



## INGENIERIA INFORMATICA

Curso Académico 2010/2011

Proyecto Fin de Carrera

Gestión de grandes cantidades de correo en listas  
de distribución: herramienta colaborativa web 2.0  
y experiencias.

Autor : César Fernández Gago

Tutor : Gregorio Robles Martínez



# Agradecimientos

*A Gregorio, por todas las tardes imaginativas que  
hemos pasado, y la paciencia que le hizo falta,  
A Alo, por sus enseñanzas, y porque él me metió  
en este mundillo.*

*A mi familia,  
Para Marta.*



# Índice general

<b>1. Introducción</b>	<b>9</b>
1.1. Estudios anteriores . . . . .	10
<b>2. Entorno tecnológico</b>	<b>13</b>
2.1. Web 2.0 . . . . .	13
2.1.1. Herramientas . . . . .	15
2.1.2. El usuario como actor principal . . . . .	17
2.2. Desarrollo de aplicaciones con software libre . . . . .	18
2.2.1. Algoritmos de votación . . . . .	19
2.3. Python . . . . .	20
2.4. Mailman . . . . .	20
2.4.1. RSS . . . . .	21
2.5. Django . . . . .	22
2.6. Thunderbird . . . . .	23
<b>3. Objetivos</b>	<b>25</b>
3.1. Desarrollo de software libre 2.0 . . . . .	25
3.2. Objetivos a lograr . . . . .	27
3.2.1. Estudio de impacto . . . . .	28
<b>4. Diseño e Implementación</b>	<b>31</b>
4.1. Diseño . . . . .	31
4.2. Mailman . . . . .	31
4.3. Aplicación web Vote-MM . . . . .	38
4.3.1. Creación del proyecto . . . . .	38
4.4. Plugin Thunderbird . . . . .	42
4.5. Instalación . . . . .	42

<b>5. Estudio</b>	<b>49</b>
5.1. Publicación . . . . .	49
5.2. Encuesta . . . . .	53
5.3. Conclusiones . . . . .	56
5.4. Futuro . . . . .	60
5.5. Lecciones aprendidas . . . . .	61
<b>A. Manual de usuario</b>	<b>63</b>
A.1. Interfaz web Vote-MM . . . . .	63
A.1.1. Flujo de trabajo . . . . .	64
A.1.2. Configuración de listas de correo . . . . .	64
A.2. Plugin Thunderbird . . . . .	66
A.2.1. Instalación . . . . .	68
A.2.2. Flujo de trabajo . . . . .	68

# Índice de figuras

2.1. Plataforma para la Web 2.0 . . . . .	14
2.2. Cambio Web 1.0 a Web 2.0 . . . . .	15
4.1. Arquitectura de Vote-MM . . . . .	31
4.2. Arquitectura de Mailman . . . . .	32
4.3. Configuración pie de página . . . . .	36
4.4. Clases BBDD Vote-MM . . . . .	40
4.5. Clases Vote-MM . . . . .	41
5.1. Opciones SourceForge . . . . .	50
5.2. Visitas en los días de promoción del proyecto . . . . .	51
5.3. robots.txt . . . . .	51
5.4. Opciones SourceForge . . . . .	52
5.5. Número de visitas totales del proyecto . . . . .	53
5.6. Orígenes de las visitas. Mayo-Agosto . . . . .	54
5.7. Trove categorization . . . . .	54
5.8. Freshmeat . . . . .	55
5.9. Número de visitas en Freshmeat . . . . .	55
5.10. Evolución de votos diarios . . . . .	56
5.11. Encuesta realizada a los miembros de FANS . . . . .	57
5.12. Encuesta realizada a los miembros de FANS . . . . .	58
5.13. Evolución de correos enviados a la lista . . . . .	58
5.14. Descargas de archivos desde Sourceforge . . . . .	59
A.1. Pagina principal Vote-MM . . . . .	63
A.2. Correo de la lista . . . . .	64
A.3. Correos más votados en el día . . . . .	65
A.4. Lista de correos elegidos como principales . . . . .	65

A.5. Configuración: Configuración parámetros . . . . .	66
A.6. Actualizamos el RSS . . . . .	67
A.7. Mensaje de actualización . . . . .	67
A.8. Cliente de correo thunderbird con plugin Vote-MM instalado . . . . .	67
A.9. Menú Add-on de Thunderbird . . . . .	68
A.10. Opciones de configuración . . . . .	69
A.11. Elegimos habilitar la columna <i>Votes</i> . . . . .	69



# Capítulo 1

## Introducción

Durante los últimos años, la evolución de los contenidos servidos por Internet ha cambiado; el contenido es ahora dinámico y ensamblado usando servidores en la intranet, codificado usando diferentes aplicaciones y ofrecido mediante servidores web con una interfaz unificada. Con cada vez más contenido y con un número exponencial de usuarios, las infraestructuras hardware/software se han tenido que adaptar cubriendo las nuevas necesidades. Pero esta revolución no ha sido solamente tecnológica, al generalizarse el uso de Internet desde diferentes medios y situaciones, el usuario ha pasado de ser un mero consumidor de datos a proporcionarlos al resto de usuarios.

El intercambio de información e ideas ha aumentado gracias a la adopción de diferentes paradigmas y mediante el uso de diferentes tecnologías. Se ha pasado de una web 1.0 estática, técnica, poco integradora y no participativa donde al usuario se le contaba a base de *clicks*, a proporcionar *feedback* continuamente al creador del contenido, promoviendo discusiones y la inclusión o mejora del contenido aportado, convirtiéndose en un actor con una importancia mejor definida en la web.

Las tecnologías desarrolladas se han convertido en una herramienta generadora de cambios estructurales en la web. Se utiliza la web como plataforma, donde se ejecutan aplicaciones dentro del navegador web. Estos cambios permiten la elaboración de interfaces de comunicación más complejas aunque más intuitivas para el personal no técnico, facilitando la integración del conocimiento aportado por el creciente número de usuarios.

Sin embargo, no todos estos métodos han sido reutilizados para mejorar las metodologías de desarrollo. Algunas herramientas de las más extendidas en el mundo del desarrollo de software libre no han evolucionado como la web 2.0 (ver punto 2.1). Este es el caso

del popular gestor de listas de correo Mailman<sup>1</sup>.

El objetivo de este proyecto de fin de carrera ha sido adaptar el gestor de listas de correo a las nuevas expectativas en la web 2.0. Las listas de correo manejan una gran cantidad de información heterogénea que no es categorizada ni etiquetada, por lo que no es fácil priorizarla u organizarla. En muchas ocasiones, la gran cantidad de tiempo perdido leyendo correos sin interés (flames, spam..) frustran al usuario haciendo que la lista pierda importancia como mecanismo de comunicación.

Este proyecto propone una serie de mejoras y parches para Mailman aprovechando las tecnologías desarrolladas en la web 2.0. Se ha añadido una nueva funcionalidad para exportar la información de las listas de correo por RSS (ver punto 2.1.1 ), se ha liberando una aplicación web que mejora la interacción de los usuarios de la lista priorizando y ordenando el contenido de las listas, añadiendo la capacidad de reorganizar cada correo desde diferentes ámbitos:

- Desde el cliente de correo, incluyendo la información necesaria en la firma del correo.
- Mediante un plugin desarrollado para thunderbird con el que visualizar la información referente a cada correo o incluso votar, utilizando el cliente de correo.
- Desde la aplicación web desarrollada, que ofrece una interfaz más cómoda para el usuario.

## 1.1. Estudios anteriores

Existen publicaciones donde se estudia la productividad de los usuarios respecto a la cantidad de correos que reciben, demostrando que el abuso en el número de correos provoca el desinterés y la pérdida de tiempo de los lectores del mismo. Una publicación de la *Open University Australia*<sup>2</sup> propone una herramienta donde cada usuario dispone de unos créditos limitados con los que poder enviar correos. Según la cantidad de créditos asociado al correo, este será más o menos importante. Se demostró que los usuarios leen más rápidamente los correos con más créditos.

---

<sup>1</sup><http://www.gnu.org/software/mailman/index.html>

<sup>2</sup><http://firstmonday.org/htbin/cgiwrap/bin/ojs/index.php/fm/article/view/2100/1963>

Este comportamiento es análogo al uso de las cabeceras de prioridad que algunos clientes de correo implementan. Sin embargo, el mal uso de estas características ha desfavorecido su uso, ya que muchos usuarios etiquetan todas sus comunicaciones con la máxima prioridad.

En este proyecto lo que se persigue es que sean el resto de usuarios y no el propio autor los que etiqueten la relevancia de los correos. Ya que los votos estarán limitados para cada usuario del sistema, el autor del correo no podrá afectar de manera relevante las votaciones.

Por otro lado, cuantas más votaciones tenga un correo, más fácilmente llegará al resto de usuarios que evaluarán de nuevo la relevancia del mismo. El sistema por tanto no tendrá en cuenta las visitas, si no los votos positivos (y los negativos).



# Capítulo 2

## Entorno tecnológico

En este capítulo se van a revisar las tecnologías involucradas en la construcción del sistema propuesto. En primer lugar se profundizará en el concepto Web 2.0, para evaluar las características principales y encontrar las tecnologías que podrán mejorar el proyecto. A continuación se revisarán las plataformas de apoyo al desarrollo en el mundo del software libre (forjas), para buscar aplicaciones que se podrían mejorar con las tecnologías revisadas.

### 2.1. Web 2.0

El término Web 2.0 nació en una sesión de 'brainstorming'<sup>1</sup> realizada con O'Reilly y MediaLive International. Cuando el vicepresidente de O'Reilly, Dale Dougherty hizo notar que lejos de haberse estrellado después del estallido de la burbuja tecnológica del 2001 la web seguía evolucionando y su importancia seguía aumentando con la inclusión de nuevas tecnologías y la aparición de nuevos sitios web. De hecho, estos sitios que sí habían sobrevivido al estallido de la burbuja tenían algo en común.

Conceptualmente podemos definirlo como una plataforma con prácticas y principios definidos donde el usuario cobra más importancia y se convierte en un actor que proporciona contenido; donde la web se convierte en una plataforma de la que un usuario puede beneficiarse para interactuar con otros usuarios. Se confía en el usuario para editar contenido, por lo que el éxito o fracaso de una aplicación depende en gran medida del número de usuarios que la mantengan. Los creadores del sitio web tendrán por tanto un especial interés en que los usuarios realicen las tareas de la forma más

---

<sup>1</sup><http://oreilly.com/web2/archive/what-is-web-20.html>

cómoda.

Las herramientas desarrolladas han facilitado el intercambio de información y de ideas. Han aparecido metodologías para priorizar y clasificar contenidos, para interconectar plataformas y flujos de información. Se han creado comunidades de usuarios y se han desarrollado nuevos métodos para poner en contacto a los miembros de esas comunidades.

Una visualización estéticamente más agradable y con contenidos organizados según los usuarios y sus preferencias marcan el antes y el después de la web 2.0.

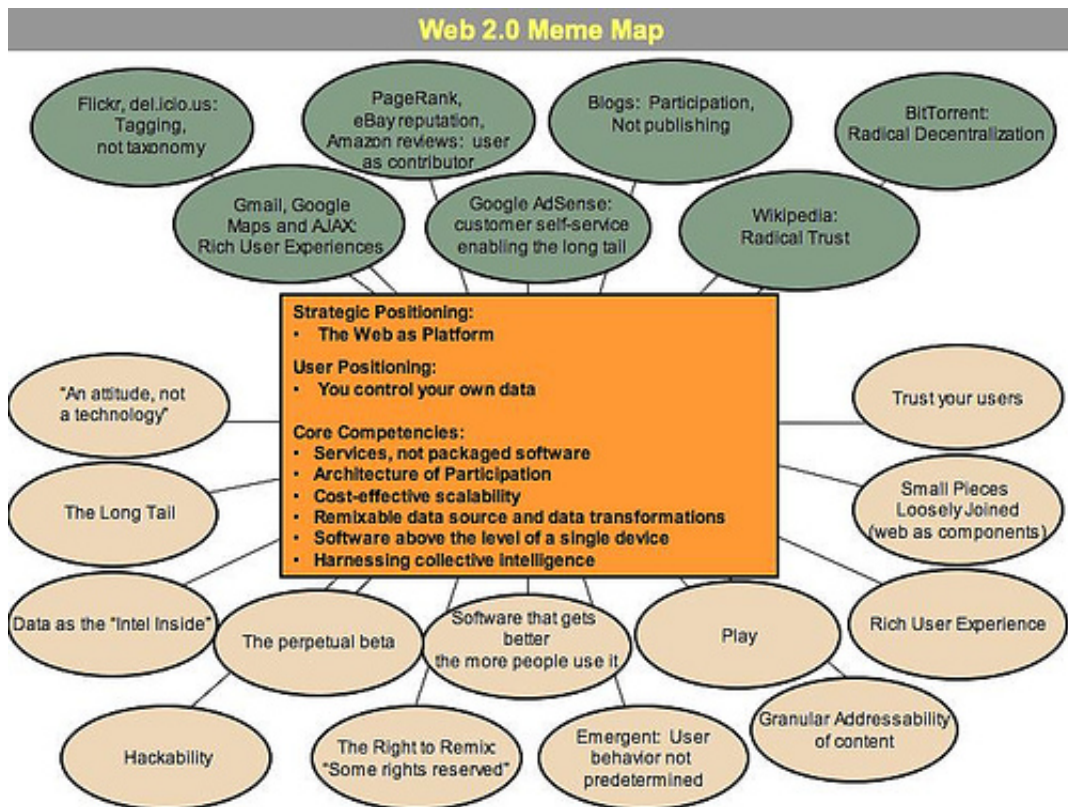


Figura 2.1: Plataforma para la Web 2.0

Se definieron algunas transiciones entre servicios y aplicaciones como muestra la figura 2.2:

Web 1.0		Web 2.0
DoubleClick	-->	Google AdSense
Ofoto	-->	Flickr
Akamai	-->	BitTorrent
mp3.com	-->	Napster
Britannica Online	-->	Wikipedia
personal websites	-->	blogging
evite	-->	upcoming.org and EVDB
domain name speculation	-->	search engine optimization
page views	-->	cost per click
screen scraping	-->	web services
publishing	-->	participation
content management systems	-->	wikis
directories (taxonomy)	-->	tagging ("folksonomy")
stickiness	-->	syndication

Figura 2.2: Cambio Web 1.0 a Web 2.0

### 2.1.1. Herramientas

Las herramientas más importantes que facilitaron esta transformación se exponen de una forma resumida a continuación.

