

Actas de las III Jornadas de Software Libre de la Universidad de Cádiz

Manuel Palomo Duarte
J. Rafael Rodríguez Galván
Coordinadores



© 2006 Universidad de Cádiz
Oficina de Software Libre
<http://softwarelibre.uca.es>
© 2006 Servicio de Publicaciones de la Universidad de Cádiz
© De cada artículo, sus autores

Edita: Servicio de Publicaciones de la Universidad de Cádiz
C / Doctor Marañón, nº 3, 11002 Cádiz
<http://www.uca.es/publicaciones>
ISBN10: 84-9828-055-9
ISBN13: 978-84-9828-055-5

Comité organizador

Coordinadores

Manuel Palomo Duarte[†]

J. Rafael Rodríguez Galván^{◇*}

Secretario

Gerardo Aburruzaga García^{†**}

Colaboradores

José Manuel Frías Carnero[•]

Juan Carlos González Cerezo^{**}

Ignacio Montoya García^{*}

Francisco Palomo Lozano[†]

Comité científico-técnico

Gerardo Aburruzaga García^{†**}

Maria del Carmen de Castro Cabrera[†]

Juan José Domínguez Jiménez[†]

Antonia Estero Botaro[†]

Fernando Fernández Palacín[‡]

Israel Herráiz Taberner^{*}

Inmaculada Medina Bulo[†]

Francisco Palomo Lozano[†]

Manuel Palomo Duarte[†]

Carlos Rioja del Río[†]

J. Rafael Rodríguez Galván^{◇*}

Mercedes Ruiz Carreira[†]

[†] Departamento de Lenguajes y Sistemas Informáticos, Universidad de Cádiz

[‡] Departamento de Estadística e Investigación Operativa, Universidad de Cádiz

^{*} Grupo Libresoft, Universidad Rey Juan Carlos

[◇] Departamento de Matemáticas, Universidad de Cádiz

^{*} Oficina de Software Libre, Universidad de Cádiz

[•] Área de Informática, Universidad de Cádiz

Índice general

I Ponencias	3
1. Software libre para prácticas en la E.S.I. de Sevilla. Una experiencia educativa	5
2. Navegación web Segura	17
3. Implantación de LDAP como sistema de autenticación centralizada	31
4. MOVICUO: Comunicaciones móviles y software libre para la ubicuidad	45
5. Ruby on Rails y otros entornos MVC de licencia libre: Un análisis práctico	59
6. Implantando GNU/Linux: Una visión desde un Ayuntamiento	71
7. TOMAS ³ : Towards an Open Management Architecture for Systems, Software and Services	81
II Proyectos fin de carrera	97
8. Intérprete de diagramas de flujo.	99
9. Plataforma libre de desarrollo de algoritmos para el proyecto Pelican	105
III Talleres	111
10. Edición de audio digital con Audacity	113
11. Introducción a la creación y tratamiento de imágenes digitales mediante <i>Gimp</i>	119
12. Creación de presentaciones en DVD con dvd-slideshow	127
Índice de Autores	133

Índice de figuras

1.1. Diseño lógico de bloques de CESIUS.	6
1.2. Ejemplo de ejecución del traductor.	11
1.3. Aspecto de la pantalla del depurador de CESIUS.	13
4.1. Estructura del sistema	47
4.2. Pila de protocolos del enlace PPP	49
4.3. Activación del contexto PDP con comandos AT+	50
4.4. Aspecto inicial de GNOME-GPRS	52
4.5. Aspecto al conectarse	52
4.6. Gráfica resultante para el Caso 1	54
4.7. Gráfica resultante para el Caso 2	54
4.8. QoS solicitada y Qos minima negociadas	54
4.9. Gráfica resultante para el Caso 3	56
4.10. Gráfica resultante para el Caso 4	56
5.1. Esquema típico de MVC	61
5.2. Ejemplo de recuperación y modificación de datos con ActiveRecord	65
5.3. Formato de URL	65
5.4. Ejemplo de definición de acción	65
7.1. Sistema Gestor de Configuraciones	87
7.2. Sistema Gestor de Instalaciones	89
8.1. captura de pantalla del intérprete	102
8.2. captura con watch(vigía) de variables	102
9.1. Arquitectura software de la biblioteca paralela integrada para el cálculo numérico científico.	107
9.2. Esquema genérico del funcionamiento del servicio web	107
10.1. Audacity trabajando en modo multipistas digital.	115
10.2. Audio antes y después del tratamiento con Audacity.	115
11.1. Interfaz gráfica de <i>Gimp</i>	121
12.1. Ejemplo de código para mostrar una serie de fotos	129
12.2. Programa Slideshow Creator en funcionamiento.	129

