales de la nueva herramienta.

Descripción	Coste
Gastos de transporte	200 €
Marketing aplicación	100 €
Otros	100 €
Total	400 €

Tabla 16: Estimación costes de las actividades

3.6.4. Estimación costes de desarrollo

Descripción	Coste	Coste mensual
Coste mantenimiento	incalculable, hasta fin de vida del software	100 €/mes
Total		100 €/mes

Tabla 17: Estimación costes de desarrollo

3.6.5. Resumen y análisis de costes

La siguiente tabla mostrará el resumen total de costes de la aplicación.

Descripción	Coste	Coste mensual
Coste personal	13020 €	1085 €/mes
Coste recursos	6523 €	543,5 €/mes
Coste actividades	400 €	34 €/mes

Descripción	Coste	Coste mensual
Total	19943 €	Incluiríamos los 100€/mes del mantenimiento

Tabla 18: Resumen de costes

Se puede apreciar un coste mensual que equivaldría al sueldo mensual de un empleado de empresa media actualmente. En una previsión a un año es un proyecto de coste medio, la amortización puede ser rápida con la captación de clientes y de envíos periódicos de los mismos.

4. MARCO TEÓRICO

En este apartado se mostrará una descripción de todas las herramientas usadas para el desarrollo del proyecto.

Se entrará en el detalle de las mismas aportando el aspecto diferencial sobre otras para este proyecto. Dichas herramientas, son los lenguajes de programación, entornos de desarrollo y librerías.

Se comentará también el sistema de alojamiento del mismo.

4.1. Entornos de desarrollo

4.1.1. Eclipse

Entorno de desarrollo(IDE) gratuito. Este entorno de desarrollo es uno de los mas famosos en la comunidad Java. Tiene soporte para muchos tipos de lenguaje, Java, PHP, Coldfusion, etc. Simplemente con la instalación de ciertos plugins ya puedes desarrollar en ese lenguaje.

Proporciona herramientas de debug, permite compilar el proyecto con breakpoints, esto hace que sea mas cómoda la identificación de posibles bugs en el desarrollo.

La fácil instalación de plugins hace que sea muy flexible con el lenguaje que uses. Permite tener herramientas que facilitan el desarrollo, instalando estos plugins de los repositorios que nos ofrece el propio software.

También facilita el desarrollo de pruebas con la herramienta de test unitario.

Quizá es un software, el cual se podría definir como “cañón para matar moscas”. Con esto me refiero a que su consuma de memoria es elevado y se necesita una máquina con prestaciones altas para su uso frecuente. Al tener tantas opciones, hace que la máquina

donde se esté ejecutando sufra sus consecuencias. Con lo de cañón, me refiero que quizá para proyectos de poca envergadura, o de los cuales no necesitemos ciertos plugins, quizá el consumo de memoria es excesivo y puede retrasar el tiempo del proyecto.

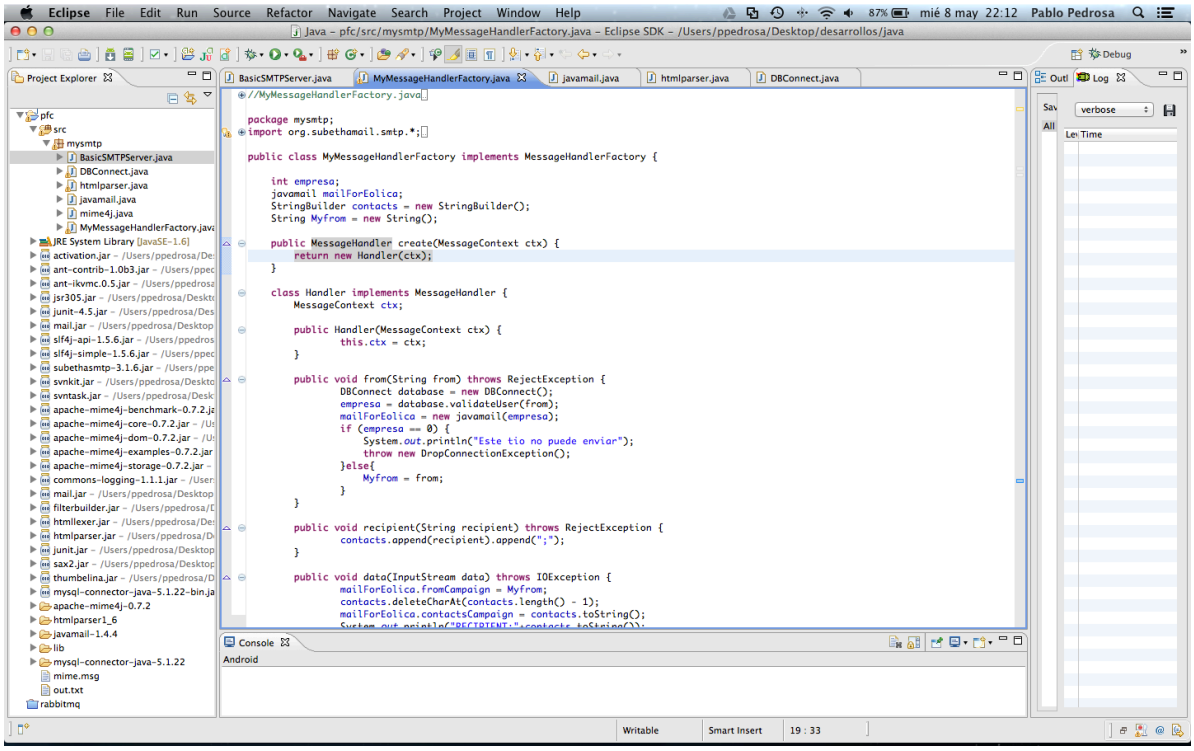

Ilustración 10: IDE Eclipse

Un ejemplo actual de uso común sería la programación Android. Android es una librería "súper heavy", que con cuatro clicks puede ser instalada en Eclipse con sus plugins para el desarrollo de aplicaciones.

Estado actual de Eclipse

Líneas de código fuente	2.063.083
Esfuerzo estimado de desarrollo (persona-año / persona-mes)	604,33 / 7.251,93
Estimación de tiempo (años-meses)	6,11 / 73,27
Estimación del nº de desarrolladores en paralelo	98,98
Estimación de coste	$ 81.636.459

Ilustración 11: Envergadura Eclipse

En conclusión, una herramienta imprescindible para desarrollar en Java, pero que puede ser substituible en desarrollos con otros lenguajes.

Trabaja sobre la licencia Eclipse public licence

4.2. Herramientas

4.2.1. MySQL

Es el sistema gestor de base de datos gratuito más famoso. Es un sistema de base de datos relacional, que permite un número determinado de conexiones simultáneas y también la creación de múltiples usuarios.

Utiliza el lenguaje SQL para la interacción con las bases de datos. Existen todo tipo de conectores y librerías para su uso con los distintos lenguajes de programación. En el caso de Magic SMTP he utilizado la librería Java.sql. Simplemente importando esta librería y la llamada a *Java.sql.DriveManager.getConnection(host,user,password)*, esta nos devuelve un objeto conexión(*Java.sql.Connection*) el cual podemos usar en las futuras consultas, con la llamada a *Java.sql.Connection.prepareStatement.executeQuery()*. Más adelante entraremos en el detalle del código.

Trabaja sobre la licencia GPL, esta permite el uso de la aplicación de forma abierta.

Pertenece a la empresa Sun Microsystems.

4.2.2. MySQL Workbench

Herramienta que permite crear base de datos de una forma fácil e intuitiva. Recordemos que estas son de tipo relacional. No permite el uso de otro tipo de base de datos, como por ejemplo NoSQL.

Esta herramienta ha sido usada para modelar la base de datos, pero puede ser usada para administrar un servidor de base de datos. Esta sección del programa permite ver

cuantas conexiones están activas y cuales de ellas hacer que la base de datos se colapse.

También existe la sección de consultas sobre la base de datos, pero se ha usado otro programa especifico para las mismas.

Trabaja bajo la licencia GPL.

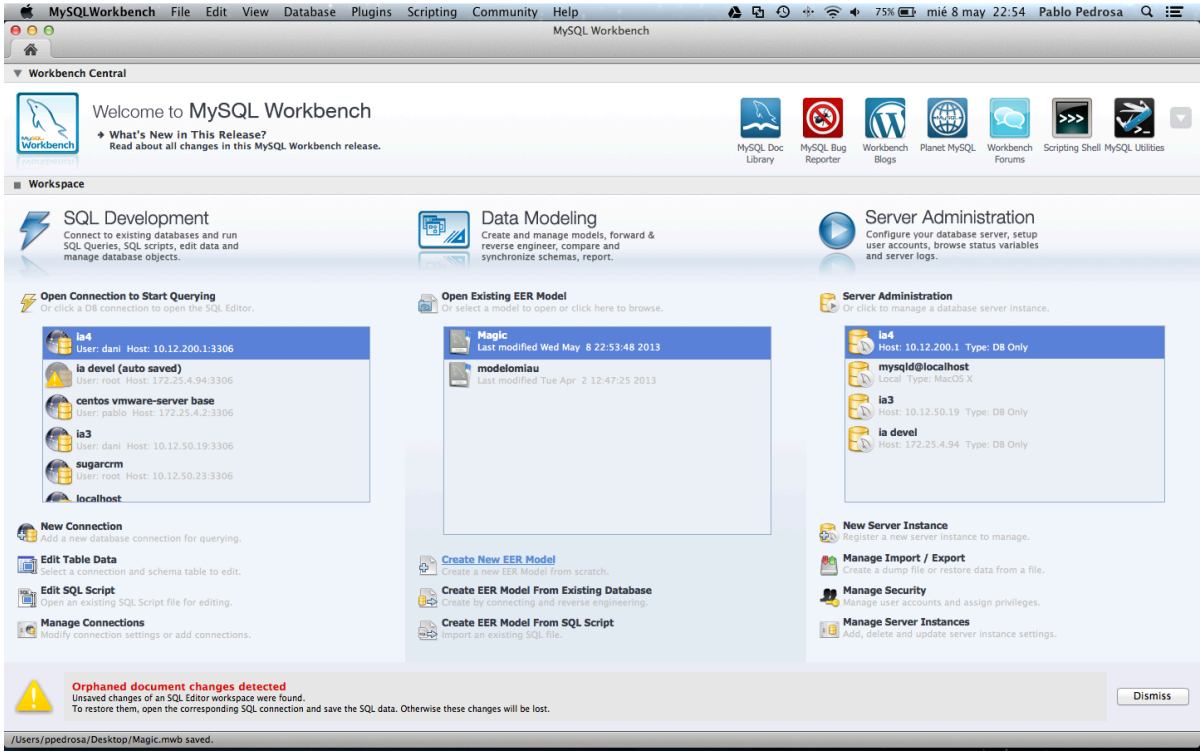

Ilustración 12: MySQL Workbench

4.2.3. Sequel Pro

Sequel Pro es la herramienta usada para hacer las consultas sobre la base de datos. Es una herramienta súper ágil, no consume tanta memoria como MySQL Workbench y al ser dedicada a consultas, tiene mejores prestaciones.