#### Ajax

Esta tecnología permite realizar diferentes conexiones asíncronas desde un mismo documento web, permitiendo realizar varias tareas en un espacio menor de tiempo o enviando solamente la información necesaria para ejecutar una acción en concreto. El empleo de Ajax<sup>2</sup> ha facilitado la interacción con los documentos web al minimizar las latencias innatas en el protocolo HTTP. Los resultados de las acciones del usuario sobre la web son más rápidas, posibilitado un flujo más activo de información y de interacción persona-máquina.

#### RSS

RSS<sup>3</sup> es un formato utilizado para ofrecer contenido que sea modificado frecuentemente, basado en XML. El formato añade ciertos metadatos sobre el contenido sindicado (Autor, fecha de publicación, formato del contenido..), con los que se puede automatizar la tarea de compartir información entre diferentes aplicaciones.

La tecnología XML<sup>4</sup>, ofrece un estándar de modelado de datos basado en documentos de texto, capaz de almacenar estructuras complejas de información, como las nece-

<sup>2</sup>Asynchronous JavaScript and XML

<sup>3</sup>Really Simple Syndication

<sup>4</sup>Extensible Markup Language

sarias para los servicios web.

Los sindicadores RSS son aplicaciones capaces de importar datos de diferentes fuentes, facilitando al usuario la obtención de la información para su posterior consulta.

### **CSS/Xhtml/Frameworks/**

La evolución propia de HTML ha facilitado la generación de interfaces más complejas. El formato de los documentos web se ha simplificado gracias a CSS<sup>5</sup>, permitiendo separar la capa de presentación en los documentos web. Mediante DOM<sup>6</sup> es posible interactuar con objetos en HTML, XHTML o JavaScript, permitiendo ocultar elementos de la vista del usuario mientras siguen existiendo en la aplicación.

La generalización de distintos entornos de trabajo (frameworks, entornos donde se facilitan un conjunto de librerías, prácticas y conceptos) de desarrollo para la web, facilitan la refactorización del código y evitan al programador las tareas más tediosas propias del desarrollo web: programar para diferentes versiones y distintos navegadores web, además de facilitar tareas repetitivas como la gestión de usuarios, acceso a bases de datos, filtrado de datos, etc.

### **RPC/JSON**

Los *servicios via web* o *web services* son interfaces de comunicación que facilitan un protocolo de envío de mensajes y ofrecen una interfaz para su uso. El aplanamiento de tipos complejos de datos a través de la red facilita la comunicación entre actores, permitiendo entre otros la ejecución de rutinas.

### **Folksonomy**

Este término designa la organización mediante etiquetas del contenido, de una forma distribuida y generalmente realizada por los usuarios. Con este mecanismo podemos categorizar contenidos facilitando la organización y la búsqueda de información relacionada por materias.

---

<sup>5</sup>Cascading Style Sheets

<sup>6</sup>Document Object Model



### 2.1.2. El usuario como actor principal

Con las tecnologías anteriormente mencionadas, entre otras, se han desarrollado las aplicaciones que llamamos 2.0, tales como blogs, wikis, redes sociales (personales, profesionales, de fotografía..), mensajería instantánea a través de la web, galerías de fotografía y vídeo, etc.

Para comprender el funcionamiento de las aplicaciones web 2.0, podemos estudiar con más detalle el funcionamiento (además del éxito) de los cuadernos de bitácora, o blogs. En estas aplicaciones el autor publica un texto permitiendo a los usuarios comentar sobre el mismo, habilitando una retroalimentación usuario-autor beneficiosa para ambas partes.

Los textos son catalogados mediante etiquetas o *tags* que estructuran la información y facilitan al usuario la búsqueda de información relacionada. Estos blogs suelen facilitar canales RSS con esta información opcionalmente filtrada, permitiendo compartir la información relevante por otros blogs, que a su vez se enlazan creando comunidades.

Los nuevos servicios de compartición de datos también fomentan la inclusión de comentarios e información relacionada. Los datos van desde imágenes (*flickr*), vídeos (*youtube*), información (*wikipedia*), noticias (*menéame.net*, *digg.com*) hasta la propia información personal (*facebook*, *twitter*, ) o profesional (*linkedin*).

El factor común de estas aplicaciones es el usuario, que es en la mayoría de los casos quien crea el contenido. Las aplicaciones se centran en aportar las herramientas necesarias para que los usuarios aporten el contenido.

Los programadores, al disponer de un entorno tecnológico potente, el programador ha sido capaz de realizar acciones a la vez más complejas, con menos esfuerzo. Se ha facilitado la generalización del uso de Internet para personal no especializado en éstas tecnologías; este soporte a la socialización necesita de una capacidad de estructuración de la información para no saturar al lector. Los canales de información cobran especial relevancia al aumentar exponencialmente la cantidad de información disponible.

- Ahorro de tiempo, al facilitar los procesos en la web.
- Se facilita la comunicación entre procesos y personas.
- Se promueven nuevos medios de comunicación, creando contenidos.
- La búsqueda y consulta de contenidos se mejora.

- Se promueven y aprovechan los esfuerzos individuales, promoviendo la cooperación entre usuarios, con un beneficio común
- Facilita la interacción persona-ordenador

## 2.2. Desarrollo de aplicaciones con software libre

El desarrollo de software hace un uso intensivo de herramientas distribuidas. La mayoría de proyectos desarrollados por un grupo de personas crea las mismas necesidades:

- Comunicación, entre los miembros del proyecto y con los usuarios.
- Control de versiones, para el software desarrollado y para los posibles errores y soluciones aportadas.
- Documentación del producto.
- Publicitación del proyecto.

El mundo del desarrollo de software libre es dinámico y evoluciona rápidamente. Al estar constituido por personal generalmente proactivo y especialmente interesado en tecnología las herramientas de trabajo son actualizadas o sustituidas con cierta frecuencia.

La forma de trabajo suele depender del tamaño del proyecto. En la etapa inicial es frecuente el uso de forjas de software (como *sourceforge*, *launchpad*, *code.google.com*..) donde se integran distintos productos. Usualmente, una forja permite el uso de listas de correo y foros para facilitar la comunicación, algún sistema de control de versiones como SVN, CVS o bazaar, un sistema wiki para incluir documentación del proyecto y espacio web para alojar documentos y archivos. Es común que los usuarios necesiten autenticarse en el sistema, obteniendo unas credenciales que les permitirán ejecutar acciones según sus privilegios.

En los proyectos en los que la comunidad participa más activamente, se utilizan recursos propios integrados bajo una misma interfaz, siendo frecuente el uso de las mismas herramientas.

Control de versiones	CVS, SVN, bazaar..
Listas de correo	Mailman, Majordomo
Documentación	Wiki, blogs..
Gestión errores	Bugtracker, track
Servicio de noticias	CMS, blogs..
Gestión de tareas	Dotproject, trac

Cuadro 2.1: Herramientas de desarrollo de software libre

La heterogeneidad de los perfiles de desarrolladores que pueden llegar a estar interesados en un proyecto dificulta la comunicación entre los miembros. Los desarrolladores, al pertenecer a distintas zonas geográficas y con una acceso a la tecnología desigual, suelen elegir como medio de comunicación las listas de correo o foros web, en detrimento de otro métodos como las videoconferencias o IRC. Las listas de correo tienen una importancia fundamental, al servir como medio de comunicación y aprovechar sus capacidades de almacenamiento para archivar los mensajes y permitir a los usuarios futuras referencias.

### 2.2.1. Algoritmos de votación

Actualmente existen muchas aplicaciones web que utilizan distintos algoritmos para priorizar votaciones. Uno de los sitios web españoles con más tráfico es *meneame.net*, que además es software libre<sup>7</sup>. Es una página de noticias, donde cada usuario manda los enlaces con una pequeña descripción del evento y el resto de usuarios revisan la cola de envíos votando los más interesantes. Cuando una noticia llega a un nivel suficiente de votaciones, se publica en portada.

El algoritmo utilizado es bastante complejo (*promote*), ya que los votos de cada usuario asignan una prioridad según el karma<sup>8</sup> del usuario. Este coeficiente evoluciona ponderando el número de votos del usuario (teniendo en cuenta la antigüedad de la noticia), los votos que reciben las noticias enviadas (ponderadas según las metacategorías de la noticia), las puntuaciones de sus comentarios, etc. Existen incluso algoritmos para detectar cuando un grupo de usuarios se votan recíprocamente sus noticias<sup>9</sup>, para evitar que ningún grupo pueda publicar en portada más rápidamente

<sup>7</sup><http://www.meneame.net/COPYING>

<sup>8</sup><http://meneame.wikispaces.com/Karma>

<sup>9</sup><http://blog.meneame.net/2008/09/08/penalizacion-de-endogamia-de-votos/>

que otros.

Análogamente la versión inglesa original sobre la que se ha basado *meneame.net* es *digg.com*, con unos algoritmos similares en los que se evalúa la importancia de cada voto.

En la aplicación desarrollada estas aproximaciones no pueden aplicarse, ya que las votaciones son anónimas. En los objetivos (punto 3.2) se ha priorizado la facilidad y rapidez de uso para fomentar la utilización de la aplicación, evitando el proceso de registro y autenticación que iba a provocar la pérdida de posibles usuarios.

Sin embargo sí que se evalúa la antigüedad del correo votado a la hora de publicarse en portada, se ha considerado que cuanto mayor tiempo lleve publicado el correo es menos relevante, y se ha implementado, cumpliendo los objetivos, mecanismos para descartar autobombo y ataques contra la prioridad de otros envíos.

## 2.3. Python

Python es un lenguaje de programación interpretado liberado en los años 90, de propósito general y de alto nivel. Es especialmente útil para desarrollos rápidos, ya que posibilita un desarrollo sencillo y eficaz. La sintaxis que utiliza es muy sencilla, y la librería ofrece una buena cantidad de herramientas. Soporta múltiples paradigmas de programación: además de orientación a objetos, imperativa y funcional, soporta un sistema de tipos dinámico (especialmente en las últimas versiones) lo que lo convierte junto con la simplicidad de la sintaxis como un buen candidato para el desarrollo de este PFC. CPython es la librería de referencia, gestionada por la Python Software Foundation<sup>10</sup>, utilizando un modelo de desarrollo basado en la comunidad. Python posee una licencia de código abierto, la Python Software Foundation License, compatible con la GPL<sup>11</sup>.

## 2.4. Mailman

Mailman es un gestor de listas de correo. Se encarga de la gestión de suscriptores y del manejo de los correos de la lista, así como de su archivado. Aunque existen otros

---

<sup>10</sup> [http://en.wikipedia.org/wiki/Python\\_Software\\_Foundation](http://en.wikipedia.org/wiki/Python_Software_Foundation)

<sup>11</sup> General Public License

MLMS (Mailing List Management System) como Majordomo<sup>12</sup> o SmartList<sup>13</sup>, el más extendido es Mailman. Este ofrece una interfaz web para cada lista, y permite a los usuarios suscribirse/desuscribirse desde la web, además de ofrecer otras características importantes:

- Archivado de correos
- RFC934 and MIME digest delivery
- Smtplib delivery: bounces, spam..
- Dominios virtuales
- Moderación en listas, control de usuarios
- Administración via e-mail
- Administración web

Para archivar los correos de Mailman, tendremos que habilitarlo. Pipedmail es el archiver por defecto incluido en Mailman, aunque existen otras opciones.

- Lurker: Ordena los threads con una métrica interna que muestra la relevancia de cada hilo, pero los usuarios no interaccionan en el reparto de la relevancia. Ofrece búsquedas mejoradas integradas.
- Mhonarc. Es el más utilizado porque muestra una interfaz con los correos con HTML y no en texto plano. Existen varios parches para incluir RSS.
- Para realizar búsquedas tenemos HTdig (se integra con Pipedmail)
- Sympa. Proporciona archivado con HTML, RSS, y permite el almacenamiento en base de datos.

### 2.4.1. RSS

La integración de Mailman con RSS (ver sección 2.1.1) no está planeada para futuras versiones. Las opciones actuales para añadir RSS a Mailman requieren del

---

<sup>12</sup>[www.greatcircle.com/majordomo/](http://www.greatcircle.com/majordomo/)

<sup>13</sup>[www.procmail.org/era/lists.html](http://www.procmail.org/era/lists.html)

parqueo del código del paquete, i/o el uso de otro *archiver* para el almacenado de los datos.

Las opciones más comunes actualmente son:

- Script en perl que parsea el código generado desde Piplermail y genera un RSS<sup>14</sup>.
- Parche no oficial para Piplermail<sup>15</sup>. Esta opción no incluye el cuerpo del correo en el RSS, ni metadatos asociados (fecha del correo, autor..)

## 2.5. Django

Django es un framework para crear aplicaciones web, escrito en Python y basado en el principio DRY<sup>16</sup>, siguiendo el patrón MVC<sup>17</sup>, con la distinción de la plantilla en el caso de la vista, siendo el controlador el propio framework. Las características principales:

- Correlación objeto-relacional: Los modelos de datos se basan en tipos de Python, utilizando un API para la interacción con las bases de datos.
- Ofrece una interfaz para crear/editar contenido en ese modelo de datos definido automáticamente.
- La gestión de URL es dinámica y basada en expresiones regulares.
- Se utiliza un lenguaje para las plantillas potente, extensible con el que separa el diseño, el contenido y el código de la aplicación.
- Soporte para cacheo (memcached, o otros cache frameworks)
- Soporte multilinguaje.
- Incluye gran cantidad de funciones auxiliares facilitando el desarrollo de cualquier aplicación web: gestión de usuarios/permiso, generación de RSS, interacción con RPC-Json...

---

<sup>14</sup><http://taint.org/mmRSS/>

<sup>15</sup>[http://sourceforge.net/tracker/?func=detail&aid=657951&group\\_id=103&atid=300103](http://sourceforge.net/tracker/?func=detail&aid=657951&group_id=103&atid=300103)

<sup>16</sup>Dont repeat yourself, <http://c2.com/cgi/wiki?DontRepeatYourself>

<sup>17</sup>Model View Controller

El uso de este framework en el proyecto facilitará la reutilización de código y ayudará a un desarrollo más rápido.

## **2.6. Thunderbird**

La arquitectura utilizada en este cliente de correo facilita un entorno de desarrollo para la implementación de plug-ins o add-ons. La arquitectura ofrecida hace uso de diferentes tecnologías:

- Gecko: Motor del cliente de correo.
- Javascript: lenguaje en el que se implementa la lógica del plugin. Se permite el uso de librerías de la misma forma que está disponible para documentos web.
- XUL: Es la tecnología de definición de interfaces utilizada en el cliente. Mediante el uso de CSS y XML se permite la creación del layout del plugin, así como la inclusión de botones, creación de ventanas, etc.