Prólogo

En los últimos años, las universidades españolas (siendo pionera entre ellas la Universidad de Cádiz) están tomando conciencia de la importancia de las estrategias relacionadas con el software y el conocimiento libres como medio hacia los fines últimos recogidos en sus estatutos, racionalizando gastos, subrayando aspectos éticos como el conocimiento, la libertad y la colaboración, fomentando el uso innovador e independiente de las tecnologías de la información y las comunicaciones para la educación, investigación y gestión y destacando el papel de la universidad como motor de conocimiento y tecnología en su entorno.

Conscientes de todo ello, la Oficina de Software Libre de la Universidad de Cádiz, con el respaldo del Área de Informática y el apoyo del Departamento de Lenguajes y Sistemas Informáticos, ha apostado desde sus orígenes por la celebración de estas jornadas anuales, enfatizando su papel divulgativo y acercando el mundo de la empresa a la comunidad universitaria.

Manteniendo esta filosofía, ha llegado el momento de dar un salto de calidad. Así, en esta tercera edición de las Jornadas se ha realizado un esfuerzo por enriquecerlas, abriéndolas a la participación mediante la posibilidad de presentar ponencias, talleres o proyectos de fin de carrera que, tras un riguroso proceso de revisión, están recogidos en este libro de actas para su distribución a los asistentes.

La organización de las III Jornadas de Software Libre ha supuesto un importante esfuerzo que no hubiera sido posible sin la colaboración de un nutrido grupo de personas, entre las cuales se encuentran los miembros del comité organizador y del comité científico-técnico. En especial, nos gustaría expresar nuestra gratitud a Gerardo Aburrizaga García, secretario de las mismas. Sin ninguna duda, nada hubiera sido posible sin ellos. Asimismo, nos gustaría expresar nuestro agradecimiento a los autores de los trabajos recogidos en las presentes páginas, a las empresas colaboradoras y, en general, a todas aquellas personas que, desarrollando o apoyando el desarrollo de software y conocimiento libres, han hecho posibles estas jornadas.

Manuel Palomo Duarte
Departamento de Lenguajes
y Sistemas Informáticos
de la Universidad de Cádiz.

J. Rafael Rodríguez Galván.
Departamento de Matemáticas.
Director de la Oficina de Software Libre
de la Universidad de Cádiz.

Parte I

Ponencias

Software libre para prácticas en la E.S.I. de Sevilla. Una experiencia educativa

José Julio Hernández Fernández

jjhf@jjhf.com

Dpto. Ing. de Sistemas y Automática. Universidad de Sevilla.

Manuel Ruiz Arahal

arahal@esi.us.es

Dpto. Ing. de Sistemas y Automática. Universidad de Sevilla.

Resumen

En este artículo se muestra una aplicación del software libre en la docencia. Se trata de un conjunto de programas que permiten a los alumnos de Ingeniería Industrial realizar prácticas de programación en un nivel bajo próximo al código binario.

CESIUS es una computadora inexistente que sirve para ilustrar muchos aspectos de las computadoras reales. El diagrama de la misma es muy simple y ha sido creado a partir del esquema de Von Neumann para la computadora de programa almacenado. Con ella, los alumnos de la asignatura de

"Fundamentos de Informática" pueden familiarizarse con los conceptos básicos del código binario y ensamblador, programando, depurando sus propios programas, y siguiendo la ejecución de los mismos de una manera gráfica muy didáctica.

1.1. Introducción

En el plan de estudios de la titulación de Ingeniería Industrial de la Universidad de Sevilla aprobado en el año 1998 aparece en el primer curso la asignatura de "Fundamentos de Informática", dotada con 9 créditos del total de los 72 de los que consta el