Es un programa esencial para hacer test y Debug de las consultas que acabarán siendo las protagonistas del código fuente Java. Si no usamos un ORM, las consultas SQL son las que al final acabará ejecutando el Drive Java para recuperar todo aquello que haga

falta. Si estas consultas no son testeadas, podemos encontrarnos con resultados no esperados, o consultas lentas. Esto puede ser catastrófico para el proyecto.

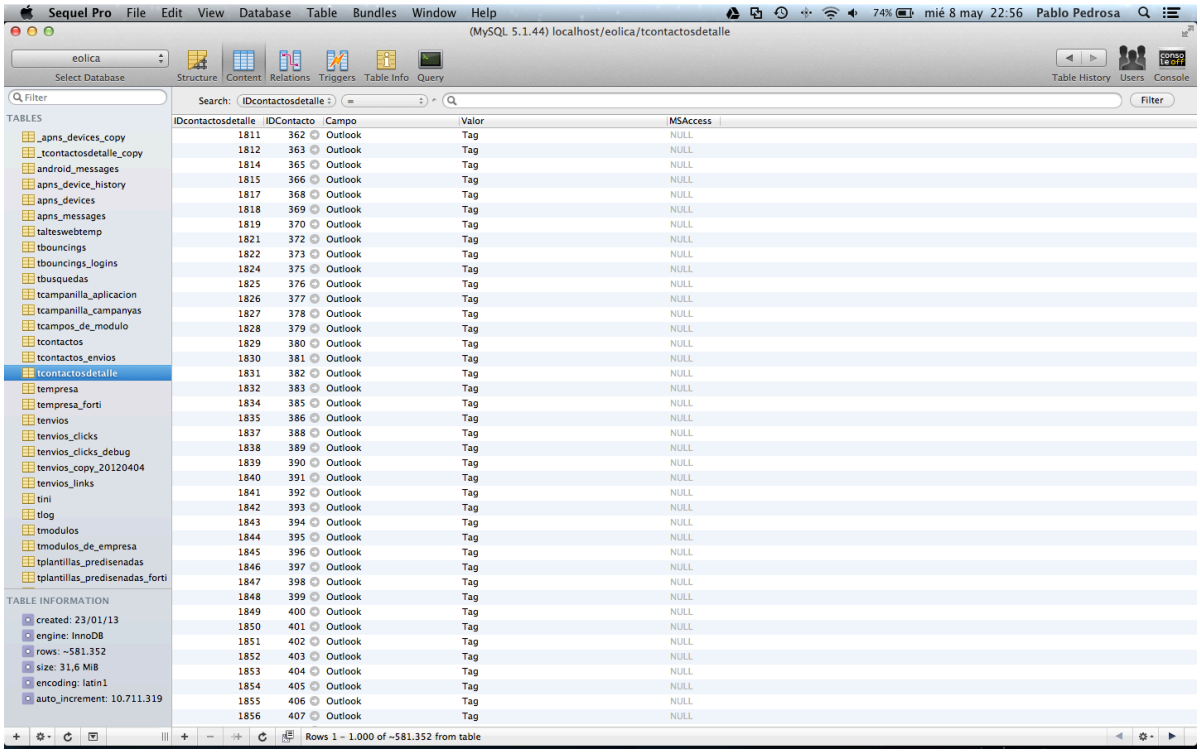

Ilustración 13: Sequel Pro

4.2.4. Java Virtual Machine (JVM)

La JVM es imprescindible para ejecutar todo proceso programado en el lenguaje Java. Esta se encarga de interpretar el bytecode creado por el compilador de Java (JDK). A nivel arquitectónico se sitúa justo por encima del hardware y hace de puente entre el bytecode y el lenguaje nativo del dispositivo.

Toda aplicación Java está pensada ser interpretada por esta JVM.

Este concepto de JVM es transparente para el programador Java ya que va integrada en toda instalación del software. Pero creo que se debe aportar a la memoria y se debe

conocer ya que sin esta, no podríamos ejecutar nuestros programas Java.

La JVM fue desarrollada por Sun Microsystems. Cualquier implementación de JVM debe ser aprobada por Sun Microsystems.

Gracias a esta “máquina virtual”, podemos ejecutar un .class en entornos Windows y Linux, simplemente instalando la JVM necesaria en cada entorno. “Write once, run anywhere”.

4.2.5. Git

Herramienta para la gestión de versiones del software en desarrollo. Git consiste en un sistema que permite tener varias implementaciones de versiones de nuestro software en paralelo, para no perder nada de lo que vamos desarrollando. Estas versiones en paralelo son las ramas(branches). Aquí podemos ir guardando nuestros desarrollos sin tener que modificar el código fuente que tenemos en producción. Con este sistema podemos confiar siempre en la rama de producción como código estable.

Git fue diseñado por el famoso Linus Torvalds, el padre de Linux.

4.3. Lenguajes de programación

4.3.1. Java

Java es un lenguaje orientado a objetos desarrollado por Sun Microsystems. Es un lenguaje el cual precisa de un compilador para ser interpretado por una JVM previamente explicada. Es de alto nivel, por este motivo precisa del compilador para traducirlo a bajo nivel (bytecode).

Una de sus principales ventajas es que gracias a la Java Virtual Machine permite ejecutarse en cualquier tipo de hardware “write once, run anywhere”. Gracias al compilador JIT tenemos que la JVM transforma el bytecode al código nativo de la máquina donde se está ejecutando.

```java
// Hola.java
import javax.swing.JOptionPane;
public class Hola //Declara una clase llamada Hola, que es descendiente de la clase Object
{
//Entre llaves se declaran los atributos y métodos de la clase

    public static void main(String[] args)
    //public: indica que el método main()es público
    //void: indica que la función main() no devuelve ningún valor
    //El método main()debe aceptar siempre como parámetro un vector de strings
   {
        JOptionPane.showMessageDialog(null, "Hola Mundo");
    //Esta línea indica que se va a ejecutar el método showMessageDialog(), encargado de mostrar
    //en un cuadro de diálogo un valor (en nuestro caso, un String)
    //El primer parámetro no es necesario y podemos usar el valor nulo
    //(el parámetro indica el JFrame asociado, el contenedor o entorno del diálogo a crear)

    }
}
```

Ilustración 14: Ejemplo código Java

El lenguaje fue desarrollado en 1995 y a lo largo de los años han sido muchos los desarrolladores que han invertido horas para generar miles de aplicaciones con este lenguaje. De esta forma, encontramos todo tipo de librerías para hacer casi cualquier cosa con nuestras computadoras. En Magic SMTP he utilizado varías librerías escritas en Java para poder realizar todo el proceso de escucha, clasificación y parseo de email.

Ilustración 15: Logo Java

Dos elementos importantes para que un programa Java funcione son el JDK y JRE. JDK (Java Development Kit) es el software que incluye las herramientas necesarias para programar en Java. Una de ellas es el compilador de Java, código alto nivel a código bajo nivel (bytecode). JRE (Java Runtime Environment) es el software que permite ejecutar un programa Java. JRE incluye una JVM y se puede decir que es el “intermediario entre” el sistema operativo y Java.

Para finalizar, como opinión personal, es un lenguaje que permite hacer muchas aplicaciones de ámbitos completamente distintos, pero con la experiencia que tengo en otros lenguajes, es un tanto estricto a la hora de hacer cosas sencillas y eso requiere de un conocimiento exhaustivo del lenguaje. Un ejemplo de esta afirmación, se podría ver en los cast de los valores, o en los valores que parecen simples y que son objetos como el tIpo String.

4.3.2. HTML

El famoso HTML (HyperText Markup Language). Este lenguaje es el dominante en cuanto a páginas web. Se utiliza para describir y traducir la estructura y la información en forma de texto, así como para complementar el texto con objetos tales como imágenes. A diferencia de Java, es un lenguaje interpretado, es decir, no necesita de un compilador para ser interpretado por una máquina. Los navegadores web se encargan de interpretar los archivos HTML para mostrar la información.

Es un lenguaje que funciona con un sistema de etiquetas sencillo. Normalmente una estructura HTML comienza por la etiqueta *<html>* y termina con su cierre *</html>*. Podemos distinguir entre dos partes importantes dentro del HTML. El tag *<head></head>* que es donde se define la cabecera del documento HTML. Sirve para dar información al navegador sobre recursos que necesita para mostrar la información de manera correcta.

```
<!DOCTYPE HTML>
 <html>
   <head>
     <title>Ejemplo1</title>
   </head>
   <body>
     <p>ejemplo1</p>
   </body>
 </html>
```

Ilustración 16: Ejemplo HTML

La segunda parte sería el tag *<body><body>*. En esta parte se define el contenido principal del HTML. Aquí está definida toda la información que va a ser vista por el usuario en el navegador. Gracias a otras etiquetas podemos distribuir de forma flexible la información.

Dentro de la gran variedad de etiquetas HTML que existen, podemos distinguir dos grupos, los elementos o etiquetas de bloque y los elementos de linea. La diferencia, un elemento de bloque no puede aparecer dentro de uno de linea, en cambio, uno de linea si puede aparecer dentro de uno de bloque. Por ejemplo la etiqueta * Lorem Pisum... * es un elemento de bloque que indica un párrafo con el texto “Lorem Pisum”. En cambio la etiqueta ** es un elemento de linea que puede alterar la visualización de un texto dentro de un párrafo, * Lorem Pisum * quiere decir que el texto “Lorem” irá en negrita dentro del párrafo. Quedaría “**Lorem** Pisum”.Actualmente existe el HTML5. Esta denominación viene atribuida al uso de nuevas etiquetas HTML y la combinación de CSS3 y Javascript. Esta combinación permite mostrar autenticas obras de arte en nuestras páginas web.

CSS o CSS3 (Cascade Style Sheets) permite dar estilo a los tags HTML. Javascript permite que podamos interactuar con los tags HTML.

Para Magic SMTP hemos usado el conocimiento de HTML para su PARSEO y modificación para explotar todas las estadísticas necesarias para el usuario. Gracias al conocimiento de los tags (etiquetas) he podido modificar los links y insertar el píxel de seguimiento para la confirmación de apertura.

4.4. Librerías utilizadas

4.4.1. SubEtha SMTP

SubEtha SMTP ha sido una librería imprescindible para Magic SMTP. Al evaluar las características de Magic SMTP, una de ellas es que tenía que ser un servidor SMTP. Más que una característica, es su función core.

Tenía dos opciones, desarrollar un servidor SMTP desde cero, o, bien apoyarme en algún software ya desarrollado o una librería que me ofreciera las propiedades de un servidor SMTP, escucha de un puerto 25 TCP y permitirme trabajar con el email recibido para su tratamiento.

SubEtha SMTP fue la opción perfecta. Esta librería es capaz de ejecutarse en un puerto 25 TCP y poder recibir emails.

Al principio fue difícil encontrar una librería que fuese capaz de escuchar el puerto 25 y poder permitir al programador manipular la entrada de flujo de datos.

La librería devuelve todo lo que recibe en la conversación SMTP. Recibimos el MAIL FROM, el RCPT TO y el DATA. Es capaz de recibir más instrucciones SMTP.