# Capítulo 3

## Objetivos

### 3.1. Desarrollo de software libre 2.0

No todas las aplicaciones se han adaptado a la evolución de la web 2.0, y algunas de las tecnologías utilizadas hoy en día en otros sitios web podrían beneficiar a los proyectos de software libre de la interacción del usuario y de las opciones que nos brinda la web 2.0.

Una de las mejoras sería poner en valor la apreciación del usuario ante un determinado problema, permitiendo la categorización de los reportes de errores, mejoras o información. Esto posibilitaría un flujo de información entre el desarrollador y los usuarios más fluido, el cual en algunos proyectos (en función de sus dimensiones) ya no es posible debido a la gran cantidad de tiempo que es necesario para evaluar de una forma objetiva la impresión que está causando el producto en la comunidad.

Los sistemas de votaciones son muy interesantes en la medida en que facilitan de una forma rápida una visión sobre las preferencias de la audiencia. Mediante este sistema se realiza automáticamente una encuesta sobre la información publicada, pudiendo extraer fácilmente estadísticas y con ello las preferencias de los usuarios. Este principio sería aplicable tanto a tickets abiertos como errores en los programas (bugs), propuestas (enhancements) o correos (listas de correo).

Además de contribuir con documentación, los usuarios pueden organizar el contenido siguiendo los principios *folksonomy* ( ver apartado 2.1.1) en las listas de correo. Actualmente, las listas de distribución actúan como valiosos repositorios de conocimiento en los que no es sencillo encontrar información, a no ser que se utilice un buscador web. Sin embargo, poder utilizar jerarquías de etiquetas en cada correo, sería intere-

sante.

Si bien el uso del correo electrónico sigue ampliamente extendido, no es el único canal rápido de comunicación. La generalización del uso de agregadores RSS desplaza el uso del correo personal en muchos casos, ya que las ventajas del RSS son significativas.

- Filtrado de temáticas en los distintos canales de información.
- Repositorio de fuentes de información
- Canal limpio de publicidad.
- Actualización inmediata de la información.
- Disponibilidad de agregadores RSS en la web, o como aplicaciones integradas en los distintos escritorios.

Uno de los mayores problemas de las comunicaciones actuales radica en la calidad del contenido. Al tratar con un público muy numeroso, con una cultura tecnológica diversa, la facilidad en el envío de mensajes se traduce en un esfuerzo para encontrar la información realmente interesante. Este esfuerzo involucra grandes cantidades de tiempo para procesar toda la información disponible, ya que al no estar categorizada no tenemos criterios objetivos para realizar filtrados sobre ella.

Con este proyecto se pretenden mejorar todos los aspectos mencionados. Un sistema de votaciones en los correos podría priorizar la información de una forma sencilla, además con esa información podemos categorizar los correos en función del impacto que han tenido en la comunidad.

Con el fin de aumentar los canales de información por los que la información estará disponible se creará un RSS con la información de cada correo, que será actualizado permanentemente. De esta forma, podemos ofrecer listas de correo en otras aplicaciones tipo lectores RSS (para aplicaciones de cada escritorio, o para la web).

Para facilitar al usuario la acción de votar, se incluirán enlaces en cada correo con 2 links, para votar consecuentemente con la importancia (a juicio del usuario) del correo. Además, para hacer más atractiva la aplicación, se creará un plugin para el lector de correo Thunderbird donde poder visualizar los votos anteriores, y votar los nuevos correos.

Se creará un web service donde se posibilitará la recogida de información asociada a cada correo de la lista, las votaciones actuales, etc, además de posibilitar votaciones a través de este nuevo canal, para facilitar la integración con futuros productos.

La cantidad de votos recogidos depende de la experiencia del usuario con el sistema y de lo que el sistema pueda aportar al usuario.

## **3.2. Objetivos a lograr**

Las necesidades que se han identificado podemos resumirlas de la siguiente forma:

1. Organización de la información disponible, de modo que no dependa del autor de la noticia.
2. Facilitar el acceso a la información:
3. Distintos medios (desde el cliente de correo, navegador web, lector RSS..)
4. Optimizar el tiempo dedicado a la tarea de la lectura, obteniendo los mejores resultados.
5. Facilitar el acceso a la información
6. Priorizar la información
7. Utilizar los recursos disponibles. Si un usuario decide dedicar más tiempo, y quiere aportar más información, gestionarla para que otros usuarios puedan beneficiarse
8. Utilización de las características de la web 2.0 anteriormente expuestas para los entornos de desarrollo:
9. Aplicación en Mailman, para las listas de correo.

Como requisitos deseables en la aplicación, se proponen:

- La usabilidad y rapidez de uso en la aplicación web. Como objetivo se fija que la aplicación debe ser capaz de aportar información valiosa a la ya disponible por otros medios, por lo que si el usuario va a utilizar el servicio debería hacerlo como mínimo con el mismo esfuerzo.

- La prioridad consiste en obtener el máximo número de votos, o conseguir la mayor información para discriminar los correos importantes de los menos importantes.
  - Procesos intermedios como la validación en la aplicación, implican que el usuario debe aportar más tiempo del que quizá esté dispuesto a emplear. Por este motivo se descarta la obligación a validarse en la aplicación, evitando también el proceso de registro aunque sea automatizado. También se descartarán captchas<sup>1</sup> y cualquier otro proceso que aumente el tiempo del proceso de validación de un voto.
- La aplicación seguirá la estética marcada por las aplicaciones más utilizadas de la web 2.0. Por tanto, la interfaz mantendrá la usabilidad como prerrequisito fundamental y la facilidad y la rapidez de uso como objetivos fundamentales.

Las páginas más importantes serán las que incluyen la lista de correos, ordenados por hilos decrecientes, con los correos de cada hilo por orden cronológico, incluyendo la lista de votos para visualizar rápidamente cuales son los hilos más relevantes de la lista.

La otra sección incluirá un histórico con los correos más votados del día. Con un parámetro configurable indicaremos el número de correos a incluir, de forma que siempre se incluyan un número mínimo de correos diarios para que esta sección sea interesante para los usuarios.

También se podrán visualizar los correos votados según la fecha, ordenados según su número de votos, pudiendo estudiar con esta sección en qué momento se accede más a las funcionalidades del proyecto.

### **3.2.1. Estudio de impacto**

Se realizará un pequeño estudio para poder evaluar los objetivos de la herramienta. Para ello, la herramienta se pondrá a disposición de una lista de correo que actualmente dispone de 50 usuarios del entorno de la Universidad Rey Juan Carlos, en concreto una lista genérica del grupo de investigación Libresoft.

---

<sup>1</sup>Proceso por el que se pide alguna información que solamente un humano sea capaz de proporcionar, generalmente reconocimiento de texto con la consiguiente validación

Se definirá un formulario para los usuarios, donde podremos recoger y evaluar las impresiones de su utilización.



# Capítulo 4

## Diseño e Implementación

### 4.1. Diseño

La arquitectura del sistema consta de una aplicación web que recibirá la información de Mailman y facilitará una interfaz para las votaciones. Los correos que Mailman envía a la lista de usuarios irán modificados con los enlaces para las votaciones. Se ofrece también un servicio web mediante el cual se facilita la integración con productos de terceros, así como del plugin para el cliente de correo Thunderbird, que facilita la votación y la visualización de las votaciones.



Figura 4.1: Arquitectura de Vote-MM

A continuación, se detallan las modificaciones necesarias y el diseño de los componentes a desarrollar para la aplicación.

### 4.2. Mailman

La arquitectura de Mailman está basada en la clase principal MailList. Esta hereda de múltiples componentes específicos diseñados para realizar pequeñas tareas como el envío de correos o el archivado. En el gráfico 4.2 se detallan las clases principales

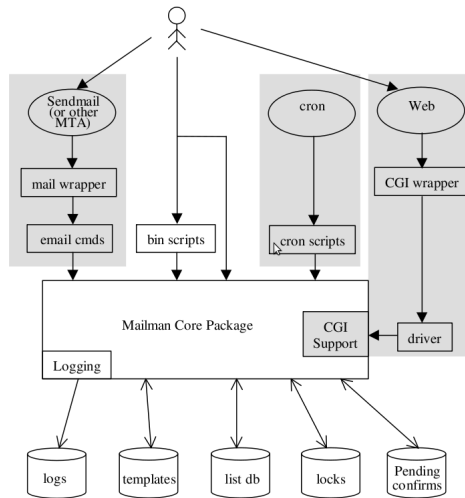


Figura 4.2: Arquitectura de Mailman

La arquitectura actual de Mailman 2.1.X no provee de ningún componente donde todos los datos del correo en trámite estén disponibles. Utilizaremos el archiver por defecto, Pimpermail (un módulo que en principio no pertenecía a Mailman, aunque con el tiempo ha pasado a formar parte del proyecto) para generar el RSS, ya que en ésta clase están disponibles los datos necesarios. Al utilizar el archiver nos aseguramos de no interferir en ningún proceso interno de Mailman, o de insertar/modificar datos que hagan que la compatibilidad del parche para futuras versiones de Mailman se vea comprometida.

La clase Archiver ofrece una interfaz para los archivadores como Pimpermail<sup>1</sup>. Estos archivadores se encargan de generar y almacenar una copia de cada correo en HTML, y de reconstruir los índices y listados (por threads, meses, etc) cada vez que se envía un correo a la lista.

El componente de RSS utilizará la lista que Mailman mantiene en caché con los últimos N correos para generar el XML. En la interfaz de Archiver (HyperArch) podemos utilizar métodos para acceder a esta base de datos interna.

```

mailitems = []
try:
    date, msgid = self.database.dateIndex.first()
    mailitems.append(msgid)
  
```

<sup>1</sup>Desarrollado por Andrew Kuchling, <http://starship.skyport.net/crew/amk/maintained/pimpermail.html>



title	Asunto del mensaje.
link	Enlace a la copia de Pippmail
author	Propietario del mensaje
description	Cuerpo del mensaje
guid	Necesario el estándar RSS
pubDate	Fecha de publicación
thread	Cadena para indentar correctamente el correo en las discusiones
msgid	Identificador único del mensaje

Cuadro 4.1: Componentes RSS

```

except KeyError:
    pass

while 1:
    try:
        date, msgid = self.database.dateIndex.next()
        mailitems.append(msgid)
        mailitems = mailitems[-mm_cfg.RSS_NUM_MAILLS:]
    except KeyError:
        break

```

Desde la versión de Mailman 2.1.4, en la base de datos de Mailman no se almacena el cuerpo del mensaje. Esto fue una optimización añadida por el parche<sup>2</sup> consiguiendo que los procesos en memoria de Mailman reduzcan en tamaño considerablemente. Al no ser posible recuperarlos de la base de datos, tendríamos que buscar ese cuerpo en el mbox de la lista, lo cual no está contemplado en el API de Mailman. Se utilizará el texto almacenado en HTML, el cual ya está correctamente convertido a caracteres fácilmente manipulables.

Los datos que son incluidos en el RSS:

Pippmail calcula un identificador para cada correo en función de las cabeceras del correo para indentar correctamente las conversaciones. Se ha aprovechado este mismo identificador para realizar la ordenación en la aplicación, por lo que el dato ha sido incluido en el RSS. El identificador es una cadena de texto separada por guiones. El

<sup>2</sup>[http://sourceforge.net/tracker/?func=detail&atid=100103&aid=835332&group\\_id=103](http://sourceforge.net/tracker/?func=detail&atid=100103&aid=835332&group_id=103)

número de guiones muestra el nivel de anidación del correo, y la cadena de texto el orden:

```
1- 01280823839.868 => 1
2- 01280823839.868-01280827010.869- => 2
3- 01280823839.868-01280827966.870- => 3
4- 01280823839.868-01280828936.873- => 5
5- 01280823839.868-01280827966.870-01280830144.872- => 4
```

## PyRSS2Gen

Para la generación del RSS se ha optado por la inclusión de un generador de RSS específico de Python. PyRSS2Gen<sup>3</sup> se encarga de generar la estructura específica del DTD<sup>4</sup> para el XML que publicamos. Al utilizar este paquete, evitamos tener que tratar a bajo nivel con la estructura de RSS, como el manejo de canales, formato adecuado para los distintos elementos como fechas, encoding, etc. Además se genera automáticamente el GUID (un identificador de cada noticia), con lo que nos aseguramos de pasar los test de validación de la W3c<sup>5</sup>

Para la generación del RSS sería suficiente con el siguiente código:

```
item = MMPyRSS2Gen(
    title = ..
    link = ..
    author = ..
    description = ..
    guid = ..
    pubDate = format_date(..),
)
item.add_values("thread", ..)

rss = PyRSS2Gen.RSS2(
    title = ..
    link = ..
    description = ..
    lastBuildDate = datetime.datetime.now(),
    items = item,
)
```

Para incluir nuevos atributos, se deben derivar las clases y utilizar `publish_extensions` para publicarlas. Para añadirlas, debemos redefinir `element_attrs` en la subclase derivada.