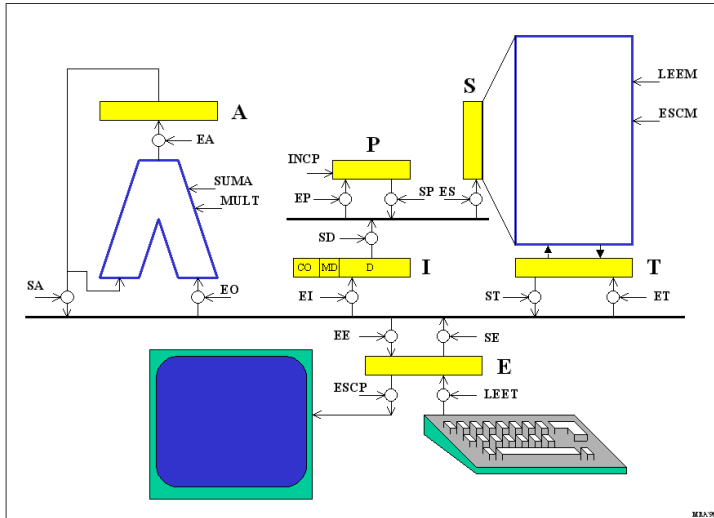


Figura 1.1: Diseño lógico de bloques de CESIUS.

primer curso, dentro de los 390 de la carrera completa (primer y segundo ciclo).

El bajo número de créditos es una restricción muy importante, que obligó en su momento a realizar un plan docente para la asignatura con un delicado equilibrio. Se optó por proporcionar una visión lo más amplia posible, abarcando aspectos de bajo nivel [1] [2], de algoritmia [3] y de programación [4]. A fin de poder profundizar más en los aspectos considerados más importantes ha sido preciso sacrificar la presentación de otros. En particular en la programación del curso se dedica muy poco tiempo a las descripciones de equipos [5] y de los sistemas [6] pues son conocimientos que se adquieren hoy en día antes de llegar a la universidad con el uso de equipos domésticos o en centros de enseñanza.

La herramienta que se muestra en el presente trabajo es utilizada en la realización de prácticas de laboratorio durante el primer cuatrimestre, el cual se dedica a presentar el funcionamiento básico de la computadora digital.

La computadora CESIUS no existe como un conjunto de circuitos: es una computadora virtual, basada como se ha comentado anteriormente en el modelo de Von Neumann. Se puede ver su diseño en la figura 1.1. Existen, en cambio, un conjunto de programas capaces de simular el funcionamiento de una computadora CESIUS. Este hecho, en lugar de representar una desventaja, proporciona la oportunidad de que la máquina sea usada por cualquier persona en cualquier parte del mundo, sin más condición que el disponer de una computadora

tipo PC o de cualquier otro tipo con mínimos requisitos.

1.2. Visión general

La computadora CESIUS consta de los siguientes bloques lógicos:

- UAL: unidad aritmético-lógica
- UC: unidad de control
- UM: unidad de memoria
- UES: unidad de entrada-salida

y de los siguientes registros:

- A: acumulador, 16 celdas
- P: contador de pasos del programa, 11 celdas
- I: registro de instrucción, 16 celdas
- S: selector de memoria, 11 celdas
- T: tampón de memoria, 16 celdas
- E: tampón de entrada-salida. 16 celdas

El funcionamiento de la máquina queda perfectamente definido conociendo el juego de instrucciones, consistente en las 16 operaciones básicas que se indican a continuación:

- ALT: Parar
- ALM: Almacenar
- CAR: Cargar

- ESC: Escribir Número
- ECA: Escribir Carácter
- LEE: Leer Número
- LCA: Leer Carácter
- SUM: Sumar
- RES: Restar
- MUL: Multiplicar
- DIV: Dividir
- MOD: Módulo
- SAL: Salto
- SAN: Salto si A es negativo
- SAC: Salto si A es cero
- SAP: Salto si A es positivo

La computadora es capaz de realizar operaciones directas e indirectas. En el modo directo, el operando de la instrucción es interpretado como la dirección sobre la que se ha de trabajar. En el modo indirecto el operando es tomado como la dirección que contiene la dirección sobre la que se ha de trabajar. La computadora puede programarse por medio de un lenguaje ensamblador de dos pasos llamado LS-2. Las sentencias en LS-2 constan de: etiqueta (separada por dos puntos), mnemónico de instrucción, indicador de modo (letra I) y operando. Por ejemplo:

```
X:  ESP 1
    CAR X
M:  LEE I PUN
```

Las pseudoinstrucciones son órdenes para el traductor que facilitan la programación. Éstas son:

- **ORG:** Instrucción (opcional) que indica el punto inicial de carga en memoria del programa.
- **ESP:** Permite dejar espacios vacíos entre instrucciones
- **CTE:** Para definir constantes
- **DRE:** Deja un espacio con el valor representado por la etiqueta
- **FIN:** Indica el fin del programa

El simulador de CESIUS se ha dividido en varios programas con objeto de facilitar su uso. Cada programa se ocupa de una tarea concreta, de forma que entre todos se satisfagan las necesidades que se plantean.