En el MAIL FROM tenemos el remitente del envío, nos permite identificarlo y saber si puede enviar o no, así como asociar la campaña y generarle las estadísticas.

En el RCPT TO recibimos todos los destinatarios del envío, estos los recibimos en un método del objeto receptor. Este método es llamado para cada destinatario. La aplicación los recibe y los concatena separados por ";".

En el DATA recibimos el cuerpo del email, con todas las cabeceras del mismo, este contenido se lo pasamos a la librería Javamail para tratarlo de forma que podamos generar estadísticas y el envío.

SubEtha está desarrollada bajo la licencia Apache license 2.0.

4.4.2. Javamail

Javamail es la librería de los emails. Con esto quiero decir, que es la reina del parseo de email. Es capaz de devolver cualquier dato que venga en el cuerpo del email.

La estructura de un email es un tanto compleja, sus cabeceras identifican muchos parámetros del envío. Javamail mastica esas cabeceras y nos da todos los datos, bueno los que le pidamos.

Gracias a sus "Getters" podemos hacer un simple *getSubject()* y nos devuelve el asunto del email.

Librería imprescindible para poder sacar toda la información del email. Esto se podría haber programado, pero hubiese hecho que el proyecto fuese inviable en un tiempo razonable, ya que el parseo de un email es complejo.

Javamail se ha utilizado para extraer todos los datos necesarios para poder hacer el envío y poder asociar el mismo a un usuario y a sus estadísticas.

Ejecutando el *getContent()* podemos extraer el body del email, lo que verá el usuario final. Este body es el que pasamos a la siguiente librería, HTML Parser.

Javamail está desarrollada por Oracle y se ejecuta bajo la licencia Common Development and Distribution License (CDDL).

4.4.3. HTML Parser

Librería imprescindible para la interpretación de los bodies de los emails. Cuando el body que recibimos es de tipo HTML, HTML Parser desglosa todas sus etiquetas, siendo sencillo el poder interactuar con el HTML.

Cuando digo interactuar, me refiero a poder modificar el contenido del HTML, respecto al enviado originalmente, para poder generar datos que mostraremos al usuario final.

En Magic SMTP se ha usado para implementar plantillas dentro de un envío sin HTML, para introducir parámetros en los links del mismo para poder hacer un tracking de los mismos. Con lo de tracking, me refiero a saber a que links a clicado el usuario que recibe el email.

También nos permite añadir un píxel de seguimiento, este píxel es parte de la magia. Con él podemos saber si el usuario final ha abierto el email y por siguiente tener estadísticas del envío.

HTML Parser está desarrollado bajo la licencia Common Public Licence Version.

4.5. Software de apoyo

Este software es Eolica Services. Eolica Services es la plataforma de email marketing de Infoavisos. Con la herramienta podemos realizar envíos, editados en el panel de administración.

4.5.1. Eolica Services

Magic SMTP se integra con Eolica, es decir, el cliente se dará de alta en Eolica Services y se le dará permiso para usar Magic STMP. El cliente podrá habilitar un cliente de correo, con la IP de Magic SMTP y poder hacer sus envíos desde su cliente de correo.

En el panel de administración de Magic SMTP podrá ver el impacto que está generando la campaña enviada a través de Magic SMTP.

Podemos decir que el software Magic SMTP es un enchufe de campañas de Eolica Services.

4.6. Comparativa con otras tecnologías

4.6.1. Comparativa con otros lenguajes

C++

C++ es un lenguaje parecido a Java, pero este no dispone de tantas herramientas como el utilizado. Con esto me refiero a que si se hubiese desarrollado en C++, tendría que haber desarrollado desde cero un servidor SMTP, un parseador de emails y de archivos HTML.

Cuando estuve haciendo investigación sobre librerías que me pudiesen ayudar, no encontré muchas en C++. La comunidad de desarrolladores de Java parece más grande que la de C++.

Phyton

Phyton es un lenguaje en auge. Muchas aplicaciones se están desarrollando en el mismo y parece que da unos resultados inmejorables. Opté por Java gracias a que en el proceso de investigación obtuve mejores resultados con librerías Java.

En un futuro no descartaría usar este lenguaje de programación ya que su comunidad parece muy activa y parece menos complejo que Java.

4.6.2. Comparativa con otros servidores SMTP

Esta sección es en honor a Sendmail, Qmail, Postfix... todos aquellos servidores SMTP que estuve barajando para el desarrollo de Magic SMTP.

Sendmail

Sendmail es el rey de los servidores SMTP, lo podemos encontrar en la gran mayoría de distribuciones Linux.

Sinceramente en un principio quería usarlo para tener la escucha que necesitaba en el puerto 25. Pero existe un problema con Sendmail, su configuración. Si queremos un pequeño Sendmail que haga de relay o que simplemente haga de entrada de nuestros emails, es verdad, es sencillo trabajar con él. Pero cuando queremos conectarlo a un software propio, aquí se complica.

Fue descartado por su compleja configuración, fue un intento frustrado. En un futuro puede ser una de las mejoras de Magic SMTP.

Qmail

Qmail sería un competidor de Sendmail. También se estuvo barajando la posibilidad de que fuese la puerta de entrada de los emails, pero se descartó por configuración y integración complejas.

Durante el proceso de investigación, no se encontró la forma de poder aprovechar el potencial de este software para la gestión de emails en cuanto al tratamiento de los mismos por una aplicación de terceros.

Postfix

Misma solución que Sendmail y Qmail. Se investigó la forma de integración con software propio para la gestión del contenido de los emails. Parecía bastante compleja y se optó por descartarlo.

Postfix es una gran herramienta en cuanto SMTP.

4.6.3. Comparativa con otros parsers HTML

Jericho HTML Parser

Jericho prometía ser un buen parser de HTML. Se eligió esta solución en un principio. Al iniciar el desarrollo se detectó que existía otra librería llamada HTML Parser, usada por una comunidad más activa.

Se decidió cambiar a esta otra librería en una fase inicial del proyecto.

5. DiSEÑO

El diseño de la aplicación está pensado para que en forma de API (enchufe). Explicaré cuales son las tres partes para que el sistema funcione como tal. Al ser un API, tiene trabajo interior (trabajo oscuro), para hacer que sea transparente al usuario. Este es un trabajo sacrificado pero poco vistoso. El gran logro de estas lineas de programación es que el usuario vea que es un enchufe totalmente transparente.

5.1 Comparativa con otras tecnologías

En el mercado actual no existe una aplicación tan especifica para un producto como el nuestro. Eolica necesitaba un desarrollo a medida. Este desarrollo, enchufe o API, es comparable a un Sendmail o a un Qmail, pero con el valor añadido de poder generar datos adicionales de los emails que pasan por la pasarela. La intención de generar datos es la posterior explotación de los mismos. Queremos saber el ratio de apertura y que links son clicados en las campañas.

Todo este proceso queremos que sea transparente para el usuario, es decir, que el pueda conectar su cliente de correo (ej: Outlook) a un servidor SMTP, el cual será mágico para él. Una vez conecte este cliente al servidor SMTP (Magic SMTP) verá como podrá ver toda una serie de estadísticas generadas por su envío, ya sea a una o a miles de personas.

Comparativa con esta característica, algún producto hay, pero es una aplicación muy enfocada al producto ya existente (Eolica). No es un producto que se pueda comparar con otros productos del mercado, está hecho muy a medida.

Eolica es un servicio web que lo vamos a usar para ver las estadísticas de envío. En la actualidad ya genera estadísticas para envíos generados por la propia aplicación. En este caso, vamos a aprovechar todo su potencial para mostrar datos estadísticos de los envíos realizados a través de Magic SMTP.

Explicaremos esta tanto la conexión inicial al puerto 25 como la conexión final a Eolica para hacer el envío y mostrar los datos.

5.2 Actores

El sistema contiene diferentes actores, según el uso del software Magic SMTP. En concreto tenemos dos usos. El usuario administrador y el usuario empresa.

5.2.1 Usuario administrador

El usuario administrador es el encargado de dar de alta las empresas que pueden enviar emails a través de Magic SMTP. Al ser un servicio publicado en el puerto 25 de una IP de internet, es posible que muchos spammers intenten enviar sus emails de spam por nuestra IP. El administrador al dar de alta cuentas de empresa, evita que se pueda colar cualquier usuario en el software. El software con la configuración adecuada es capaz de discriminar las entradas SMTP por el puerto 25.
El usuario administrador también es encargado de monitorizar el servicio y ver que todo funciona dentro de la normalidad.
Debe informar de cualquier anormalidad a los usuarios empresa, esto también incluye el hecho de enviar emails a cuentas las cuales son erróneas o susceptibles a desprestigiar la IP de envío.
Por último debe documentar la conexión a Magic SMTP desde diferentes clientes de envío estándares (Outlook, Gmail, Yahoo, Apple mail...),

5.2.2 Usuario empresa

Es el usuario estrella de Magic SMTP. Es el encargado de hacer las campañas de Email Marketing para que Magic las saque por sus respectivos canales de envío.
El usuario empresa es el encargado de configurar su cliente de correo en base a una documentación previa facilitada por el usuario administrador. Una vez configurado su cliente de correo, es el encargado de generar tráfico para Magic SMTP.
Definimos tráfico como cualquier tipo de envío el cual quiera saber que ratio de apertura tiene, que links han sido clicados o si simplemente a llegado o no ha llegado a destino.
El concepto de API hace que sea mágico. Queremos que el usuario empresa envié como siempre, pero con estadísticas.

5.3 Casos de uso

Un caso de uso es la explicación a la acción que se realiza a través del software Magic SMTP.

Los usuarios que participan en el sistema se denominan actores. Intentaremos mostrar cómo interactúan estos actores con el software para ver los diferentes casos de uso.

5.3.1 Casos de uso del Actor usuario administrador

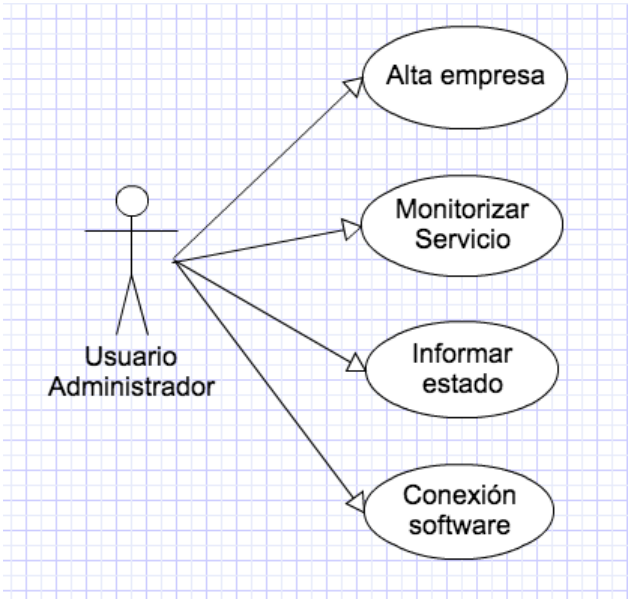

Ilustración 17: Casos de uso administrador

Caso de uso: Alta empresa

Descripción: Permite al usuario empresa hacer envíos a través del software.

Actores: Todos.