---

<sup>3</sup><http://dalkescientific.com/Python/PyRSS2Gen.html>

<sup>4</sup>Document Type Definitions

<sup>5</sup>[validator.w3.org/feed/](http://validator.w3.org/feed/)

```

class MMPyRSS2Gen(PyRSS2Gen.RSSItem):
    #Add own values to our PyRSS2Gen class
    def add_values(self, key, val):
        if not hasattr(self, 'attr'):
            self.attr = {}
        self.attr[key] = val

    def publish_extensions(self, handler):
        for key in self.attr.keys():
            PyRSS2Gen._opt_element(handler, key, self.attr[key])

```

## Links votaciones

Para incluir los links de votaciones se ha creado una nueva categoría de *footer*, o firma para los correos. De este modo, se puede configurar Mailman para utilizar estos links en caso de necesidad, como se muestra en la figura 4.3 .

```

diff -bcrB mailman/Mailman/Handlers/CookHeaders.py mailman/Mailman/Handlers/CookHeaders.py
*** mailman/Mailman/Handlers/CookHeaders.py 2009-11-30 18:13:12.000000000 -0100
--- mailman/Mailman/Handlers/CookHeaders.py 2010-03-28 19:08:07.000000000 -0100
*****
*** 67,72 ****
--- 67,79 ----

    def process(mlist, msg, msgdata):
+
+     #Add X-Votemm information header for Thunderbird plugin VoteMM
+     try:
+         import hashlib
+         msg['X-Votemm'] = hashlib.md5(msg['Message-Id'][1:-1]).hexdigest()
+     except Exception, e:
+         syslog('error', 'error with Votem-MM Thunderbird ext. integration: %s', e)
+
+     # Set the "X-Ack: no" header if noack flag is set.
+     if msgdata.get('noack'):
+         del msg['x-ack']
diff -bcrB mailman/Mailman/Handlers/Decorate.py mailman/Mailman/Handlers/Decorate.py
*** mailman/Mailman/Handlers/Decorate.py 2009-11-30 18:13:12.000000000 -0100
--- mailman/Mailman/Handlers/Decorate.py 2010-03-28 19:36:50.000000000 -0100
*****
*** 66,71 ****
--- 66,78 ----

        pass

        # These strings are descriptive for the log file and shouldn't be i18n'd
        d.update(msgdata.get('decoration-data', {}))

+
+     # Add X-Votemm to dictionary for correct info on footer/herder
+     try:

```

```

+         d['x_votemm'] = msg['X-Votemm']
+     except Exception, e:
+         syslog('error', 'error with Votem-MM Thunderbird ext. integration: %s', e)
+
+     header = decorate(mlist, mlist.msg_header, 'non-digest header', d)
+     footer = decorate(mlist, mlist.msg_footer, 'non-digest footer', d)
+     # Escape hatch if both the footer and header are empty

```

La clase Decorate procesa cada correo con la información contenida dentro del mismo. Al no existir ninguna forma de conseguir el message-id (ya que no se guarda en *msg*, que son los elementos de *Mailing*, objeto principal de Mailman) debemos incluirlo en un estado anterior, cuando se parsean las cabeceras contenidas en el mensaje cargado en memoria. Una vez tenemos el dato disponible, podemos añadirlo en el diccionario

**Administración de la lista de distribución Testlist1**  
**Sección de Opciones de entrega regular**

---

**Categorías de configuración**

- [Opciones Generales](#)
- [Claves](#)
- [Opciones de idiomas](#)
- [Administración de los suscriptores...](#)
- **[Opciones de entrega regular](#)**
- [Opciones de recopilaciones](#)

**Otras actividades administrativas**

- [Ocuparse de las peticiones pendientes de moderar](#)
- [Ir a la página de información general sobre la lista](#)
- [Editar el código HTML de las páginas de acceso y los ficheros de texto públicos](#)
- [Ir al archivo de la lista](#)

• **[Desconexión](#)**

---

Haga sus cambios a continuación y confírmelos utilizando el botón del final.

**Opciones de entrega regular**

Política referente a la entrega del tráfico de la lista a medida que llega

Descripción	Valor
¿Pueden los suscriptores elegir entre recibir el correo inmediatamente o recibirlo agrupado en un solo mensaje (digests)? <a href="#">(Editar nondigestable)</a>	<input type="radio"/> No <input checked="" type="radio"/> Sí
Cabecera a añadir al correo enviado a los suscriptores regulares de la lista <a href="#">(Detalles de msg_header)</a>	
Pié de página a añadir al correo enviado a los suscriptores regulares de la lista <a href="#">(Detalles de msg_footer)</a>	<pre> up! %(web_page_url)s%(x_votemm)s no: %(web_page_url)s%(x_votemm)s/-1 %(real_name)s mailing list %(real_name)s@%(host_name)s %(web_page_url)slistinfo%(cgtext)s/% (_internal_name)s </pre>

Figura 4.3: Configuración pie de página

En la firma de los correos se debe incluir la información suficiente para que la aplicación pueda identificar con un enlace el correo votado, de manera inequívoca. El identificador único de cada correo es el message-id, dato interno de Mailman calculado a partir de una función hash del asunto, información parcial de la hora de entrega en el servidor, y parte de información acerca del remitente. De esta forma, es razonable que no se vaya a llegar a ninguna colisión entre correos.

Este identificador en algunos casos es excesivamente largo, de tal forma que al ser

incluido en la firma del correo dentro del enlace, esta línea sería probablemente partida en 2 renglones (la longitud del enlace no debería exceder los 90 caracteres).

Para evitar el cambio de línea, y para mejorar la estética del mensaje se planteó utilizar algún reductor de direcciones, al estilo de tinyurl.com, bit.it, etc versus utilizar un md5 del message-id, con una longitud definida.

Los *reductores* de url asocian a un id y el enlace a acortar. En la forma más básica de estos reductores, se cambia la base del identificador a base64, utilizando un alfabeto formado por letras + LETRAS + números, a partir de un identificador que no está relacionado con la url, por lo que utilizar esta aproximación generaría otro punto de fallo en la aplicación.

```
>>> base64= BaseConverter('ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789abcdefghijklmnopqrstuvwxyz')
>>> base64.from_decimal(942344)
'DxJG'
>>> base64.to_decimal('DxJG')
942344
```

Usando la secuencia de correos en cada lista, la dirección de las votaciones con la lista más larga de gsync quedaría:

```
http://vote.gsync.es/libresoft-commit-watchers/B1Dg
http://vote.gsync.es/99a18b7f5238ef5cb92264cc432fe935
```

Ya que la información del message-id es suficiente como para no tener que incluir la dirección de la lista, en el peor caso no excedemos la longitud de la línea en el cuerpo del mensaje.

## Interfaz Piplermail

La interfaz web de Piplermail se ha modificado para incluir la información html respecto al RSS. De esta forma, los navegadores web encuentran existe un enlace al RSS de cada lista en las páginas de cada una.

Para generar correctamente el enlace necesitamos la ruta completa al recurso. Como Mailman, a través de los archivos de configuración, puede variar las rutas relativas, se debió incluir una nueva variable en la clase HTMLFormatter (<mm-archive-url>) con esta información:

<code>__init__.py</code>	Fichero vacío, opcional
<code>manage.py</code>	Clase con las opciones por defecto, iniciador de la aplicación
<code>settings.py</code>	Configuración de la aplicación, módulos cargados
<code>urls.py</code>	Clase con las expresiones regulares de las direcciones definidas, instanciará un objeto por cada dirección

Cuadro 4.2: Ficheros generados por Django

```
diff -bcrB mailman/Mailman/HTMLFormatter.py mailman/Mailman/HTMLFormatter.py
*** mailman/Mailman/HTMLFormatter.py 2009-11-30 18:13:10.000000000 -0100
--- mailman/Mailman/HTMLFormatter.py 2010-03-28 10:56:01.000000000 -0100
*****
*** 388,393 ****
--- 388,394 ----
        '<mm-form-end>' : self.FormatFormEnd(),
        '<mm-archive>' : self.FormatArchiveAnchor(),
        '</mm-archive>' : '</a>',
+       '<mm-archive-url>' : self.GetBaseArchiveURL(),
        '<mm-list-subscription-msg>' : self.FormatSubscriptionMsg(),
        '<mm-restricted-list-message>' : \
            self.RestrictedListMessage(_('The current archive'),
```

## 4.3. Aplicación web Vote-MM

### 4.3.1. Creación del proyecto

Podemos referirnos a la sección 4.5 para instalar Django en el sistema. Para comprobar si podemos cargar el entorno, ejecutaremos `import django` desde una consola de Python, si no se muestran errores podremos continuar con la creación del proyecto. Django ofrece un binario con el que administrar los diferentes proyectos, llamado `django-admin.py`.

```
alks@alkslapt:~/pfc/code$ django-admin.py startproject mmwvs
alks@alkslapt:~/pfc/code$ ls mmwvs/
__init__.py  manage.py  settings.py  urls.py
```

El asistente creará los ficheros con las clases iniciales del proyecto:

Una vez creada la aplicación, y definidos en el `settings.py` los datos de acceso a la base de datos y las aplicaciones que serán utilizadas en el proyecto, podemos crear las estructuras de base de datos necesarias. Con el asistente podemos sincronizar la base

de datos con la información definida.

Por defecto Django incluye algunos módulos, entre ellos la gestión de usuarios, *Admin*, etc que serán incluidos en la base de datos en la primera sincronización.

```
alks@alkslapt:~/pfc/code/mmwvs$ python manage.py syncdb
Creating table auth_permission
Creating table auth_group
Creating table auth_user
Creating table auth_message
Creating table django_content_type
Creating table django_session
Creating table django_site
```

```
You just installed Django's auth system, which means you don't have any superusers defined.
Would you like to create one now? (yes/no): yes
Username (Leave blank to use 'alks'): alks
E-mail address: alks@testing.net
Password:
Password (again):
Superuser created successfully.
Installing index for auth.Permission model
Installing index for auth.Message model
alks@alkslapt:~/pfc/code/mmwvs$
```

Una vez el proyecto está definido, crearemos la aplicación MMWVS (MailMan Web Voting Service).

```
python manage.py startapp mmwvs
```

Añadimos la nueva aplicación en la variable `INSTALLED_APPS` dentro del archivo `settings.py` para a continuación definir las estructuras de datos que serán utilizadas por la aplicación. Django ofrece una serie de tipos de datos predefinidos que podemos utilizar, aportando algunas características importantes que lo diferencian de otros proyectos similares como la sincronización automática de la base de datos y la posibilidad de generar automáticamente formularios y estructuras para imprimir la información en HTML. Una vez creadas las clases con la información a almacenar, podremos sincronizar la base de datos:

```
python manage.py validate
python manage.py syncdb
```

La figura 4.4 representa la estructura de la base de datos creada para la aplicación. Por otro lado, la figura 4.5 muestra las clases generadas para la aplicación web del proyecto.

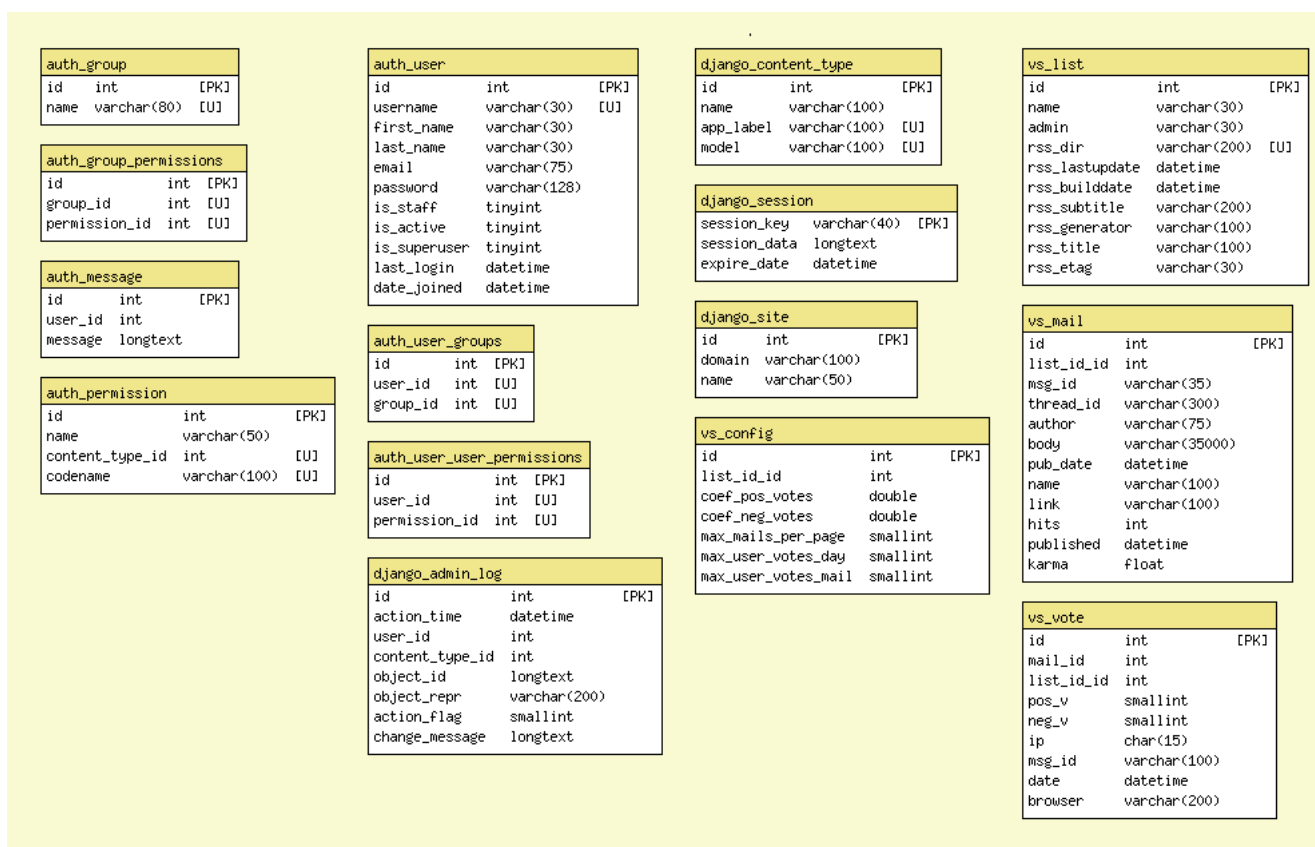


Figura 4.4: Clases BBDD Vote-MM

El binario *manage.py* ofrece algunas funcionalidades interesantes para la etapa de desarrollo de la aplicación. Al ser Python un lenguaje interpretado, podemos obtener una shell de Python con las clases en memoria del proyecto, de esta forma podemos interactuar directamente con la aplicación y los datos.

```

alxs@votemm:/var/www/mmwvs$ python manage.py shell
Python 2.5.2 (r252:60911, Jan 24 2010, 14:53:14)
[GCC 4.3.2] on linux2
Type "help", "copyright", "credits" or "license" for more information.
(InteractiveConsole)
>>> from mmwvs.vs.models import *
>>> from mmwvs.vs.views import *
>>> l = List.objects.get(name="Votemm")
>>> m = Mail.objects.all().filter(list_id=1)
>>> m = Mail.objects.all().filter(list_id=1).filter(name="noexiste")
>>> m
[]
>>> m = Mail.objects.all().filter(list_id=1)
>>> m
[<Mail: 12016e9a5d17e41d8449510369652556>, <Mail: ed3d883fd176d873a88eb3e0874dd5fe>, <Mail: a170a64b3323