A continuación se proporciona una breve descripción de cada uno de los programas. En apartados posteriores se explicará detalladamente la forma de uso de cada uno.

Traductor: es un ensamblador de dos pasos que produce archivos ejecutables a partir de archivos que contienen el programa en lenguaje simbólico LS2.

Ejecutor: permite simular la ejecución de un programa, previamente traducido en CESIUS. Los resultados obtenidos en pantalla son los que produciría dicha computadora si existiese realmente.

Depurador: permite ejecutar los programas instrucción a instrucción, mostrando al mismo tiempo el contenido de los registros y la memoria de CESIUS. De este modo es posible buscar errores de programación.

Los programas están pensados para ser usados en entornos no gráficos, esto es, en

línea de órdenes. La salida en pantalla consiste en texto únicamente. Esta característica los convierte en fácilmente transportables.

1.3. El traductor

El traductor es un programa que crea archivos ejecutables de CESIUS a partir de archivos que contienen un programa en LS2. Para ello realiza las mismas operaciones que un ensamblador de dos pasos.

El archivo que contiene el programa en LS2 recibe el nombre de archivo fuente. El archivo que contiene el programa traducido es llamado ejecutable de CESIUS y abreviadamente archivo ECE. Este archivo ejecutable es posteriormente usado por el ejecutor o por el depurador, que simulan el comportamiento de CESIUS.

Para el correcto funcionamiento del traductor es preciso que el archivo fuente cumpla ciertos requisitos que se comentan a continuación.

1.3.1. Escritura del fuente

El archivo fuente puede crearse con cualquier programa de redacción de textos, observando las normas siguientes:

- La pseudoinstrucción **ORG** ha de aparecer en primer lugar o no aparecer en absoluto.
- Las etiquetas se definen situándolas a la izquierda de la sentencia, separadas de los códigos por dos puntos.
- Los operandos han de estar separados por espacios o tabulaciones del resto.

- El indicador de direccionamiento indirecto ha de estar separado por espacios del resto de códigos.
- Se pueden introducir comentarios después del operando, separados de éste por el signo de apóstrofe.
- Ha de existir una pseudoinstrucción FIN.
- Toda etiqueta usada ha de estar definida.

Como se ve, las restricciones son muy suaves, lo cual permite cierta despreocupación a la hora de escribir el código. En el cuadro 1.1 se muestra un programa correctamente escrito. Las etiquetas aparecen alineadas por comodidad para el lector, pero esto no es necesario.

Se observa que el programa calcula la media de dos números que se leen de teclado. Este ejemplo servirá para ilustrar el manejo de cada una de las partes del simulador de CESIUS, comenzando por el uso del traductor que se comenta a continuación.

1.3.2. Modo de uso

Para usar el programa traductor es preciso iniciar una sesión en línea de órdenes, y escribir `traducir` y a continuación el nombre del archivo fuente.

Es preciso indicar el nombre completo del archivo fuente, incluyendo la extensión. Si no se proporciona esta información el programa pregunta por pantalla dicho nombre. Si el archivo proporcionado no existe en el directorio actual aparece un mensaje de error.

Si la traducción se realiza sin fallos el programa traductor muestra en pantalla cierta información relativa al programa que se está traduciendo incluyendo la lista de etiquetas y del valor representado. Esta información ayuda a determinar si el programa ha sido correctamente traducido.

1.3.3. Mensajes de error

En la escritura del código fuente pueden deslizarse errores que causarán un mal funcionamiento del traductor. El programa traductor detecta los siguientes errores:

Código no reconocido. Aparece cuando se escribe mal una instrucción o pseudoinstrucción, como por ejemplo, escribir SUN en lugar de SUM. También puede deberse a que se olviden los dos puntos que separan la etiqueta de la operación.

Etiqueta no definida.

Operando fuera de rango. Se ha utilizado un número que CESIUS no puede utilizar debido a las limitaciones impuestas por la memoria y el ancho de palabra. También puede deberse el error a utilizar un número negativo con ORG o ESP

Operando no numérico incorrecto. Se ha utilizado una etiqueta como operando de ORG, ESP, CTE en lugar de un número.