Precondición: Que el usuario no esté dado de alta en el software.

Flujo principal:

1. El usuario administrador entra en la base de datos.
2. Inserta el registro en la tabla adecuada.

Postcondición: El usuario empresa pueda enviar emails de forma correcta.

Caso de uso: Monitorizar el servicio

Descripción: El usuario administrador debe garantizar el servicio las 24 horas.

Actores: Usuario administrador.

Precondición: Conocer software de monitorización (Nagios)

Flujo principal:

1. Instalación software servidor monitorización.
2. Instalación software cliente monitorización.
3. Monitorizar todos los puntos conflictivos de Magic SMTP.

Postcondición: El administrador debe saber al instante cualquier problema de la aplicación.

Caso de uso: Informar estado aplicación

Descripción: El usuario empresa debe saber si su herramienta de envío está disponible al 100%

Actores: Todos

Precondición: Ninguna

Flujo principal:

1. Software monitorización avisa de una anomalía.
2. Usuario administrador notifica de esta anomalía al usuario empresa por el canal establecido.

Postcondición: Servicio de calidad para el usuario empresa.

Caso de uso: Conexión a Magic SMTP

Descripción: El usuario empresa tiene que conectarse al servidor SMTP para hacer sus envíos.

Actores: Todos.

Precondición: Tener instalado software cliente de envío de emails.

Flujo principal:

1. Información necesaria sobre datos de conexión al servidor SMTP.
2. Configuración del servidor SMTP en el software cliente.
3. Prueba de conexión mediante envío de un email.

Postcondición: Envío correcto, viendo que llega a destino.

5.3.2 Casos de uso del Actor usuario empresa

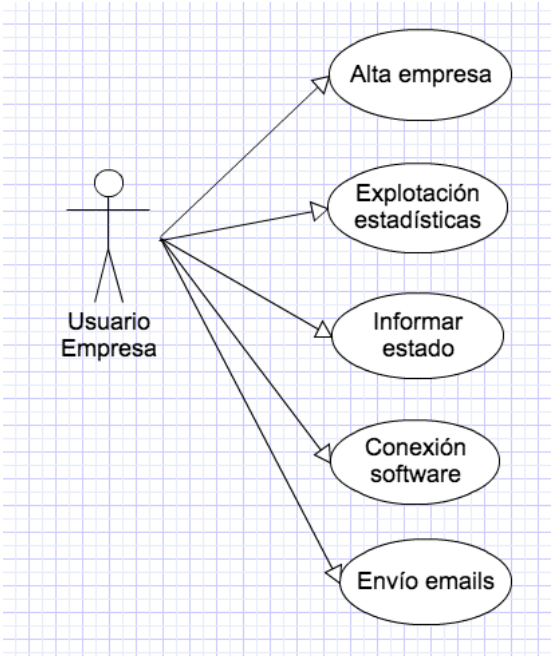

Ilustración 18: Casos de uso empresa

Caso de uso: Envío de email

Descripción: El usuario empresa puede enviar un email.

Actores: Usuario empresa.

Precondición: Conexión correcta del cliente de correo.

Flujo principal:

1. Preparación correo electrónico en software cliente del usuario empresa.
2. Envío a través de la IP de Magic SMTP.

Postcondición: El envío del correo electrónico debe llegar correctamente al destino

Caso de uso: Explotación de estadísticas

Descripción: El usuario empresa quiere saber toda la información posible sobre sus envíos.

Actores: Usuario empresa.

Precondición: Envío de email a través de Magic SMTP.

Flujo principal:

1. Envío de email a través de la aplicación.

2. Entrar en cuenta de Eolica.
3. Visualizar todas las estadísticas de nuestro interés.

Postcondición: Usuario empresa tiene toda la información deseada sobre su envío.

5.4 Arquitectura básica del sistema

El proyecto ha sido desarrollado bajo la siguiente arquitectura:

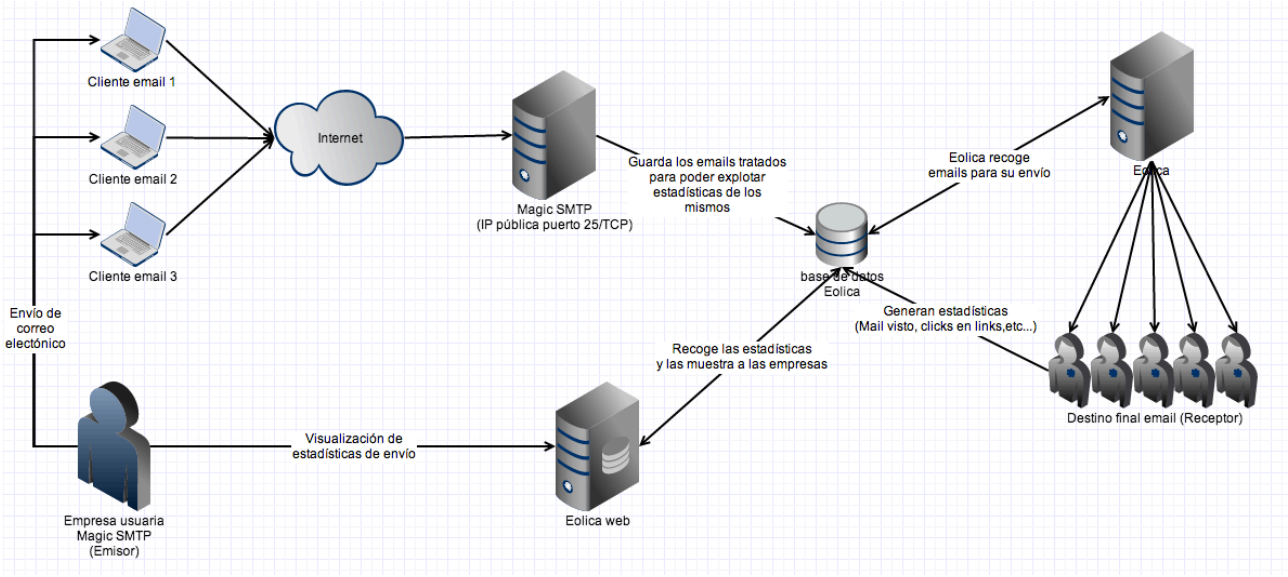

Ilustración 19: Arquitectura Magic SMTP

5.5 Componentes

5.5.1 SMTP entrada (Magic)

Es el componente de la aplicación el cual se encarga de recibir todas las entradas de email de los clientes de correo de los usuarios finales.

Podemos entender el SMTP de entrada como un proceso o servicio de un sistema operativo el cual se queda permanentemente activo en el puerto conocido 25 (puerto del protocolo SMTP) para la recepción de todas las peticiones que hagan todos los clientes de correo o sistemas de envío.

Éste SMTP de entrada es capaz, en primer lugar, de discriminar todas aquellas peticiones que no son válidas para el sistema como posibles usuarios que intentan enviar y no son autorizados por el mismo SMTP. Este proceso se lleva a cabo en la instrucción MAIL FROM del protocolo SMTP. En esta instrucción el servidor SMTP (Magic) consulta en la base de datos si el cliente está dado de alta y autorizado para poder enviar emails a través del mismo.

En segundo lugar hace un análisis de los destinatarios a los cuales será enviado el email y los guarda en el componente Eolica. Este análisis se realiza en la instrucción RCPT TO del protocolo SMTP

En tercer lugar se encarga de analizar el contenido que viene en la instrucción DATA del ya nombrado protocolo SMTP. Este análisis consiste en extraer la información necesaria

del contenido del email para poder extraer información en el siguiente paso.

Por último se encarga de recoger el contenido HTML que le da en el tercer paso y inserta el código necesario en el cuerpo del envío para el posterior análisis estadístico del mismo. Inserta etiquetas en los links del cuerpo del email y inserta un pixel de 1x1 para cuando el email sea abierto, pueda marcar en algún lugar que este ha sido abierto.

5.5.2 Eolica Services

Eolica Services (http://www.eolicaservices.com) es una herramienta de email marketing que ya tenía infoAvisos. Esta herramienta sirve para realizar campañas de email marketing y poder analizar el resultado de las mismas. Este módulo dentro de Magic SMTP se encarga de mostrar lo mágico que puede llegar a ser.

En primer lugar, Eolica recibe el email ya trabajado del servidor SMTP antes nombrado. Éste contenido es tratado por Eolica de forma que genera un email personalizado para cada destinatario. En este email ya vienen links personalizados para que cuando se haga click se recoja estadística del mismo y también tiene el seguimiento de apertura.

Una vez generados estos emails individuales para cada destinatario, Eolica es capaz de conectarse al servidor SMTP que requiera el dominio del destinatario. Esto lo hace consultando los registros MX del dominio del destinatario. Según estos registros MX, Eolica sabe a qué IP o dominio debe conectarse por SMTP y entregarlo a destino.

Eolica ya sabía enviar emails pero necesitaba de una recepción adecuada para que pudiese enviar los mismos. Ha existido un trabajo de conexión entre los dos módulos para que se pueda crear el "Magic SMTP".

Para finalizar, Eolica ya tiene un panel implementado el cual permite ver las estadísticas de envío para un usuario concreto, apertura de links, recepción del email en el SMTP de destino y lectura del email.