```



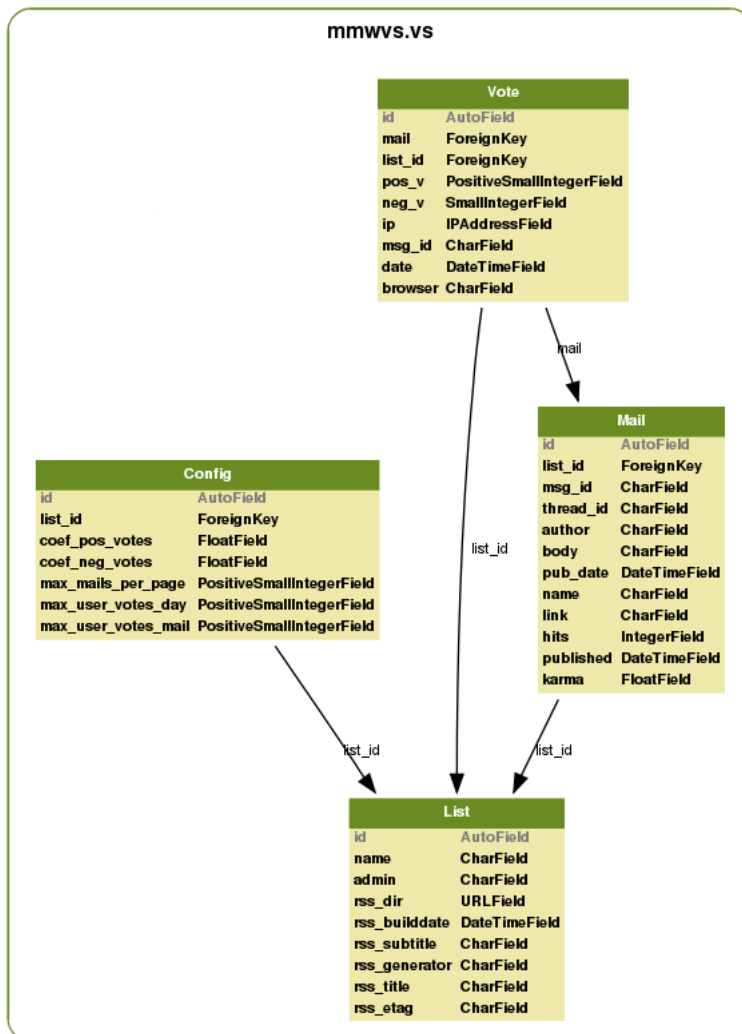


Figura 4.5: Clases Vote-MM

## Templates

Django utiliza plantillas para separar la presentación de los datos de los datos en sí. De una forma resumida, las plantillas permiten la utilización de palabras clave además de la inclusión de HTML. Los *tags* permiten el uso de filtros, bucles, variables, inclusión de archivos o otros templates, etcétera.

```

{% if mostrar_informacion %}
<ul>
{% for elemento in lista%}
<li> Esto es una lista de: {{ elemento }}</li>
{% endfor %}

```

```
</ul>
{% endif %}
```

```
<p />Aquí utilizamos tipos de datos de Python para fechas: {{ variable_fecha|date:"F j, Y"}}
```

Los filtros son funciones que realizan modificaciones, desde truncar palabras hasta acciones más complejas como incluir calendarios. Esto facilita el desarrollo de documentos web, ya que podemos tener acciones específicas para cada tipo de datos (fechas, tamaños de archivos..) utilizando el formato más legible con poco trabajo.

## 4.4. Plugin Thunderbird

Actualmente, las versiones más extendidas de Thunderbird utilizan el motor de gecko 1.8.1X. Aunque la versión de desarrollo (1.9.X) provee de un framework mucho más potente, ya que cuenta con objetos instanciables desde javascript de JSON y xmlrpc además de nuevas interfaces con las que el desarrollo de interfaces se simplifica, pudiendo utilizar un API más extenso de detección de eventos, listeners, y una serie de interfaces que permiten la inclusión de librerías externas como componentes (javascript, c++), no se ha elegido esta con el fin de maximizar el número de usuarios; de otra forma, al estar gecko 1.9 disponible únicamente para Thunderbird 3, limitaría el número de usuarios a los dispuestos a actualizar el cliente de correo a las últimas versiones no estables.

El plugin está definido por una jerarquía de directorios, con una serie de archivos con una utilidad determinada, como se muestra a continuación:

## 4.5. Instalación

El servidor sobre el que se ha realizado la puesta a producción de la aplicación ha sido una máquina virtual basada en XEN[]

```
Linux votemm.libresoft.es 2.6.26-2-xen-686 #1 SMP Fri Aug 14 04:00:01 UTC 2009 i686 GNU/Linux
```

La distribución instalada es Debian Lenny (versión actual estable de la distribución). A continuación se van a repasar las configuraciones e instalaciones necesarias para la puesta en marcha del proyecto.

Archivo	Propósito
./locale/en-US/prefwindow.dtd	Mensajes de texto expuestos en la interfaz. Traducciones utilizando LOCALE
./defaults/preferences/votemm.js	Preferencias del plugin.
./content/options.xul	Archivos XUL con la definición del layout
./content/json2.js	Librería javascript con una implementación JSON.
./content/votecol.js	Javascript con la lógica del plugin.
./content/thunderbirdOverlay.xul	Archivo XUL que modifica el layout por defecto del cliente.
./install.rdf	Información sobre actualizaciones, identificadores y características del plugin.
./chrome.manifest	Localización de los recursos.
./skin/overlay.css	CSS para el layout de los XUL.
./config_build.sh	Script que empaqueta los archivos en un fichero .xpi

Cuadro 4.3: Jerarquía de archivos del plugin

## Python 2.6

La versión instalada por defecto en la máquina es Python-2.5, versión que no provee de algunas características de introspección de tipos que son necesarias para el middleware de JSON instalado en Django. Para subir de versión a Python2.6 las preferencias del gestor de paquetes apt han tenido que ser modificadas incluyendo los repositorios de Debian *testing* y *unstable*. Asignando un *Pin-priority* diferente para cada distribución podemos incluir paquetes de otras versiones Debian.

```
votemm:~# cat /etc/apt/sources.list
deb      http://ftp.de.debian.org/debian/      stable main contrib non-free
deb      http://ftp.de.debian.org/debian/      testing main contrib non-free
deb      http://security.debian.org/ stable/updates main contrib non-free
deb      http://security.debian.org/ testing/updates main contrib non-free
votemm:~# cat /etc/apt/preferences
Package: *
Pin: release a=stable
Pin-Priority: 700
Package: *
Pin: release a=testing
Pin-Priority: 650
Package: *
Pin: release a=unstable
Pin-Priority: 600
```

De esta forma, podemos instalar Python versión 2.6:

```
apt-get install -t testing python2.6 python2.6-dev
```

## Django

La instalación de Django es sencilla, como requisito principal debemos tener el paquete *python-mysqldb* instalado, además de la versión de desarrollo de Python versión

### 2.6. Utilizando la paquetería oficial<sup>6</sup>:

```
Django-1.1.1\>$ sudo python setup.py install
tar xvzf Django-1.1.1.tar.gz
cd Django-1.1.1/
sudo python setup.py install
..
alks@alkslapt:~/pfc/apps/Django-1.1.1\>$ python
Python 2.6.2 (release26-maint, Apr 19 2009, 01:56:41)
[GCC 4.3.3] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import django
>>>
```

## Servidor web Cherokee

Como servidor web, se ha elegido Cherokee dada la facilidad de administración que presenta gracias a la interfaz web de configuración. Este servidor web soporta el protocolo fastcgi así como uwsgi (versión mejorada para fastcgi), apto para la comunicación con la aplicación web escrita en Python.

```
pfc-votemm:~# wget http://www.cherokee-project.com/cherokee-latest-tarball
pfc-votemm:~# tar xzf cherokee-0.99.44.tar.gz
pfc-votemm:~# cd cherokee-0.99.44
pfc-votemm:~/cherokee-0.99.44# ./configure --localstatedir=/var --prefix=/usr --sysconfdir=/etc --with-
----
alks@test1:~/cherokee-0.99.28$ make
alks@test1:~/cherokee-0.99.28$ sudo make install
```

Ejecutamos *cherokee-admin* para levantar la interfaz de configuración del servidor web.

```
pfc-votemm:/home/alks/code/mmwvs# cherokee-admin -b 0.0.0.0
Login:
  User:                admin
  One-time Password:  1lCa1KE2HVPqYNv2
```

```
Web Interface:
  URL:                 http://localhost:9090/
```

```
Cherokee Web Server 0.99.44 (Apr 12 2010): Listening on port ALL:9090, TLS
disabled, IPv6 disabled, using epoll, 4096 fds system limit, max. 2041
connections, caching I/O, single thread
```

---

<sup>6</sup><http://www.djangoproject.com/download/>

## uWSGI

Para servir el contenido generado por django, utilizaremos WSGI<sup>7</sup>. Es una interfaz que detalla un protocolo de comunicación (uwsgi) entre servidores web y aplicaciones web o entornos de desarrollo (frameworks). El servidor de WSGI utilizado será uWSGI<sup>8</sup>, es un servidor rápido y sencillo, escrito en C y con un uso muy eficiente de la memoria. Algunas características:

- Soporte para múltiples aplicaciones en el mismo servidor
- Servidor de archivos estáticos via `sendfile()`
- Modo Harakiri, para sus propios procesos en caso de incomunicación.
- Configuración con XML
- Aplicación en Django para manejar el servidor uWSGI (`uWSGIDjangoApp`)
- Soporte para VirtualHosts

El proceso de instalación es sencillo, las únicas dependencias son solventadas con el intérprete de Python:

```
pfc-votemm:~# wget http://projects.unbit.it/downloads/uwsgi-0.9.4.3.tar.gz
pfc-votemm:~# tar xzf uwsgi-0.9.4.3.tar.gz
pfc-votemm:~# cd uwsgi-0.9.4.3
pfc-votemm:~/uwsgi-0.9.4.3# make
pfc-votemm:~/uwsgi-0.9.4.3# cp uwsgi /usr/local/bin/
```

## Servidores Postfix, Mysql

La instalación del MTA se ejecuta utilizando apt, al igual que el servicio Mysql. Se debe configurar el *relayhost* para el envío de correos, que será la pasarela autorizada de la red Libresoft para el envío de los mismos.

```
pfc-votemm:~# aptitude install mysql-server postfix
```

---

<sup>7</sup>Web Server Gateway Interface

<sup>8</sup><http://projects.unbit.it/uwsgi/>

## Mailman

La instalación de Mailman requiere de la integración con el MTA del servidor una vez instalada la aplicación. Descargaremos la versión estable y la instalaremos como se detalla a continuación:

```
wget http://ftp.gnu.org/gnu/mailman/mailman-2.1.12.tgz
..
tar xvzf mailman-2.1.12.tgz
cd mailman-2.1.12
adduser mailman
groupadd mailman -u mailman
groupadd mailman
passwd mailman

mkdir /var/mailman
chown mailman:mailman /var/mailman/
chmod 02775 /var/mailman
./configure --prefix=/var/mailman/ --exec-prefix=/var/mailman/ --with-mail-gid=postfix --with-mailhost=
make
make install
```

Editamos `/etc/postfix/mail.cf` para incluir los alias generados por Mailman, y indicaremos a Mailman que el MTA del sistema es Postfix:

```
alias_maps = hash:/etc/aliases, hash:/var/mailman/data/aliases
[mailman@test1 mailman]$ ls -la data/aliases*
-rw-rw---- 1 mailman mailman 2628 Nov 28 16:40 data/aliases
-rw-r----- 1 mailman mailman 12288 Nov 28 16:40 data/aliases.db
```

Una vez instalado Mailman, configuraremos una lista de correo:

```
pfc-votemm:/var/mailman/bin# ./newlist
Enter the name of the list: Votemm
Enter the email of the person running the list: cesar@pk2.org.org
Initial votemm password:
Hit enter to notify votemm owner...
pfc-votemm:~/mailman-2.1.13# echo "test" | mail -s "test2" votemm@votemm.gsync.es
pfc-votemm:~/mailman-2.1.13# mail
Mail version 8.1.2 01/15/2001. Type ? for help.-s "testt" votemm@votemm.gsync.es
"/var/mail/root": 1 message 1 new
>N 1 mailman-bounces@v Mon Apr 19 18:47 40/1348 Your message to Mailman awaits moderator approval
```

## Instalación aplicación web

Como el servidor web Cherokee no incluye un intérprete de Python integrado, utilizaremos otra aplicación entre el servidor web y el intérprete. Para ello utilizaremos

uwsgi, es un demonio que se comunica con el servidor a través del protocolo del mismo nombre. Para configurar uwsgi utilizaremos el siguiente archivo:

```
pfc-votemm:/home/alks/code/mmwvs# cat uwsgi.conf
<uwsgi>
  <pythonpath>/home/alks/pfc/code/mmwvs/</pythonpath>
  <pythonpath>/home/alks/pfc/code/mmwvs/vs/</pythonpath>
  <app mountpoint="/">
    <script>django_wsgi</script>
  </app>
</uwsgi>
```

También será necesario un script para lanzar el servicio WSGI de django.

```
pfc-votemm:/home/alks/code/mmwvs# cat django_wsgi.py
import inspect
import os
import sys

apache_dir = os.path.dirname(os.path.realpath(inspect.getfile(inspect.currentframe())))
project_dir = os.path.dirname(apache_dir)
if project_dir not in sys.path:
    sys.path.append(project_dir)

import settings
from django.core.management import setup_environ
from django.core.handlers.wsgi import WSGIHandler

setup_environ(settings)
application = WSGIHandler()
```





# Capítulo 5

## Estudio

La implementación del estudio consta de varias etapas. Primero definiremos la publicación en sí, la repercusión obtenida y el análisis de cómo se ha llegado a los usuarios. Para el estudio, se ha elegido un una lista de distribución de correo que lleva varios años en funcionamiento, se trata de la lista FANS, del grupo de investigación Libresoft de la Universidad Rey Juan Carlos de Madrid, y se ha estudiado el uso de la nueva interfaz y características del mismo.

Además, se ha realizado una encuesta de evaluación donde se ha obtenido *feedback* sobre la aplicación, se repasarán esas impresiones de los usuarios en el apartado 5.2.

### 5.1. Publicación

En primer lugar, el 5 de marzo de 2010 se expuso el proyecto en su fase de desarrollo al grupo de investigación Libresoft. Con una pequeña presentación de una hora y media, se expusieron los objetivos de la aplicación al grupo de investigación, junto con algunas capturas de pantalla de la interfaz web desarrollada.

En general el concepto interesó a la audiencia y en el turno de preguntas se propusieron algunas mejoras que finalmente han sido también incorporadas, tales como la interfaz JSON para aumentar los canales por los que la información puede llegar al usuario y la implementación de un plugin para el cliente de correo, donde poder visualizar las votaciones e interactuar con el sistema; el grupo estaba de acuerdo en que es probable que el usuario que lea los archivos de la lista desde un cliente de correo no deje de utilizar el programa.

La publicación del proyecto dentro de la lista el 21 de Mayo se realizó sin inci-

dentes para los usuarios. La migración a la nueva plataforma generó un número importante de visitas únicas en los 2 primeros días.

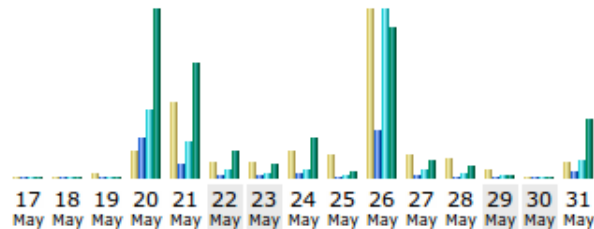


Figura 5.1: Opciones SourceForge

Una serie de usuarios reportaron a la lista la posibilidad de votar un número indeterminado de veces por un mismo correo. Esta problemática ya había sido evaluada anteriormente, expuesta en el punto 3.2. Como medida preventiva ante este tipo de situaciones, se habilitó un mecanismo en el que se evalúa la dirección IP de procedencia del voto junto con la cabecera User-Agent del navegador, descartando los votos que coinciden. De esforma permitiríamos votaciones en varios navegadores de un mismo usuario, pero no cerramos las votaciones a redes enteras que naveguen a través de un servidor proxy. En las estadísticas de la figura 5.1 podemos ver el aumento considerable de peticiones el día 26, en especial el aumento de la información servida aunque no del número de usuarios.

Sin embargo no podemos distinguir entre el tráfico legítimo inspirado por el interés en la aplicación, del que generó el propio anuncio del problema en la lista.

Para la publicación de la aplicación fuera del grupo de la universidad se utilizaron varios sitios web donde se anunció el proyecto. Por una lado se anunciaron los parches que pueden ser útiles en Mailman por sí mismos (como la generación de RSS para Pippmail) y por otro se fomentó el uso de la aplicación, ofreciendo varias posibilidades:

- Uso de toda la arquitectura (Servicio web, Mailman)
- Uso de la plataforma (votemm.libresoft.es) gratuitamente:
  - Permitiendo el uso de la infraestructura, leyendo RSS de otras máquinas.
  - Suscribiendo a la máquina para recibir copias de los correos (evitando la obligación de aplicar el parche de RSS)

15 May 2010	0	0	0	0
16 May 2010	0	0	0	0
17 May 2010	0	0	0	0
18 May 2010	0	0	0	0
19 May 2010	1	1	1	1.30 KB
20 May 2010	7	553	965	4.37 MB
21 May 2010	20	184	504	3.00 MB
22 May 2010	4	42	110	724.33 KB
23 May 2010	4	34	63	365.58 KB
24 May 2010	7	65	128	1.04 MB
25 May 2010	6	8	45	151.32 KB
26 May 2010	44	662	2350	3.93 MB
27 May 2010	6	43	109	454.34 KB
28 May 2010	5	12	73	321.42 KB
29 May 2010	2	7	27	50.34 KB
30 May 2010	0	0	0	0
31 May 2010	4	82	248	1.55 MB

Figura 5.2: Visitas en los días de promoción del proyecto

Los parches para Mailman se anunciaron en primer lugar en la página del proyecto, habilitando la entrada de buscadores a través de un robots.txt (ver figura 5.3). Para que los buscadores pudieran acceder a la dirección (una referencia externa a la página) se dejó un comentario en un ticket de launchpad de un parche con soporte RSS para Mailman en el que se basó el proyecto<sup>1</sup>

Date: 2010-06-05 17:28:32 UTC

Sender: cfgago

Based on you patch, I've developed another patch for mailman.

It includes the body of the messages as well as author,published date, link to pipermail archives, thread information..

It's part of a bigger project which add's voting capabilities for mailman:

<http://votemm.libresoft.es/votemm/>

```
User-agent: *
Disallow: /mail/
Disallow: /mailman/
Allow: /
```

Figura 5.3: robots.txt

El 6 de Junio se creó el proyecto Vote-MM en la forja de *sourceforge*<sup>2</sup>.

<sup>1</sup> [http://sourceforge.net/tracker/?func=detail&aid=657951&group\\_id=103&atid=300103](http://sourceforge.net/tracker/?func=detail&aid=657951&group_id=103&atid=300103)

<sup>2</sup><http://sourceforge.net/projects/vote-mm/>

La publicación en *Sourceforge* se realizó con el objetivo de la publicación del proyecto. Aunque no todas las herramientas ofrecidas han sido usadas, se planificó que desde este sitio podríamos llegar a un grupo potencial de usuarios más amplio que dejando solamente la web disponible en Libresoft. En el proceso de alta del proyecto es necesaria una pequeña descripción y la licencia con la que vamos a publicarlo. Se deben seleccionar las aplicaciones que estarán disponibles como indica la figura 5.4

Enabled	Feature	Category	Status	Options
✓	<a href="#">Backups</a>	Data Recovery	Always On	<a href="#">XML Export, Hosted Apps</a>
✓	<a href="#">File Manager</a>	Download	Always On	<a href="#">Manage</a>
✓	<a href="#">Help Wanted</a>	Recruiting	Always On	<a href="#">Submit, Manage</a>
✓	<a href="#">Project Database (MySQL)</a>	Web Hosting	Always On	<a href="#">Manage</a>
✓	<a href="#">Project Web</a>	Web Hosting	Always On	
✓	<a href="#">Virtual Hosts (VHOSTs)</a>	Web Hosting	Always On	<a href="#">Manage</a>
✓	<a href="#">Forums</a>	Forums	Enabled	<a href="#">Manage</a>
✓	<a href="#">Screenshots</a>	Media	Enabled	<a href="#">Manage</a>
✓	<a href="#">Subversion</a>	Source Control	Enabled	<a href="#">Manage</a>
✓	<a href="#">Tracker</a>	Bug / Defects	Enabled	<a href="#">Manage</a>

Figura 5.4: Opciones SourceForge

La aplicación web ha recibido una cantidad limitada de tráfico, como podemos apreciar en la figura 5.5. La mayor parte del tráfico que se ha recibido desde la web de *Sourceforge*, seguidos de *Freshmeat* y enlaces de buscadores, con los términos de búsqueda de la tabla 5.1.

Una opción interesante para la publicitación del proyecto es *Trove Categorization*. Con este mecanismo, además de los *tags* asignados al proyecto, podemos categorizarlo de una forma más exhaustiva, haciendo más fácil la búsqueda de software relacionado.

Se utilizó el SVN proporcionado por SourceForge para subir los parches de Mailman, el código de plugin para Thunderbird y la aplicación web.

El 7 de Junio, se anunció el proyecto en *freshmeat*<sup>3</sup>, como muestra la figura 5.8. La evolución de visitas en la página a aumentado con el tiempo, según las referencias y los buscadores han ido indexando el contenido, como podemos ver en la figura 5.9.

La evolución de votos a lo largo del tiempo en la aplicación queda reflejada en la

<sup>3</sup><http://freshmeat.net/>

<i>mailman rss plugin</i>
<i>mailman rss patch</i>
<i>mailman 2.1.13 rss patch</i>
<i>mailman templates</i>
<i>rss mailman plugin</i>
<i>mailman generate rss patch</i>
<i>votemm rss</i>
<i>mailman rss</i>
<i>rss mailman</i>

Cuadro 5.1: Términos utilizados por los buscadores

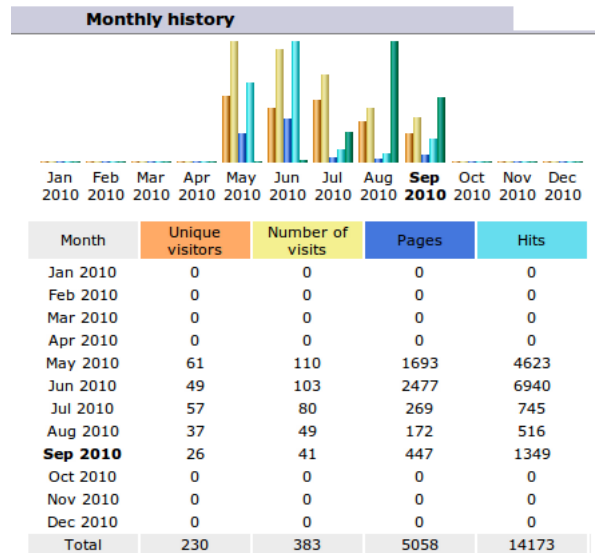


Figura 5.5: Número de visitas totales del proyecto

figura 5.10.

## 5.2. Encuesta

La encuesta realizada fue sencilla y rápida de responder, para no exigir demasiado esfuerzo al usuario y tener el mayor número de impresiones. En la figura 5.11 podemos ver las preguntas realizadas y a continuación el estudio sobre sus respuestas.

En la figura 5.12 podemos ver los resultados de la encuesta. Los datos más llamativos:

Links from an external page (other web sites except search engines)				
Total: 4 different pages-url	Pages	Percent	Hits	Percent
<a href="http://tinyurl.com/3ymgup7">http://tinyurl.com/3ymgup7</a>	3	50 %	3	50 %
<a href="http://identi.ca/oberger/all">http://identi.ca/oberger/all</a>	1	16.6 %	1	16.6 %
<a href="http://identi.ca/group/libresoft">http://identi.ca/group/libresoft</a>	1	16.6 %	1	16.6 %
<a href="http://untiny.me">http://untiny.me</a>	1	16.6 %	1	16.6 %

Links from an external page (other web sites except search engines)				
Total: 6 different pages-url	Pages	Percent	Hits	Percent
<a href="http://sourceforge.net/projects/vote-mm/">http://sourceforge.net/projects/vote-mm/</a>	6	54.5 %	6	54.5 %
<a href="http://us.mc1200.mail.yahoo.com/mc/welcome">http://us.mc1200.mail.yahoo.com/mc/welcome</a>	1	9 %	1	9 %
<a href="http://www.mail-archive.com/mailman-users@python.org/msg57247.ht...">http://www.mail-archive.com/mailman-users@python.org/msg57247.ht...</a>	1	9 %	1	9 %
<a href="http://freshmeat.net/projects/vote-mm">http://freshmeat.net/projects/vote-mm</a>	1	9 %	1	9 %
<a href="http://ar.mc463.mail.yahoo.com/mc/welcome">http://ar.mc463.mail.yahoo.com/mc/welcome</a>	1	9 %	1	9 %
<a href="http://searchassist.mediacomcable.com/mediacomsearch/ws/results/...">http://searchassist.mediacomcable.com/mediacomsearch/ws/results/...</a>	1	9 %	1	9 %

Links from an external page (other web sites except search engines)				
Exclude Filter 127.0.0.1: 4 different pages-url	Pages	Percent	Hits	Percent
<a href="http://freshmeat.net/projects/vote-mm">http://freshmeat.net/projects/vote-mm</a>	5	55.5 %	5	41.6 %
<a href="http://sourceforge.net/projects/vote-mm/">http://sourceforge.net/projects/vote-mm/</a>	2	22.2 %	2	16.6 %
<a href="http://freshmeat.net/projects/vote-mm/edit">http://freshmeat.net/projects/vote-mm/edit</a>	1	11.1 %	1	8.3 %
<a href="http://webcache.googleusercontent.com/search">http://webcache.googleusercontent.com/search</a>	1	11.1 %	4	33.3 %

Links from an external page (other web sites except search engines)				
Total: 4 different pages-url	Pages	Percent	Hits	Percent
<a href="http://mail.python.org/pipermail/mailman-users/2010-June/069668....">http://mail.python.org/pipermail/mailman-users/2010-June/069668....</a>	2	33.3 %	2	33.3 %
<a href="http://sourceforge.net/projects/vote-mm/">http://sourceforge.net/projects/vote-mm/</a>	2	33.3 %	2	33.3 %
<a href="http://es.mc296.mail.yahoo.com/mc/welcome">http://es.mc296.mail.yahoo.com/mc/welcome</a>	1	16.6 %	1	16.6 %
<a href="http://freshmeat.net/projects/vote-mm">http://freshmeat.net/projects/vote-mm</a>	1	16.6 %	1	16.6 %

Figura 5.6: Orígenes de las visitas. Mayo-Agosto

Development Status: 3 - Alpha  
 Intended Audience: End Users/Desktop, System Administrators  
 Programming Language: Python  
 Topic: Mailing List Servers, RSS Feed Readers

Figura 5.7: Trove categorization

- A todos los miembros les ha parecido buena idea el proyecto, en general.
- El desconocimiento de la existencia del plugin de Thunderbird, que quizá explique la falta de uso del mismo.
- Todos los miembros leen todos los mensajes de la lista, lo que evidencia la poca falta de priorización en la lista debido al escaso tráfico.
- No se ha utilizado la métrica propuesta para priorizar los correos, porque no hace falta.

Como razones principales para no usarlo, por orden de coincidencias, los usuarios aseguraron en primer lugar haberse olvidado de proyecto, seguido de vagancia, y que no existe demasiado tráfico. También han echo notar que la modificación de la firma

**freshmeat** In-line dedupe and compressionEnd vendor lock-in, save 75%  
[www.nexenta.com](http://www.nexenta.com)

Home Articles Browse Projects by Tag Submit new Project About Blog Help Sites  Search

[Projects](#) / [Vote-MM](#)

## Vote-MM

Vote-MM aims to improve current Mailman mailing lists with some Web 2.0 time-saving functionality (RSS and voting capabilities). It adds support for RSS to Mailman. The mail footer is modified to include voting links. A new Web app for reading mail/voting is provided. A Mozilla Thunderbird plug-in is also provided, so users can vote and see voting information.

Tags: [Mail Management](#) [voting](#) [Web 2.0](#) [Web Application](#)

Licenses: [GPL](#) [MPL](#)


Implementation: [Django](#) [xul](#) [JavaScript](#) [Python](#)


Translations: [English](#)

[Short link](#) [Tweet this project](#)

**Links**

[Project site](#)

 ofgago  
07 Jun 2010 20:23



Dependencies [Request ownership](#) [Report problem](#) [Graphs](#) [Submit a comment](#)

[filter](#) [subscribe](#)

Figura 5.8: Freshmeat

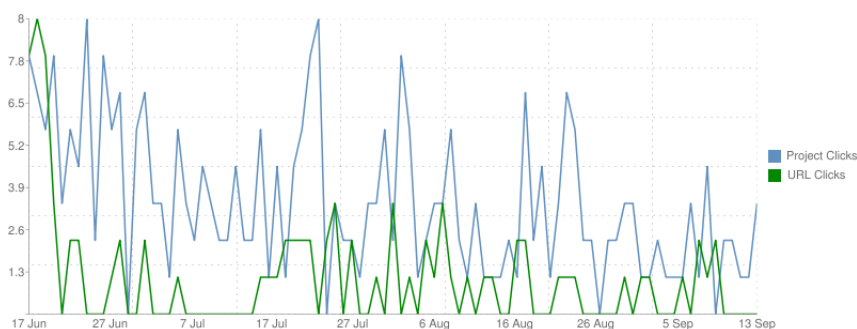


Figura 5.9: Número de visitas en Freshmeat

de Mailman parece demasiado intrusiva, y que la aplicación tiene más sentido para un webmail que como algo independiente. También se ha comentado que el uso de *gmail*, y la rapidez de sus búsquedas hacen que el sistema no sea necesario.