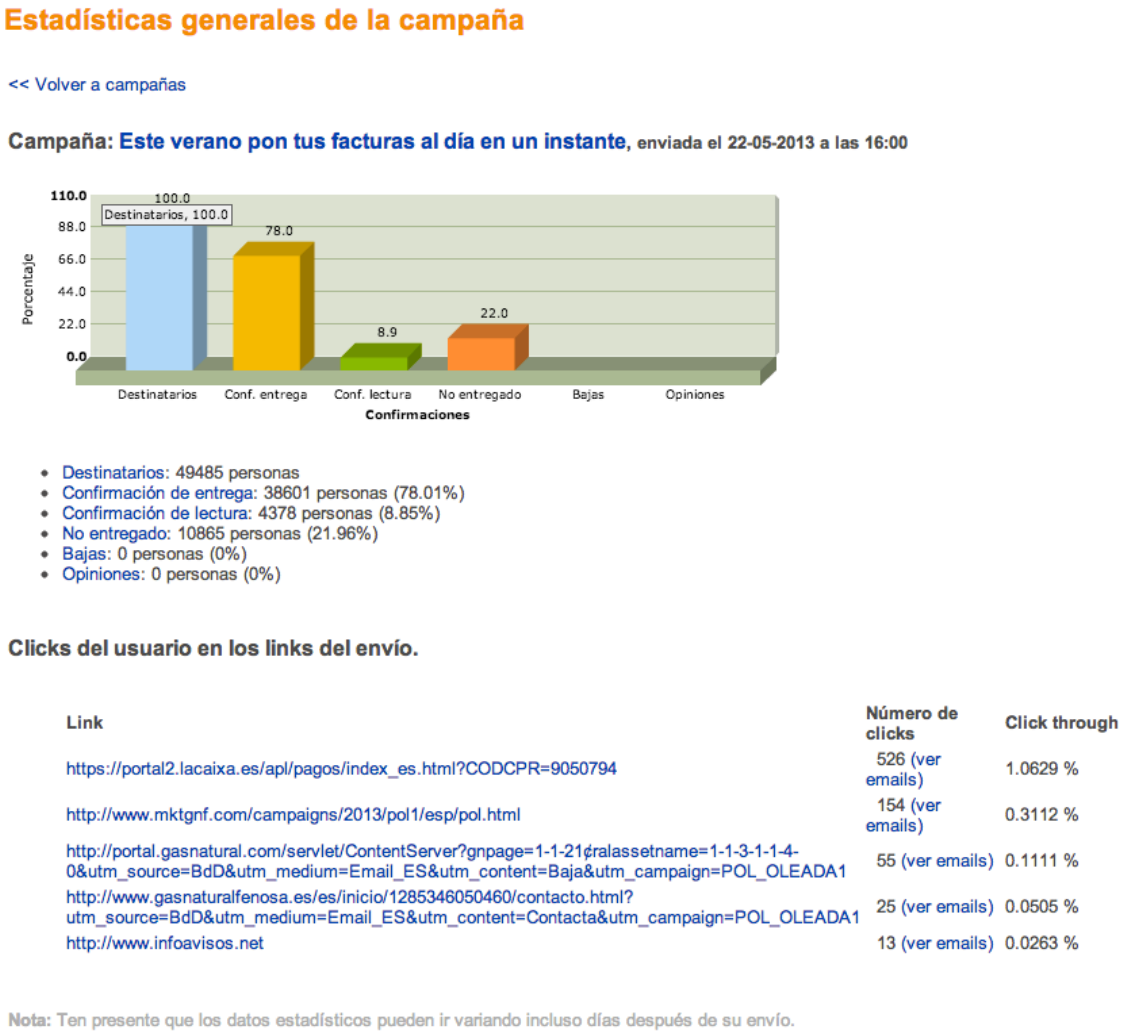

Ilustración 20: Panel de estadísticas de Eolica

5.6 Base de datos

Magic SMTP funciona a través del modelo relacional que ofrece MySQL. Mediante una base de datos con tablas relacionadas a través de “keys” podemos obtener toda la información necesaria para que el sistema sea dinámico y multiusuario.

Se han tenido que generar registros nuevos en tablas ya existentes de Eolica, Magic SMTP ha aprovechado el desarrollo de Eolica para su integración.

A continuación se explicarán las tablas que tienen protagonismo en cuanto a la integración del servidor SMTP a la herramienta Eolica.

5.6.1 Tabla tenvio

En la tabla tenvio particIpan tanto el componente SMTP como Eolica. Es la encargada de recibir todas las campañas, con todas sus características, destinatarios, asunto, cuerpo del email...

Ilustración 21: Tabla tenvios

Como campos destacados, tenemos el IDEnvio como primary key y IDEmpresa que es una foreign key asociada a la primary key de la tabla tempresa, el id de la empresa. El campo enviado, también es importante ya que es el que marca el proceso cuando lo recoge para enviarlo y lo marca como enviado.

5.6.2 Tabla tempresa

En la tabla tempresa tenemos guardadas todas las empresas dadas de alta en el sistema. Es aquí donde discriminamos que empresas pueden envíar a través de Magic SMTP. Si un cliente no está dado de alta en esta tabla, éste no puede enviar.

Ilustración 22: Tabla tempresa

Como campos destacados tenemos el IDEmpresa que es la primary key de la tabla identifica a cada usuario empresa. También destacar el campo email, ya que es el campo a validar en los envíos a través de Magic SMTP. Existen campos que no usa Magic SMTP pero existen en otras prestaciones de Eolica.

5.6.3 Tabla sms_outbox

La tabla sms_outbox es la encargada de guardar cada linea de email que se envía. El generador de emails de Eolica inserta el email personalizado en esta tabla para cada email que viene en el RCPT TO. Un sistema de desencolado de Eolica, lo desencola y lo entrega al SMTP de destino, en función del dominio del destino.

Ilustración 23: Tabla sms_outbox

Destacar los campos telefonocc que es dónde va el email de destino. El campo se denomina así por herencia del servicio de infoAvisos, los sms. Por otro el campo body, que es el email ya generado listo para ser entregado y el campo estado que marca que se ha hecho con el email. Destacar la cantidad de índices ya que se necesita velocidad de desencolado.

5.6.4 Tabla tcontactos_envios

La tabla tcontactos_envios es la que guarda las estadísticas de los envíos, así como ratios de apertura y links clicados.

Ilustración 24: Tabla tcontactos_envios

Como campos destacados tenemos el id único de contacto, línea única de estadísitca para ese contacto de ese envío. Tenemos IDcontacto foreign key del contacto, tenemos IDEnvio foreign key del envío y IDEmpresa foreign key de la empresa que ha hecho el envío.

5.7 Desarrollo del sistema

El sistema se ha creado en base al protocolo SMTP y a la lógica de un servidor SMTP de escucha activa en un puerto TCP. En este caso en el puerto conocido 25 TCP.

Se ha implementado este servidor con ayuda de la librería SubEtha SMTP. Una vez implementado este servidor capaz de recibir correos y transformarlos, se ha realizado la conexión a Eolica.

Eolica es el sistema de infoAvisos para visualizar el resultado de las campañas de email marketing.

5.7.1 Escucha activa puerto 25 TCP

Los sistemas operativos incluyen una serie de puertos para poder conectarse con otras máquinas. Estos puertos son los llamados puertos TCP/Ip. Como su nombre indica, están incluidos en la capa de transporte del modelo OSI de interconexión de sistemas abiertos.
Una vez que sabemos que necesitamos tener un puerto activo para que otra maquina se conecte con la nuestra, entendemos que Magic STMP debe estar enchufado a un puerto determinado para poder recibir las comunicaciones de otras máquinas, en este caso los clientes de correo de las empresas.

Magic SMTP recibe emails, entonces, debemos saber que el puerto que vamos a necesitar es el puerto conocido TCP 25. ¿Por qué este puerto? Existe una serie de puertos estándares o puertos conocidos que se identifica por los protocolos que trabajan sobre ellos. En internet el puerto 25 está asociado al protocolo SMTP (Send mail transfer protocol). Como Magic SMTP recibirá emails tenemos que trabajar con SMTP y por consiguiente debemos implementar nuestro servidor SMTP en el puerto 25.

Para realizar esta escucha activa, la librería utilizada nos ayuda con ello y se mantiene a la escucha de conexiones SMTP en el puerto. Una vez recibida la petición SMTP, la librería contesta y se establece la comunicación entre las máquinas.

La librería deja el email al resto de la aplicación para que lo trabaje.

5.7.2 Verificación de usuario en el MAIL FROM

Cómo se ha descrito antes, Magic SMTP trabaja con el protocolo SMTP. Este protocolo tiene una serie de instrucciones para que las máquinas se entiendan. Se podría resumir de la siguiente manera:

- Cuando un cliente establece una conexión con el servidor SMTP, espera a que éste envíe un mensaje **“220 Service ready”** o **“421 Service non available”**

- Se envía un **HELO** desde el cliente. Con ello el servidor se identifica. Esto puede usarse para comprobar si se conectó con el servidor SMTP correcto.

- El cliente comienza la transacción del correo con la orden **MAIL FROM**. Como argumento de esta orden se puede pasar la dirección de correo al que el servidor notificará cualquier fallo en el envío del correo (Por ejemplo, **MAIL FROM:<fuente@host0>**). Luego si el servidor comprueba que el origen es válido, el servidor responde **"250 OK"**.
- Ya le hemos dicho al servidor que queremos mandar un correo, ahora hay que comunicarle a quien. La orden para esto es **RCPT TO:<destino@host>**. Se pueden mandar tantas órdenes RCPT como destinatarios del correo queramos. Por cada destinatario, el servidor contestará **"250 OK"** o bien **"550 No such user here"**, si no encuentra al destinatario.
- Una vez enviados todos los RCPT, el cliente envía una orden **DATA** para indicar que a continuación se envían los contenidos del mensaje. El servidor responde **"354 Start mail input, end with <CRLF>.<CRLF>"** Esto indica al cliente como ha de notificar el fin del mensaje.
- Ahora el cliente envía el cuerpo del mensaje, línea a línea. Una vez finalizado, se termina con un <CRLF>.<CRLF> (la última línea será un punto), a lo que el servidor contestará **"250 OK"**, o un mensaje de error apropiado.
- Tras el envío, el cliente, si no tiene que enviar más correos, con la orden **QUIT** corta la conexión. También puede usar la orden **TURN**, con lo que el cliente pasa a ser el servidor, y el servidor se convierte en cliente. Finalmente, si tiene más mensajes que enviar, repite el proceso hasta completarlos.

extraído de wikipedia http://es.wikipedia.org/wiki/Simple_Mail_Transfer_Protocol

Una vez entendido el protocolo, Magic SMTP realiza una función imprescindible dentro de la instrucción MAIL FROM.

La empresa que quiere enviar un email a través de Magic SMTP, tiene que hacerlo con email remitente(MAIL FROM), entonces aquí se identifica este usuario. Si este no tiene permiso para enviar a través de la aplicación, el servidor SMTP devuelve un error 500 definitivo al cliente.

Validando usuarios, sabemos asociar las estadísticas de envío y hace que el sistema sea seguro contra clientes que buscan continuamente servidores SMTP abiertos para realizar envíos piratas.

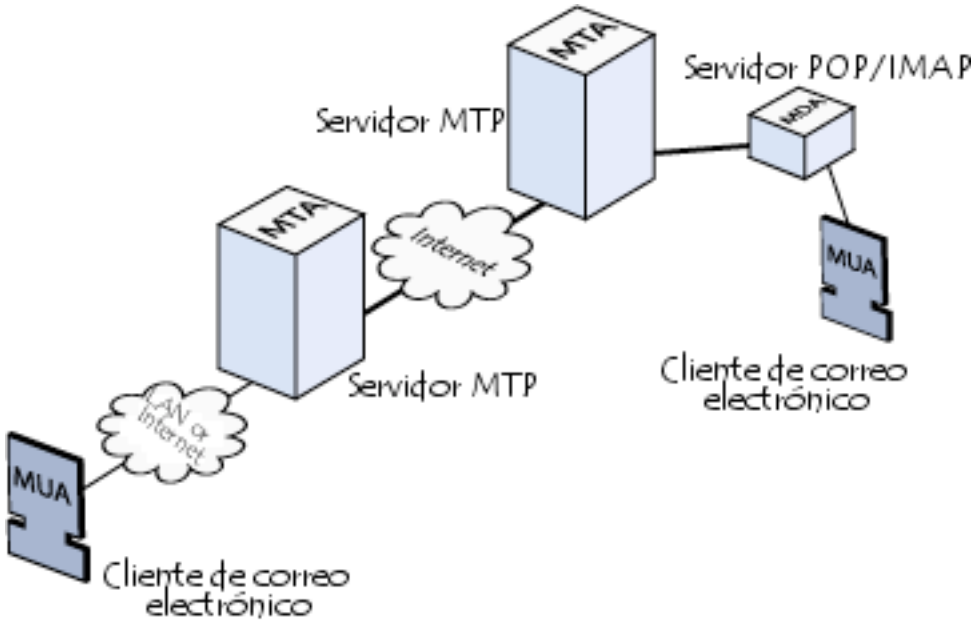

Ilustración 25: Cómo funciona SMTP

5.7.3 Parser de email

Una vez validado el usuario y pasado el protocolo SMTP, el email tiene que ser tratado para obtener el cuerpo del contenido y descartar valores de cabeceras y otros que no necesitamos. Aquí entra Javamail. Esta librería parsea el email y lo convierte en una estructura Java o objeto el cual podemos ir pidiendo los valores que queramos con simples llamadas get.