Por otro lado, en los comentarios, además de felicitar me por el buen trabajo, se han sugerido algunas mejoras que en su mayoría ya se han comentado en este capítulo. Algunos usuarios proponen que más que para uso diario la herramienta es útil realizar búsquedas en el histórico, tarea frecuente y tediosa. También se ha comentado que deberíamos haber sido más *pesados* y haber insistido en el uso de la herramienta.

Algunas propuestas novedosas, que no se habían evaluado con anterioridad:

- Utilizar un umbral de importancia, bajo el cual no se muestren los correos menos

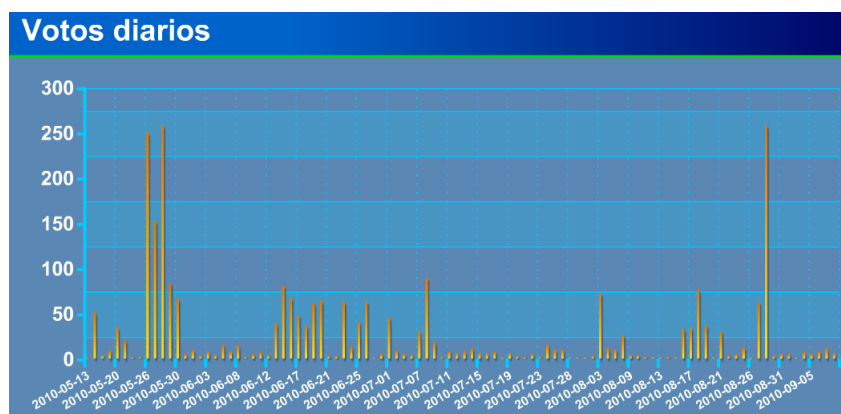


Figura 5.10: Evolución de votos diarios

votados, al estilo *Barrapunto*.

- La diferenciación entre votos locales y votos para el servidor, de forma que se elijan desde el cliente de correo qué puntuaciones compartir y cuales no.

### 5.3. Conclusiones

En la primera fase del proyecto, cuando se buscaban posibles carencias en el mundo del desarrollo del software libre y al evaluar la opción de modernizar el archivado para Mailman, no podíamos saber la aceptación que el sistema tendría para los usuarios; por una parte es un sistema que enriquece la información disponible con la que todos los usuarios podemos beneficiarnos, por otro lado, requiere del esfuerzo de los mismos para aportar el conocimiento, y no podíamos saber si los usuarios estarían dispuestos a invertir ese tiempo y esfuerzo.

El nicho donde hemos trabajado, aún siendo comparativamente grande en el mundo de desarrollo, es bastante reducido, no son muchos los usuarios que mantienen una gestor de listas de correo, pero sobre todo son pocas las nuevas instalaciones de estos gestores. Al contener información valiosa, Mailman es un sistema que se modifica con sumo cuidado y rara vez se parchea a no ser estrictamente necesario. Por tanto, la cantidad de usuarios buscando las ventajas que ofrece el proyecto es muy limitado.

El sistema compite directamente con otras tecnologías mucho más robustas y avanzadas como los buscadores. La búsqueda de información se simplifica haciendo que el uso del sistema no aporte mayor comodidad, aunque para ello los archivos tengan



**Vote-MM**

GSYC  
Grupo de Sistemas  
y Telecomunicaciones  
[votemm@votemm.libresoft.es](mailto:votemm@votemm.libresoft.es)

Home | Contact Us!

What is Vote-MM System Log-in ?

| Back

**Por favor, responde a las siguientes preguntas..**

¿Te ha parecido una buena idea, en general, el sistem?

¿Has utilizado el sistema diariamente, al menos durante el primer mes?

¿Has priorizado algún correo por los votos que tenía?

¿Conoces el plugin para thunderbird?

¿Lees habitualmente los correos de la lista?

¿Tardas más de un día en ver los correos de FANS?

¿Has encontrado algo que no haya cumplido tus expectativas?

¿Por qué no has utilizado más la aplicación?

Comentarios / Sugerencias

Gsync © 2010 | Gsync

Figura 5.11: Encuesta realizada a los miembros de FANS

que ser públicos (lo que no siempre es posible). La generalización del uso del web-mail (gracias en parte a la desaparición de las cuotas de espacio) en detrimento del cliente de correo, utilizando motores de búsqueda muchos más rápidos que el que el cliente pueda ofrecer, disminuye la necesidad de jerarquización y organización de la información. Si bien este software está pensado para priorizar el correo que un usuario tenga pendiente de leer, la inercia del comportamiento del usuario hace que se filtre la información por el título del mensaje, no necesitando otras métricas para discernir su importancia, si el volumen de correos es suficientemente pequeño.

Por otro lado, no existe un beneficio directo entre votar un mensaje o no hacerlo. Al tratarse de una comunidad ya formada, donde escasamente entran nuevos miembros y existen relaciones personales entre ellos, la importancia de los envíos radica especialmente en el emisor, conocido por todos los miembros y por ende la criticidad de la información que aporta.

Además, el tráfico de correos en FANS es muy limitado, como se muestra en la figura 5.13 el máximo de correos en un día se ha situado en 27, mientras que la media diaria de envíos de los últimos 4 meses es de 2.25. Estas cifras sugieren que la necesidad de priorizar la información es muy limitada.

Sin embargo, la implementación del sistema en la lista de FANS, ha sido un éxito.

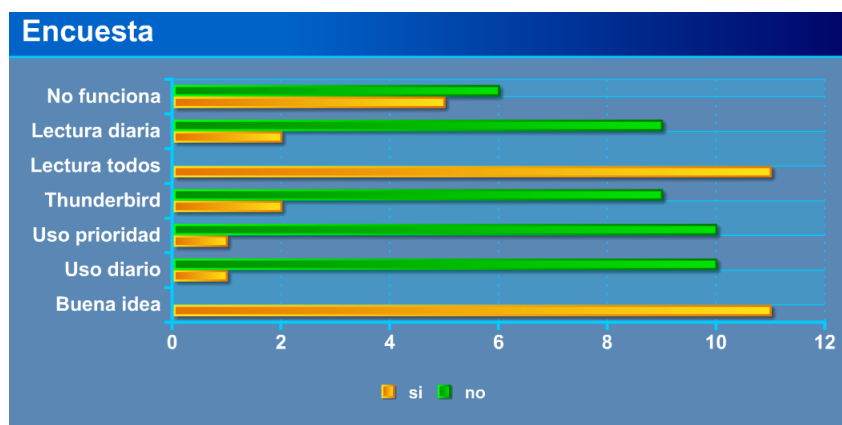


Figura 5.12: Encuesta realizada a los miembros de FANS

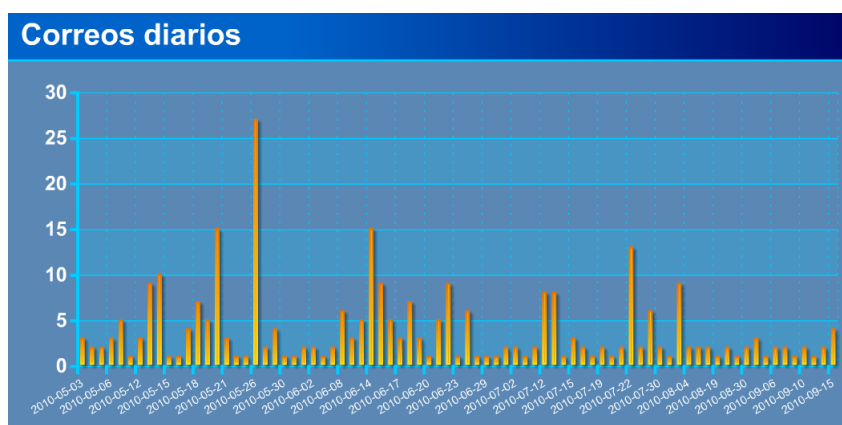


Figura 5.13: Evolución de correos enviados a la lista

No se ha percibido pérdida de rendimiento ni se han perdido correos en la migración; además, gracias a la colaboración del grupo de investigación Libresoft<sup>4</sup> se han podido realizar las pruebas suficientes para asegurarnos que todos los elementos del sistema eran estables: los MTA de correo, el archivado de la lista, el servidor web, de base de datos, etc. Esta experiencia ha demostrado que el sistema funciona, que no se pierden datos ni en la comunicación entre Mailman y la aplicación web, ni en la lista de correos.

El poco (o nulo) éxito del plugin para el cliente de correo Thunderbird (ver figura 5.14) sugiere que o bien el usuario utiliza otro cliente de correo, o bien consultan los archivos vía webmail. Para la primera cuestión no hemos podido hacer nada, sin embargo si el cliente utiliza webmail, el salto del webmail a la aplicación web es sencillo, y por ello la interfaz se ha diseñado específicamente para favorecer la lectura

<sup>4</sup><http://libresoft.es>

de listas de correo, sin embargo, tampoco se ha apreciado en las estadísticas que los usuarios hayan recorrido la lista de correos desde la interfaz.

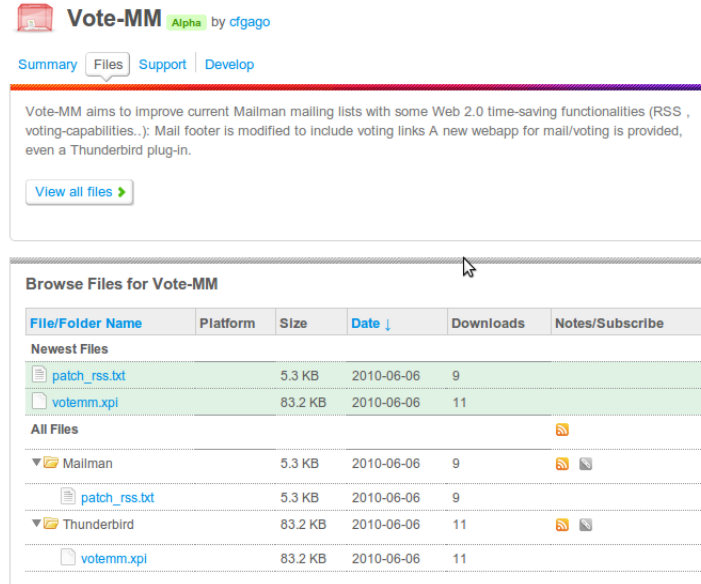


Figura 5.14: Descargas de archivos desde Sourceforge

Otra característica que en el diseño de la aplicación sí parecía muy útil, aunque en el proyecto solamente sea un subsistema utilizado con otro fin, es el RSS. El formato RSS se eligió como anzuelo para generar todo el tráfico posible de entrada a la aplicación, ya que hubiera sido mucho más sencillo y eficaz utilizar Pickle<sup>5</sup> para el envío de la misma. Sin embargo, las estadísticas de uso del RSS demuestran la poca aceptación de la idea.

En esta memoria se ha hablado del éxito incuestionable de un portal de noticias español (*meneame.net*). Sin embargo, este portal tiene un público determinado. Aunque no está pensado para un público en concreto, es cierto que no a todos los usuarios de la red el portal les atrae de la misma forma, dejando a un lado el contenido y los tintes políticos que pueda tener.

En la misma medida podríamos comparar el proyecto en la lista FANS, la aplicación al fin y al cabo debe tener un público que aprecie las ventajas que ofrece, y que esté dispuesto a hacer *click* en los enlaces de votación, y probablemente este no ha sido el

<sup>5</sup>Pickle ofrece aplanamiento de tipos de datos definidos en Python, <http://docs.python.org/library/pickle.html>

caso con Vote-MM.

La publicación del proyecto quizás ha sido demasiado tímida; con el ánimo de no interrumpir el flujo normal de trabajo en FANS quizás no se ha insistido en el uso de la herramienta, o las ventajas, esperando que fueran los usuarios los que descubrieran la información. Tal y como sugirió el director del proyecto al comienzo del mismo, otro medio de comunicación como un blog hubiera ayudado a difundir la herramienta.

## 5.4. Futuro

Como se ha comprobado con la puesta en marcha del proyecto, las votaciones anónimas no son suficientes. Habilitar el registro en la aplicación es una tarea trivial, habría que definir una categoría de grupo de usuarios, y automatizar el registro (existen *snippets*<sup>6</sup> con ese mismo objetivo), porque la aplicación ya está preparada para soportar este esquema.

Con los datos de los usuarios, se tendría que dar valor a cada uno de ellos. Según el número de correos enviados, la relación de lecturas por cada correo del usuario, el número de votaciones ejecutadas, el tiempo que tarda en leer el correo desde que se publica, etc. Estas métricas ayudarían a no dar misma puntuación a cada voto, y dar más poder a los usuarios más activos.

Una vez demostrado el correcto funcionamiento de la aplicación, y contando con la autenticación del usuario, el siguiente paso sería utilizar una lista con un número más elevado de envíos diarios (>10-15), donde sí fuera necesario un algoritmo de extracción de relevancia y donde la priorización fuera necesaria.

Otra tarea importante sería mantener la compatibilidad de los parches de Mailman. La nueva versión de Mailman será estable en un plazo relativamente corto y habría que adaptar el parche a la nueva arquitectura, en especial la inclusión de información en los pie de página, ya que la generación del RSS utiliza la interfaz del Archiver, que no será modificada.

---

<sup>6</sup><http://djangosnippets.org/>

La publicación del proyecto es una fase que no se puede dar por finalizada. Utilizar las redes sociales, *Twitter*, blogs, la inclusión del blog en diferentes planetas relacionados según la tecnología utilizada (Mailman, Django, Thunderbird-dev) lograría una mayor difusión del proyecto. Inclusive la integración con otros productos, como las plataformas de e-learning, foros, etc.

## 5.5. Lecciones aprendidas

Con el proyecto me he enfrentado a muchas tareas para las que la universidad no puede preparar a un alumno. La creación del proyecto en forjas públicas, con la publicación del código y por consiguiente el acceso para los demás usuarios y su retroalimentación, es una experiencia que pone en valor la filosofía del desarrollo de software libre.

En el plano profesional, he mejorado mis conocimientos en la web 2.0. Con el desarrollo de la aplicación, he tenido que profundizar en el uso de DHTML, CSS así como en la gestión de usuarios, formularios, el uso de Javascript para realizar acciones asíncronas, las distintas librerías de AJAX, etc.

El uso del *framework* de desarrollo de aplicaciones Django, me ha permitido un desarrollo rápido y eficaz; la calidad de la información disponible sobre Django, y en especial la disponibilidad de los *snippets* que los usuarios ponen a disposición del público me han permitido profundizar en esta potente herramienta rápidamente. La creación del *Web Service* con JSON y la generación automática de RSS de los diferentes apartados ha sido una tarea sencilla gracias a Django.

La búsqueda de la información necesaria, y la comprensión de la arquitectura de Mailman fueron las tareas que más tiempo me ha requerido. Comprender una arquitectura tan grande hasta reconocer dónde se almacenan los datos de los correos, msg-id, cuerpo de los mensajes, implican recorrer el código de Mailman una y otra vez hasta que se encuentra el flujo normal de un correo desde que es recibido hasta que se envía la copia al Archiver. Por otro lado, el conocimiento aportado por esta tarea no es en absoluto despreciable.

De igual modo, con el plugin para Thunderbird, junto con las tecnologías asociadas

(XUL, Javascript, JSON, arquitectura propia de los plugins) requieren de una cantidad de esfuerzo considerable para un add-on que en teoría no requiere del conocimiento de la arquitectura (Gecko en este caso). La documentación está desfasada, con cada versión la forma de incluir librerías, namespaces, requerimientos de seguridad cambia por lo que ha sido complicado hasta que se ha finalizado el desarrollo de un plugin compatible con las versiones 2 y 3 del cliente de correo.

En la instalación del servidor web, de bases de datos, la evaluación entre las distintas tecnologías para la comunicación de Django con el servidor (Fastcgi, Uwsgi, etc), la inclusión de software de estadísticas (Awstats, Piwik), scripts para tareas de mantenimiento (rotación de logs, backups, etc) afianzan los conocimientos adquiridos a lo largo de carrera de Ingeniería Informática.

# Apéndice A

## Manual de usuario

A continuación se revisará brevemente el uso de la interfaz de la aplicación web y el plugin para Thunderbird. En ambos casos repasaremos las funcionalidades básicas y se detallarán las opciones de configuración.

### A.1. Interfaz web Vote-MM

La pagina principal de la aplicación (figura A.1) muestra las listas disponibles, además de un enlace para promocionar el proyecto *What is Vote-MM System*.

The screenshot shows the main page of the Vote-MM web application. At the top left is a red box icon labeled 'Vote-MM'. To its right is the site's header with the text: 'GSYC Grupo de Sistemas y Telecomunicaciones votemm@votemm.libresoft.es'. Further right are links for 'Home | Contact Us!'. Below the header is a dark navigation bar with 'What is Vote-MM System' and 'Log-in ? | Back'. The main content area is divided into two columns. The left column is titled 'News' and contains three entries: '15th September System is almost ready!', 'April 28, New features added to thunderbird plugin More...', and 'March 15, Thunderbird plugin is available More...'. The right column is titled 'Available lists' and contains the text 'Lists available in the system:' followed by a bulleted list: '• [Votemm Ilist](#) Voted today Archives', '• [TestList Ilist](#) Voted today Archives', and '• [Fans Ilist](#) Voted today Archives'. At the bottom of the page is a footer with 'Gsync © 2010 | Gsync'.

Figura A.1: Pagina principal Vote-MM

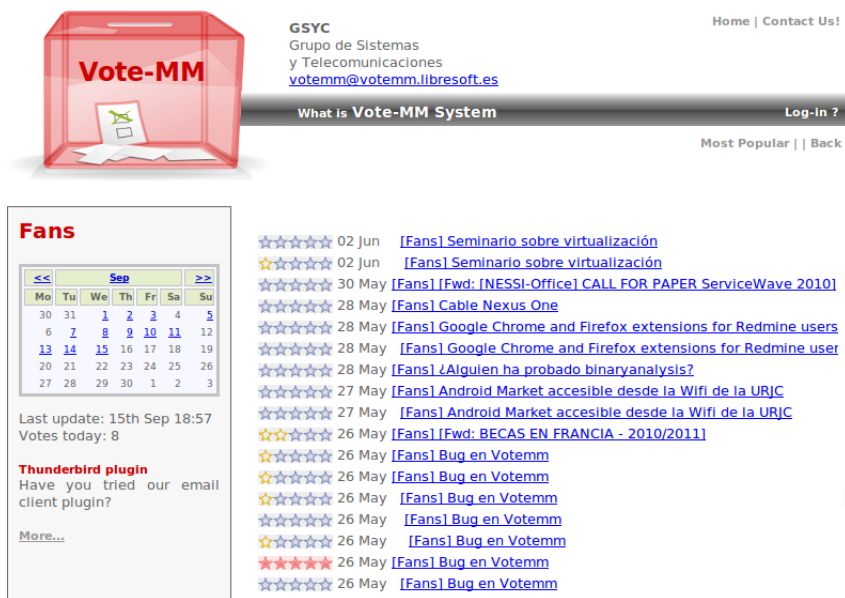


Figura A.2: Correo de la lista

### A.1.1. Flujo de trabajo

El camino más común hasta llegar a un correo de una lista, será eligiendo una lista en la página principal, como muestra la figura A.2. Aquí podemos ver los distintos *threads* de la lista, ordenados de forma descendente según la fecha de publicación el primer correo del *thread*.

A la izquierda tenemos un calendario con enlaces a los correos votados del mes, además de otras noticias de la aplicación.


Las opciones principales para ver los correos más importantes del día, son las vistas *Most voted* (figura A.3) y *Popular* (figura A.4). La primera muestra los correos más votados (con votos positivos), la siguiente utiliza un algoritmo de promoción por el que se eligen una cantidad de correos diaria.

### A.1.2. Configuración de listas de correo

Para añadir nuevas listas al sistema, tenemos que autenticarnos contra el mismo como administradores. A continuación elegimos *Add new list* en el submenú principal (figura A.1), completamos los datos requeridos (figura A.2), y configuramos los parámetros más importantes a la hora de promocionar correos, respectivamente:

- Coeficiente para votos positivos





**GSYC**  
Grupo de Sistemas  
y Telecomunicaciones  
[votemm@votemm.libresoft.es](mailto:votemm@votemm.libresoft.es)

Home | Contact Us!

What is **Vote-MM** System Log-in ?

Most voted | Most Popular | Back

## Most Voted

**Fans**

Calendar of votes


Sep						
Mo	Tu	We	Th	Fr	Sa	Su
30	31	1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	1	2	3

Last update: 15th Sep 18:57

- 2 [Fans] [Open Source test](#)
- 1 [Fans] [Índice de calidad de paquetes en Python](#)
- 3 [Fans] [Favor para Vote-MM](#)
- 1 [Fans] [¿con el COPYING basta para que tu código sea GPL?](#)
- 1 [Fans] [Sigue las noticias de LibreSoft en tu Android](#)

Gsync © 2010 | Gsync


Figura A.3: Correos más votados en el día



**GSYC**  
Grupo de Sistemas  
y Telecomunicaciones  
[votemm@votemm.libresoft.es](mailto:votemm@votemm.libresoft.es)



Home | Contact Us!

What is **Vote-MM** System Log-in ?

Most voted |  | Back

## Top voted in Fans

0 hits 15 Sep [Fans] [Seminarío modelos de negocio sw libre EO!](#) **Read**



0   0

---

15 Sep [Fans] [Oferta de trabajo en la UEL](#) **Read**

<http://jobs.uel.ac.uk/Vacancy.aspx?ref=075A2010>



0 hits Un saludo,  
Dani.

0   0



--

Daniel Izquierdo Cortázar | Libre Software Engineering Lab (GSyC)  
[dizquierdo\\_at\\_gsync.es](mailto:dizquierdo_at_gsync.es) | Universidad Rey Juan Carlos  
<http://libresoft.es> | Edif. Departamental II - Despacho 118  
 Telf: (+34) 91 488 8523 | c/Tulipán s/n 28933 Móstoles (Madrid)

0 hits 15 Sep [Fans] [Fwd: Call for Replications - RESER 2011 \(co-located with ICSE\)](#) **Read**

0   0

8 hits 15 Sep [Fans] [Favor para Vote-MM](#) **Read**

2   -1

5 hits 14 Sep [Fans] [Open Source test](#)

Figura A.4: Lista de correos elegidos como principales

Figura A.5: Configuración: Configuración parámetros

- Coeficiente para votos negativos
- Número máximo de votos diarios a cada usuario (para cada lista del sistema)
- Número máximo de votos de cada usuario para cualquier correo.

Cuadro A.1: Configuración: Add new list

Cuadro A.2: Configuración: Rel- lenar datos

## A.2. Plugin Thunderbird

El plugin Vote-MM añade una nueva columna a Thunderbird con la que se muestran los votos positivos/negativos de cada correo, como se muestra en la figura A.8



Figura A.6: Actualizamos el RSS



Figura A.7: Mensaje de actualización

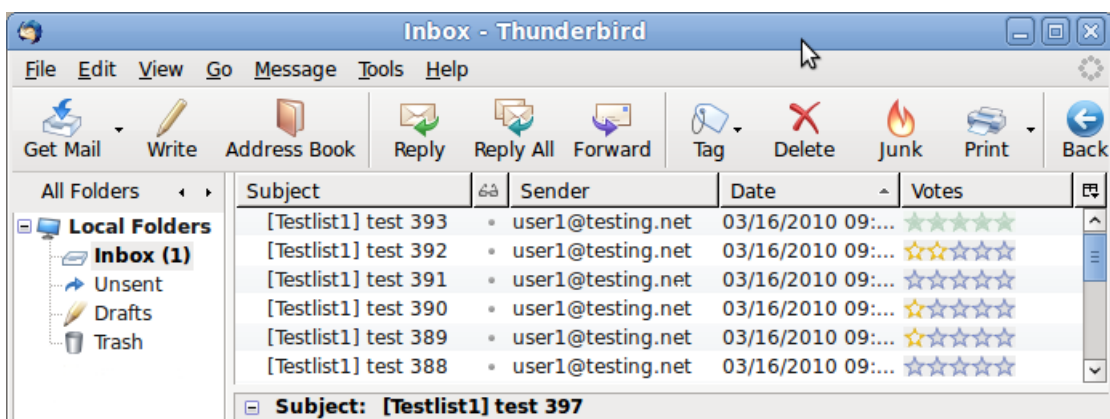


Figura A.8: Cliente de correo thunderbird con plugin Vote-MM instalado

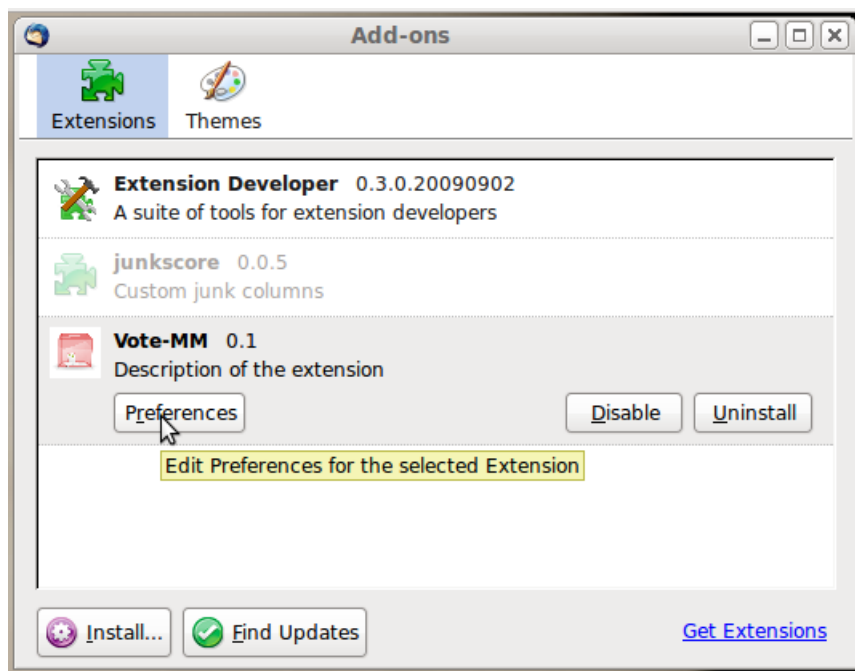


Figura A.9: Menú Add-on de Thunderbird

### A.2.1. Instalación

La instalación es sencilla, descargamos el plugin de la pagina de Sourceforge<sup>1</sup> y desde el menú del cliente instalamos la aplicación. Primero deben configurarse los parámetros requeridos (ver figura A.10), para ello, en el menú Add-on (figura A.9) en las preferencias del plugin, añadimos:

- Service URL (usually <http://votemm.libresoft.es/json/>)
- Update interval: Intervalo de actualización en segundos.
- List name: Nombre de la lista, tal y como aparece en la aplicación web.

Además, debemos habilitar la columna en el navegador, como muestra la figura A.11.

### A.2.2. Flujo de trabajo

La forma de realizar votaciones es haciendo click sobre el correo seleccionado, en la columna *Votes*. Si cuando hacemos click, pulsamos a la vez la tecla Control (crt) el

<sup>1</sup><http://sourceforge.net/projects/vote-mm/files/>

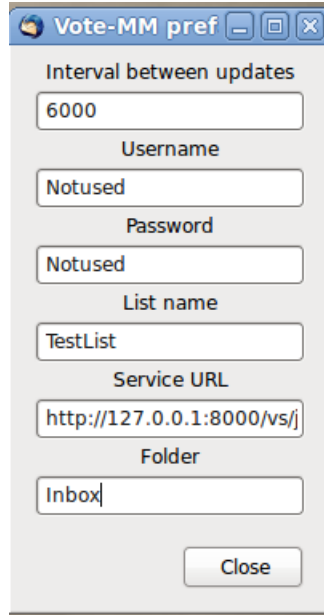


Figura A.10: Opciones de configuración

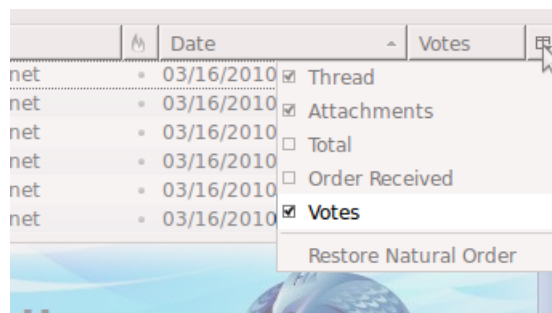


Figura A.11: Elegimos habilitar la columna *Votes*

voto enviado será negativo. Los correos ya votados cambiarán el color a verde, como se muestra en la figura A.8

Si existiera algún problema con el proceso de actualización de votos, saldría un mensaje emergente en la esquina inferior derecha, donde usualmente veremos *Votemm*.