Con un simple getMessage() se obtiene el mensaje del email, o un getSubject() y se obtiene el asunto del email. Estos valores nos van a servir para la integración con Eolica y la explotación de las estadísticas de envío de ese email o campaña. Campaña se define como el mismo email enviado a un número determinado de emails.

A continuación podemos ver un ejemplo de email, así se entiende que trabajo realiza Javamail cuando le pedimos solo el asunto o el mensaje del email.

```
Delivered-To: ppedrosa@infoavisos.com
Received: by 10.227.212.13 with SMTP id gq13csp106425wbb;
        Tue, 28 May 2013 00:26:55 -0700 (PDT)
X-Received: by 10.66.172.233 with SMTP id bf9mr32472139pac.139.1369726014370;
        Tue, 28 May 2013 00:26:54 -0700 (PDT)
Return-Path: <daguayo@infoavisos.com>
Received: from mail-pa0-f42.google.com (mail-pa0-f42.google.com [209.85.220.42])
        by mx.google.com with ESMTPS id yu7si22874268pac.44.2013.05.28.00.26.53
        for <ppedrosa@infoavisos.com>
        (version=TLSv1 cIpher=ECDHE-RSA-RC4-SHA bits=128/128);
        Tue, 28 May 2013 00:26:54 -0700 (PDT)
Received-SPF: pass (google.com: domain of daguayo@infoavisos.com designates 209.85.220.42 as permitted sender)
client-Ip=209.85.220.42;
Authentication-Results: mx.google.com;
       spf=pass (google.com: domain of daguayo@infoavisos.com designates 209.85.220.42 as permitted sender)
smtp.mail=daguayo@infoavisos.com
Received: by mail-pa0-f42.google.com with SMTP id bj1so2376259pad.1
        for <ppedrosa@infoavisos.com>; Tue, 28 May 2013 00:26:53 -0700 (PDT)
X-Google-DKIM-Signature: v=1; a=rsa-sha256; c=relaxed/relaxed;
        d=google.com; s=20120113;
        h=mime-version:from:date:message-id:subject:to:content-type
         :x-gm-message-state;
        bh=KVrBAEoIFV4EcGECRfEnL3BIgaHqLHTGo3rDl1L33vM=;
        b=N6g6F+lHJZfm3PqpgiIpv6b5XPDnkeE4NecgDwDdQ9FLuq+zedFFzHwcKAwfsy2bG2
         A1u500Lzb8SUykhz/OqUVIp+UEMTSK2FEHaCLvzjLmIDnasDcHuJGcoKUvR4/6zzv3WR
         V9YmmJpQuwNnI3sBQnoLTxHfREngrndmFHbgbEO9YHTJcIe3A0LyiRWmiYghB+MnCP92
         rmGyxOFhBq9VMzECW8PUhi1ilmnV1xTR7duKDnrahNfuB2NSeMVnq++/0QIdWvHUAShE
         xgbbR6K8ZgOBaWO3MireuLbKaFl35QBno2NSmtrFCB2DuQH9SQ/2qDnvBI5vTCUnICfX
         /8/g==
X-Received: by 10.66.19.234 with SMTP id i10mr33239886pae.152.1369726013766;
 Tue, 28 May 2013 00:26:53 -0700 (PDT)
MIME-Version: 1.0
Received: by 10.70.4.37 with HTTP; Tue, 28 May 2013 00:26:33 -0700 (PDT)
From: Daniel Aguayo <daguayo@infoavisos.com>
Date: Tue, 28 May 2013 09:26:33 +0200
Message-ID: <CAN+j1mqrNX4KU8HOuFLiZOCn7LBeqFYXAXiYKKwFThxFhojYng@mail.Gmail.com>
Subject: Resum: Manteniment sms_outbox Eolica
To: Pablo Pedrosa <ppedrosa@infoavisos.com>, Oriol Garcia <oriolg@tecnics.net>
Content-Type: multIpart/alternative; boundary=bcaec520f5017d8d7504ddc2306f
X-Gm-Message-State: ALoCoQkDV0cEZhj78/odEYhUu6lZB/3X8W84rRohPwoGVBS7OOcGqiBpGQUvaW80mmx33k96MQSb

--bcaec520f5017d8d7504ddc2306f
Content-Type: text/plain; charset=ISO-8859-1
Content-Transfer-Encoding: quoted-printable

A sms_outbox d'Eolica hi ha 18 milions de registres. Modifiquem l'scrIpt de
manteniment perqu=E8 esborri:

- Estado =3D qualsevol, si la campanya es m=E9s vella de 6 mesos (aquest =
=E9s el
temps que deixem les estad=EDstiques y les campanyes)
- Estado =3D 101, esborrem

No esborrem:
- Estado <=3D100 y campanya < 6 mesos, est=E0 en proc=E9s encara

Salut.-

--bcaec520f5017d8d7504ddc2306f
Content-Type: text/html; charset=ISO-8859-1
Content-Transfer-Encoding: quoted-printable

<div dir=3D"ltr">A sms_outbox d&#39;Eolica hi ha 18 milions de registres. M=
odifiquem l&#39;scrIpt de manteniment perqu=E8 esborri:<div><br></div><div>=
- Estado =3D qualsevol, si la campanya es m=E9s vella de 6 mesos (aquest =
=E9s el temps que deixem les estad=EDstiques y les campanyes)</div>

<div>- Estado =3D 101, esborrem</div><div><br></div><div style>No esborrem:=
</div><div>- Estado &lt;=3D100 y campanya &lt; 6 mesos, est=E0 en proc=E9s =
encara<br clear=3D"all"><div><div dir=3D"ltr"><br></div></div><div style>Sa=
lut.-</div>

</div></div>

--bcaec520f5017d8d7504ddc2306f--
```

Ilustración 26: Ejemplo de un email

5.7.4 Parser de HTML

Una vez ya tenemos el contenido, el cuerpo, del email se tiene que parsear el mismo. Este contenido se parsea cuando es HTML. Aquí entra en juego la librería HTML parser. Esta hace algo parecido a la librería Javamail. Transforma el contenido HTML en una estructura o objetp el cual le puedes pedir cualquier valor de cada tag HTML que se encuentra.

Gracias a HTML parser, se puede identificar cada link y redirigirlo a Eolica para que este lo marque como visitado y redirigir al link original mediante un parámetro get dentro de la url añadida por Magic SMTP.

También inserta un píxel en transparente antes del final de body, para que cuando el usuario final abra el email, esta imagen haga una marca en Eolica como que ha sido abierto y así mostrarlo en las estadísticas de ese envío.

A continuación se muestra un fragmento del email de la ilustración anterior dónde HTML parser realiza su función:

```
Content-Type: text/html; charset=ISO-8859-1
Content-Transfer-Encoding: quoted-printable

<div dir=3D"ltr">A sms_outbox d&#39;Eolica hi ha 18 milions de registres. M=
odifiquem l&#39;scrIpt de manteniment perqu=E8 esborri:<div><br></div><div>=
- Estado =3D qualsevol, si la campanya es m=E9s vella de 6 mesos (aquest =
=E9s el temps que deixem les estad=EDstiques y les campanyes)</div>

<div>- Estado =3D 101, esborrem</div><div><br></div><div style>No esborrem:=
</div><div>- Estado &lt;=3D100 y campanya &lt; 6 mesos, est=E0 en proc=E9s =
encara<br clear=3D"all"><div><div dir=3D"ltr"><br></div></div><div style>Sa=
lut.-</div>

</div></div>
```

Ilustración 27: Ejemplo de código html dentro de un email

5.7.5 Modificación links y inserción de píxel para estadísticas

Cómo se ha dicho antes gracias a HTML parser podemos identificar todos los tags HTML del contenido de la campaña. Una vez identificados, primero insertamos antes del tag </body> (fin de body), un píxel transparente con una url apuntando al servidor de Eolica para marcar como leído ese email.

En segundo lugar modificamos todos los links que haya en el HTML, para poner un link de Eolica especial

Este link es un pequeño scrlpt, que marca como visitado el link al que se ha clicado y redirige mediante un redirect de php a la página deseada. Veamos un ejemplo.

link original: http://www.uab.es
link en email (transformado): http://www.EolicaServices.com/link.php?url=http://www.uab.es

Al hacer click en el segundo link, el php marca como visitado y redirige a la url que venga como parámetro get.

5.7.6 Integración Eolica para envío de la campaña y seguimiento de estadísticas

Una vez trabajado el email, éste se debe insertar en la tabla de Eolica tenvio. Aquí la comunicación entre el servidor SMTP y la base de datos Eolica es mediante una conexión JDBC por base de datos. Se podrían haber implementado webservices para la comunicación entre las plataformas, pero se prefirió ir por base de datos ya que son dos plataformas muy especificas y siempre irán unidas.

Insertamos en la tabla tenvio, generando un id de campaña nuevo y asociando a una empresa, todos los contactos a los cuales se va a enviar, el contenido del email transformado y la hora de envío, que será la actual ya que no permite programar.

Eolica cogerá este envío y lo enviará mediante su sistema de envío. Si se ha realizado correctamente la inserción de la campaña se podrán ver las estadísticas de envío en el panel de la empresa que lo ha enviado.

6. CODIFICACIÓN Y PRUEBAS

En esta sección se tratará de explicar todo lo relacionado con formas de implementar el código, estándares, protocolos, etc. Las pruebas como su nombre indica, sirven para tener una fiabilidad más elevada de nuestro software. Se han realizado varios tIpos de prueba para ver la fiabilidad del servicio.

6.1 Estilo de codificación

Se describe la serie de estándares y protocolos seguidos a la hora de implementar el software. Esto se realiza para que futuros desarrolladores que tengan que actualizar o mantener la aplicación, sepan leer el código lo más rápido posible y así familiarizarse rápidamente con él.

Los nombres de las variables suelen ser nombres explícitos de su función. Es decir si una variable hace de contador, a esta la llamaremos contador y así con el resto. Debe tener un nombre el cual defina su valor.

Los nombres de los métodos de las clases deben definir la acción que realizan, es decir, si tenemos una clase Javamail y queremos que nos de el contenido de un email, llamaremos al método getContent. Así con el resto de métodos, definición de la acción que realizan.

Las clases deben ser nombres que identifiquen a la estructura de datos que representan, por ejemplo, la clase BasicSMTPServer.java identifica al servidor SMTP la aplicación arranca en el puerto 25.

La identación es fundamental, el simple hecho de que el código esté bien identado hace que su lectura sea muy fluida. Incluso a algunos entornos de desarrollo les facilitamos la vida con una buena identación. Básico para la lectura de futuros desarrolladores.

Los comentarios en el código también son básicos. Nos encontramos que como programadores realizamos todo tipo de funciones y funcionalidades las cuales solo nosotros sabemos lo que hacen. Si añadimos un comentario en un bloque de código que realiza una función, un futuro desarrollador no tiene que analizar ese bloque, sabe que se realiza la acción que se comenta en el comentario.

Dentro de la aplicación Magic SMTP se ha intentado seguir estos consejos para su fácil lectura. Se ha pensado de forma que otra gente del equipo de infoAvisos pueda seguir implementando funcionalidades a la aplicación.

Sobre las pruebas, se han realizado varias para saber de la fiabilidad del servicio. Una de ellas ha sido enviar emails que gran capacidad. Otra ha sido la de tener un número alto de destinatarios. La más crítica ha sido la de que la aplicación esté viva en el puerto 25 durante un tiempo infinito en un entorno ideal.

7. CONCLUSIONES

Como programador de la aplicación estoy satisfecho con el resultado de Magic SMTP. Se han tenido que descartar funcionalidades ya que no se disponía de más tiempo. El servidor Magic SMTP es un servidor estable, capaz de recibir campañas de clientes de correo electrónico y ofrecer estadísticas de lectura y apertura de links.

Gracias al proyecto he podido aprender como funciona un servidor SMTP en profundidad, también aprender que existen librerías que hacen las cosas por ti. Con esto me refiero a no reinventar la rueda, si no aprovechar que alguien ya lo ha hecho para sacarle el máximo rendimiento con tu idea. Librerías como Javamail o HTML Parser son imprescindibles para el proyecto y si las tuviésemos que desarrollar desde cero serían un proyecto entero.

Ha habido un periodo de investigación y desarrollo importante, esto no queda reflejado en la memoria del proyecto, bueno sí, se puede observar que existe un conocimiento de las herramientas utilizadas, pero es un tiempo difícil de plasmar en este escrito.

En conclusión, se esta satisfecho con el trabajo realizado y será de gran utilidad todo el esfuerzo de investigación para futuros desarrollos.

7.1. Desviaciones

El proyecto es muy ambicioso y se quería realizar un servidor SMTP capaz de interpretar como una misma campaña envíos con el mismo asunto en horas diferentes. Este sistema de asociación inteligente de campañas no se ha realizado por falta de tiempo.

Se puede decir que la falta de tiempo ha sido el principal causante de que el proyecto sufriera desviaciones. Esto ocurre por tener una previsión demasiado optimista del tiempo que se invertirá en el proyecto.

7.2 Ampliaciones

La versión actual de Magic SMTP permite hacer envíos de campañas mediante clientes de correo convencionales. Cada envío que se realiza es una campaña nueva, esto puede ser incomodo a la hora de hacer múltiples envíos en una misma campaña. En realidad, en esta versión 1, esto no se puede hacer. Otra función que está prevista para una versión posterior y que ahora mismo no existe, es la de añadir plantillas HTML a envíos de email que no contengan un código HTML.

La siguiente versión se ha pensado montar un sistema inteligente que reconozca de alguna forma (posible identificador de mismo asunto con una diferencia de tiempo determinada) para poder agrupar en una misma campaña diferentes envíos de un mismo cliente para una misma campaña. Por otro lado se quiere tener una versión estable de envío de texto plano con una plantilla predeterminada por el cliente para que se pueda trabajar el código HTML y así poder hacer seguimiento de la campaña en posteriores estadísticas.

A nivel de sistemas, se quiere implementar un sistema de balanceo de servidores, para balancear la carga de los envíos. Es decir, que una máquina que sepa encolar envíos, reciba todas las peticiones a Magic SMTP y este distribuya entre diferentes instancias de Magic SMTP el tráfico, para poder absorber con mayor velocidad todas las peticiones y no provocar atascos en los clientes de envío.

8. BIBLIOGRAFÍA

En el desarrollo del proyecto se ha obtenido información de las siguientes fuentes. El proceso de investigación se ha basado en tratar de encontrar las mejores soluciones a los problemas presentados a través de internet y sus buscadores, principalmente Google.

- http://es.wikIpedia.org/wiki/Simple_Mail_Transfer_Protocol -> Portal wikIpedia referencia para muchas definiciones y más en el sector informático. Explicación detallada del protocolo SMTP. Es el protocolo utilizado por la aplicación para realizar sus funciones. (Última consulta Junio 2013)

- http://es.wikIpedia.org/wiki/Transmission_Control_Protocol -> WikIpedia. Aquí podemos encontrar toda la información referente al protocolo TCP de la capa de internet. Es un protocolo mas bajo que SMTP, pero igual de importante ya que es el protocolo que indica cómo se entienden dos máquinas en internet. (Última consulta Junio 2013)

- https://code.google.com/p/subethasmtp/ -> Librería para recepción de emails. Aquí se ha podido descargar la librería y poder hacer uso de la misma. Tiene ejemplos básicos de poner en marcha un servidor SMTP en un puerto TCP y de como manipular los datos recibidos en el mismo. (Última consulta Junio 2013)

- http://www.oracle.com/technetwork/Java/Javamail/index.html -> Librería Javamail. Librería útil para tener toda la información sobre los emails que recibe el servidor SMTP. Aquí se ha podido descargar la librería y encontrado una api con todas las funciones las cuales nos permiten un uso ágil de la información. (Última consulta Junio 2013)

- http://htmlparser.sourceforge.net/ -> Librería de parseo HTML. Aquí se ha encontrado una librería para identificar todos los tags en un código HTML y poder manipularlo de forma adecuada. Existen ejemplos básicos de uso. (Última consulta Junio 2013)

- http://www.oracle.com/technetwork/Java/Javase/documentation/index-jsp-135444.html -> Documentación Java para el uso ágil de su tecnología. Aquí podemos encontrar todas aquellas funciones y funcionalidad de esta tecnología de programación. (Última consulta Junio 2013)

- Normativa de proyectos: https://neptu.uab.es/PROJFC/normativapfc20102011infor.pdf (Última consulta Junio 2013)
- Javadoc: http://docs.oracle.com/Javase/6/docs/api/ (Última consulta Junio 2013)
- Documentación GIT: http://git-scm.com/documentation -> Documentación necesaria para poder guardar versiones de código de forma segura y poder volver atrás en estas versiones. (Última consulta Junio 2013)
- http://www.agpd.es/portalwebAGPD/index-ides-idphp.php -> LOPD (Ley Orgánica de Protección de Datos). (Última consulta Junio 2013)
- http://es.wikIpedia.org/wiki/Sender_Policy_Framework -> SPF. Es una protección contra la falsificación de direcciones en el envío de correo electrónico. (Última consulta Junio 2013)

9. GLOSARIO

ANDROID: Android es un sistema operativo basado en Linux, diseñado principalmente para dispositivos móviles.

ANTISPAM: El antispam es lo que se conoce como método para prevenir el "correo basura".

APACHE LICENSE: La licencia Apache (Apache License o Apache Software License para versiones anteriores a 2.0) es una licencia de software libre creada por la Apache Software Foundation (ASF).

API: Interfaz de programación de aplicaciones (IPA) o API (del inglés Application Programming Interface) es el conjunto de funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción. Son usadas generalmente en las bibliotecas.

APPLE MAIL: Cliente de correo de Apple para recibir y enviar emails.

BODY html: Contenido principal de un código html.

BODY mail: Contenido principal de un archivo de email.

BREAKPOINT: Punto de pausa en la ejecución de un programa para identificar comportamientos anómalos en el desarrollo de software.

BUG: Error conocido del software.

COLDFUSION: Es un servidor de aplicaciones y un lenguaje de programación usado para desarrollar aplicaciones de internet, generalmente sitios web generados dinámicamente.

COMMON DEVELOPMENT AND DISTRIBUTION LICENSE: licencia de código abierto (OSI) y libre, producida por Sun Microsystems, basada en la Mozilla Public License o MPL, versión 1.1.

COMPILADOR JIT: En informática, la compilación en tiempo de ejecución (también conocida por sus siglas inglesas, JIT, just-in-time), también conocida como traducción dinámica, es una técnica para mejorar el rendimiento de sistemas de programación que compilan a bytecode, consistente en traducir el bytecode a código máquina nativo en tiempo de ejecución.

CSS: Las hojas de estilo en cascada o (Cascading Style Sheets, o sus siglas CSS) hacen referencia a un lenguaje de hojas de estilos usado para describir la presentación semántica (el aspecto y formato) de un documento escrito en lenguaje de marcas.

CSS3: Es la versión más actual de la tecnología CSS.

DATA: Dentro del protocolo SMTP, DATA sirve para indicar el comienzo del mensaje, éste finalizará cuando haya una línea únicamente con un punto.

DEBUG: Depuración de programas es el proceso de identificar y corregir errores de programación.

DIRECCIONES IP: Una dirección IP es una etiqueta numérica que identifica, de manera lógica y jerárquica, a un interfaz (elemento de comunicación/conexión) de un dispositivo (habitualmente una computadora) dentro de una red que utilice el protocolo IP (Internet Protocol), que corresponde al nivel de red del Modelo OSI.

DRIVER: Un controlador de dispositivo (llamado normalmente controlador, o, en inglés, driver) es un programa informático que permite al sistema operativo interactuar con un periférico, haciendo una abstracción del hardware y proporcionando una interfaz -posiblemente estandarizada- para usarlo.

ECLIPSE PUBLIC LICENCE: La Licencia Pública Eclipse (EPL) es una licencia de software de código abierto utilizada por la Fundación Eclipse para su software. Sustituye a la Licencia Publica Común (CPL) y elimina ciertas condiciones relativas a los litigios sobre patentes.

EMAIL MARKETING: Es la forma de comunicar campañas publicitarias a través de un envío masivo de email.

EOLICA: Eolica es un software que proporciona la gestión de campañas de email marketing.

ERROR x SMTP: El protocolo SMTP tiene varios códigos de error que identifican distintos errores.

FRAMEWORK: La palabra inglesa "framework" (marco de trabajo) define, en términos generales, un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular que sirve como referencia, para enfrentar y resolver nuevos problemas de índole similar.

GETTERS: Dentro de la programación orientada a objetos, estos deben estar encapsulados y para acceder a su información, se hace a través de los getters. Son funciones de un objeto que devuelven valores del mismo.

GMAIL: Gmail, llamado en otros lugares Google Mail (Austria y antes en Alemania -hasta 2012- y Reino Unido -hasta 2009-) por problemas legales. Es un servicio de correo electrónico con posibilidades POP3 e IMAP gratuito.

GPL: La Licencia Pública General de GNU o más conocida por su nombre en inglés GNU General Public License (o simplemente sus siglas del inglés GNU GPL) es la licencia más ampliamente usada en el mundo del software y garantiza a los usuarios finales (personas, organizaciones, compañías) la libertad de usar, estudiar, compartir (copiar) y modificar el software.

HOSTING: El alojamiento web (en web hosting) es el servicio que provee a los de un sistema para poder almacenar información, imágenes, vídeo, o cualquier contenido accesible vía web.

HTML: HTML, siglas de HyperText Markup Language («lenguaje de marcado hipertextual»), hace referencia al lenguaje de marcado predominante para la elaboración de páginas web que se utiliza para describir y traducir la estructura y la información en forma de texto, así como para complementar el texto con objetos tales como imágenes. El HTML se escribe en forma de «etiquetas», rodeadas por corchetes angulares(<,>).

HTML5: Ultima versión de HTML

JAVA: El lenguaje de programación Java fue originalmente desarrollado por James Gosling de Sun Microsystems (la cual fue adquirida por la compañía Oracle) y publicado en el 1995 como un componente fundamental de la plataforma Java de Sun Microsystems.

JAVADOC: Javadoc es una utilidad de Oracle para la generación de documentación de APIs en formato HTML a partir de código fuente Java. Javadoc es el estándar de la industria para documentar clases de Java. La mayoría de los IDEs los generan automáticamente.

JAVASCRIPT: JavaScript es un lenguaje de programación interpretado, dialecto del estándar ECMAScript. Se define como orientado a objetos, basado en prototipos, impreatico, débilmente tipado y dinámico.

JDBC: Java Database Connectivity, más conocida por sus siglas JDBC, es una API que permite la ejecución de operaciones sobre bases de datos desde el lenguaje de programación Java, independientemente del sistema operativo donde se ejecute o de la base de datos a la cual se accede, utilizando el dialecto SQL del modelo de base de datos que se utilice.

JDK: Java Development Kit o (JDK), es un software que provee herramientas de desarrollo para la creación de programas en Java. Puede instalarse en una computadora local o en una unidad de red.

JQUERY: jQuery es una biblioteca de JavaScript, creada inicialmente por John Resig, que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones (FLV) y agregar interacción con la técnica AJAX a páginas web.

JRE: En su forma más simple, el entorno en tiempo de ejecución de Java está conformado por una Máquina Virtual de Java o JVM, un conjunto de bibliotecas Java y otros componentes necesarios para que una aplicación escrita en lenguaje Java pueda ser ejecutada.

JVM: Una máquina virtual Java (en inglés Java Virtual Machine, JVM) es una máquina virtual de proceso nativo, es decir, ejecutable en una plataforma específica, capaz de

interpretar y ejecutar instrucciones expresadas en un código binario especial (el bytecode JAVA), el cual es generado por el compilador del lenguaje Java.

LICENCE: Una licencia de software es un contrato entre el licenciante (autor/titular de los derechos de explotación/distribuidor) y el licenciatario del programa informático (usuario consumidor /usuario profesional o empresa), para utilizar el software cumpliendo una serie de términos y condiciones establecidas dentro de sus cláusulas.

LINK: Comunmente llamados hiperenlaces, también llamados hipervínculos, son parte fundamental de la arquitectura de la World Wide Web, pero el concepto no se limita al HTML o a la Web. Casi cualquier medio electrónico puede emplear alguna forma de hiperenlace.

LINUX: GNU/Linux es uno de los términos empleados para referirse a la combinación del núcleo o kernel libre similar a Unix denominado Linux con el sistema GNU. Su desarrollo es uno de los ejemplos más prominentes de software libre; todo su código fuente puede ser utilizado, modificado y redistribuido libremente por cualquiera bajo los términos de la GPL (Licencia Pública General de GNU, en inglés: General Public License) y otra serie de licencias libres.

LOPD: La Ley Orgánica 15/1999 de 13 de diciembre de Protección de Datos de Carácter Personal, (LOPD), es una Ley Orgánica Española que tiene por objeto garantizar y proteger, en lo que concierne al tratamiento de los datos personales, las libertades públicas y los derechos fundamentales de las personas físicas, y especialmente de su honor, intimidad y privacidad personal y familiar.

MAIL FROM: En el protocolo SMTP para indicar quien envía el mensaje.

MARKETING: El término marketing o mercadotecnia tiene diversas definiciones. Según Philip Kotler (considerado por algunos padre del marketing moderno ó guru de la mercadotecnia),1 es «el proceso social y administrativo por el que los grupos e individuos satisfacen sus necesidades al crear e intercambiar bienes y servicios».

MARKETING ONLINE: También llamado Marketing 2.0 pretende ser una adaptación de la filosofía de la web 2.0 al marketing, se refiere a la transformación del marketing como resultado del efecto de las redes en Internet.

MODELO OSI: El modelo de interconexión de sistemas abiertos (ISO/IEC 7498-1), también llamado OSI (en inglés, Open System Interconnection) es el modelo de red descriptivo, que fue creado por la Organización Internacional para la Estandarización (ISO) en el año 1984.

MONITORIZAR: La monitorización es un proceso que se supone inmerso dentro de la llamada función ejecutiva o sistema ejecutivo. Hace referencia a la supervisión necesaria para la ejecución del plan de acción establecido en la planificación de las acciones, conductas o pensamientos encaminados al logro de una meta.

MTA: El Mail Transfer Agent es un servidor de correo es una aplicación de red ubicada en un servidor en internet, cuya función es enviar y recibir emails.
El MTA tiene varias formas de comunicarse con otros servidores de correo:

1. Recibe los mensajes desde otro MTA. Actúa como "servidor" de otros servidores.
2. Envía los mensajes hacia otro MTA. Actúa como un "cliente" de otros servidores.
3. Actúa como intermediario entre un "Mail Submision Agent" y otro MTA.

MySQL: MySQL es un sistema de gestion de bases de datos relacional, multihilo y multiusuario con más de seis millones de instalaciones.

NOSQL: En informática, NoSQL (a veces llamado "no sólo SQL") es una amplia clase de sistemas de gestión de bases de datos que difieren del modelo clásico del sistema de gestión de bases de datos relacionales (RDBMS) en aspectos importantes, el más destacado que no usan SQL como el principal lenguaje de consultas.

OBJECTIVE-C: Objective-C es un lenguaje de programación orientado a objetos creado como un superconjunto de C para que implementase un modelo de objetos parecido al de Smaltalk. Originalmente fue creado por Brad Cox y la corporación StepStone en 1980.

ONLINE: El término en línea (online) hace referencia a un estado de conectividad, frente al término fuera de línea (offline) que indica un estado de desconexión.

ORM: El mapeo objeto-relacional (más conocido por su nombre en inglés, Object-Relational mapping, o sus siglas O/RM, ORM, y O/R mapping) es una técnica de programación para convertir datos entre el sistema de tipos utilizado en un lenguaje de programación orientado a objetos y el utilizado en una base de datos relacional, utilizando un motor de persistencia.

OUTLOOK: Microsoft Outlook es un programa de organización ofimática y cliente de correo electrónico de Microsoft, y forma parte de la suite Microsoft Office.

PARSER: Un analizador sintáctico (en inglés parser) es una de las partes de un compilador que transforma su entrada en un árbol de derivación.

PHP: PHP es un lenguaje de programación de uso general de código del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico.

PLUGIN: Un complemento es una aplicación que se relaciona con otra para aportarle una función nueva y generalmente muy específica.

POSTFIX: Postfix es un servidor de correo de software libre / código abierto, un programa informático para el enrutamiento y envío de correo electrónico, creado con la intención de que sea una alternativa más rápida, fácil de administrar y segura al ampliamente utilizado Sendmail.

PRIMARY KEY: En el diseño de bases de datos relacionales, se llama clave primaria a un campo o a una combinación de campos que identifica de forma única a cada fila de una tabla. Una clave primaria comprende de esta manera una columna o conjunto de columnas. No puede haber dos filas en una tabla que tengan la misma clave primaria.

PROTOCOLO: Es un conjunto de acciones, métodos, y la observancia de determinadas reglas convencionales, que constituye un procedimiento planificado y estructurado convencional, destinado a estandarizar un comportamiento ya sea humano u artificial ante una situación específica.

PUERTO TCP/25: SMTP Simple Mail Transfer Protocol (Protocolo Simple de Transferencia de Correo).

PÍXEL: Un píxel o pixel (México) , plural píxeles, (acrónimo del inglés picture element, 'elemento de imagen') es la menor unidad homogénea en color que forma parte de una imagen digital, ya sea esta una fotografía, un fotograma de vídeo o un gráfico.

QMAIL: qmail es un servidor de correo electrónico (SMTP) hecho para Unix.

RCPT TO: En SMTP para indicar el destinatario del mensaje.

REGISTRO MX: Un registro MX (del inglés Mail eXchange record, en español "registro de intercambio de correo") es un tipo de registro, un recurso DNS que especifica cómo debe ser encaminado un correo electrónico en internet. Los registros MX apuntan a los servidores a los cuales envían un correo electrónico, y a cuál de ellos debería ser enviado en primer lugar, por prioridad.

RELAY: El Protocolo Sencillo para Relevo de Correo (SMTP, por sus siglas en inglés), le permite enviar mensajes de correo electrónico mediante su cuenta de correo usando su servicio existente de correo electrónico.

SENDMAIL: Sendmail es un popular "agente de transporte de correo" (MTA - Mail Transport Agent) en Internet, cuya tarea consiste en "encaminar" los mensajes correos de forma que estos lleguen a su destino.

SMS: El servicio de mensajes cortos o SMS (Short Message Service) es un servicio disponible en los teléfonos móviles que permite el envío de mensajes cortos (también conocidos como mensajes de texto, o más coloquialmente, textos) entre teléfonos móviles que inventó un finlandés, Matti Makkonen junto al GMS en 1985.

SMTP: El Simple Mail Transfer Protocol (SMTP) (Protocolo para la transferencia simple de correo electrónico), es un protocolo de la capa de aplicación. Protocolo de red basado en texto, utilizado para el intercambio de mensajes de correo electrónico entre computadoras u otros dispositivos (PDA, teléfonos móviles, etc.). Está definido en el RFC 2821 y es un estándar oficial de Internet.

SPAM: Se llama spam, correo basura o mensaje basura a los mensajes no solicitados, no deseados o de remitente no conocido (correo anónimo), habitualmente de tipo publicitario, generalmente enviados en grandes cantidades (incluso masivas) que perjudican de alguna o varias maneras al receptor.

SPAMMER: Persona o proceso que hace envíos masivos de spam.

SPF: SPF (Convenio de Remitentes, del inglés Sender Policy Framework) es una protección contra la falsificación de direcciones en el envío de correo electrónico. Identifica, a través de los registros de nombres de dominio (DNS), a los servidores de correo SMTP autorizados para el transporte de los mensajes.

SQL: El lenguaje de consulta estructurado o SQL (por sus siglas en inglés structured query language) es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones en ellas.

TAG: Etiqueta dentro del código html.

TCP: Transmission Control Protocol (en español Protocolo de Control de Transmisión) o TCP, es uno de los protocolos fundamentales en Internet. Fue creado entre los años 1973 y 1974 por Vint Cerf y Robert Kahn.

TRACKING: Seguimiento de estadísticas de un envío.

WEBSERVICE: Un servicio web (en inglés, Web services) es una tecnología que utiliza un conjunto de protocolos y estándares que sirven para intercambiar datos entre aplicaciones.

YAHOO: Yahoo! Inc. es una empresa global de medios con sede en Estados Unidos, posee un portal de Internet, un directorio web y una serie de servicios, incluido el popular correo electrónico Yahoo!.

Pablo Pedrosa Manchón