Falta operando. Indica que se ha olvidado consignar el operando en la sentencia.

No hallado FIN.

	ORG	0	
	SAL	INI	'inicio instrucciones
A:	ESP	1	'sumando
B:	ESP	1	'otro sumando
MEDIA:	ESP	1	'media
DOS:	CTE	2	'constante
INI:	LEE	A	'lecturas
	LEE	B	
	CAR	A	
	SUM	B	'operaciones
	DIV	DOS	
	ALM	MEDIA	
	ESC	MEDIA	'escritura resultado
	ALT		
	FIN		

Cuadro 1.1: Ejemplo de programa en LS-2.

ORG fuera de sitio. Esta pseudoinstrucción ha de ser la primera en aparecer o no aparecer en absoluto.

1.3.4. Archivos ECE

El resultado de la traducción es un archivo ejecutable de CESIUS (ECE). Es imprescindible conocer que este archivo sólo puede ejecutarse a través del simulador de CESIUS, es decir, no es un archivo ejecutable del sistema operativo anfitrión sobre el que se realiza la simulación.

El archivo ECE contiene el programa tal y como debe colocarse en la memoria de CESIUS para su ejecución. El archivo es binario, por lo que no es posible interpretar su contenido a menos que se sepa la forma en que ha sido construido. Esto significa en particular que no es posible utilizar un editor de textos para modificarlo.

El tamaño del archivo ECE depende de la cantidad de memoria que use el programa

que se ha traducido, siendo el máximo 8.1 kilocaracteres.

El programa traductor produce siempre el archivo **prog.ece**. Si se desea se puede renombrar este archivo, para de este modo poder tener en el mismo directorio varios ejecutables de CESIUS.

1.3.5. Ejemplo de uso

Supongamos que el programa del cuadro 1.1 ha sido escrito y almacenado en el archivo **media.ls2**. En la figura 1.2 se muestra el resultado de la ejecución de la orden **traducir**.

Además, debe aparecer el archivo **prog.ece** en el directorio de trabajo.

El archivo **traducir** ha de estar colocado en el directorio de trabajo o en alguno de los directorios donde el sistema operativo busca los programas ejecutables. En caso contrario se producirá un mensaje de error.

```

/home/jjhf/CESIUS@lab01>traducir media.ls2

Traductor CESIUS : LS2 -> ECE
M.R. Arahal, Jose Julio H.F., 1999
E.S.I. Univ. Sevilla, España.
Traduciendo media.ls2
Línea Etiqueta Valor Propos Operando
0
1
2          A      1   SAL   INI
3          B      2   ESP   1
4      MEDIA      3   ESP   1
5          DOS     4   CTE   2
6          INI     5   LEE   A
7
8          7      6   LEE   B
9
10         8      7   CAR   A
11         9      8   SUM   B
12        10     9   DIV   DOS
13        11    10   ALM   MEDIA
14        12    11   ESC   MEDIA
15        13    12   ALT
16        14    13   FIN
Creado archivo ECE. Fin del traductor.

/home/jjhf/CESIUS@lab01>_

```

Figura 1.2: Ejemplo de ejecución del traductor.

1.4. El ejecutor

El ejecutor es el programa encargado de simular la ejecución de un programa de CESIUS, proporcionando los mismos resultados que se obtendrían si se tuviese la máquina construida realmente.

El ejecutor toma el programa CESIUS del archivo ECE y realiza las transferencias elementales entre registros correspondientes a cada instrucción. De este modo obtiene los resultados a partir de los datos suministrados.

Los registros de la UCP son variables internas del programa ejecutor que el usuario no puede conocer. El programa CESIUS se ejecuta de forma oculta al usuario, el cual solo puede introducir por teclado los valores correspondientes a las operaciones de lectura que el programa realice. La pantalla muestra los valores que el programa escri-

be.

1.4.1. Modo de uso

Para ejecutar un programa CESIUS con el simulador basta con escribir

```
ejecutar prog.ece
```

El archivo **ejecutar** es el programa simulador, y debe estar colocado en el directorio de trabajo o en alguno de los directorios donde el sistema operativo busca los programas ejecutables. En caso contrario se producirá un mensaje de error.

Para ejecutar un programa CESIUS incluido en un archivo con otro nombre, como por ejemplo, **otro.ece**, se escribe

```
ejecutar otro.ece
```

1.4.2. Mensajes de error

Se pueden presentar errores durante la ejecución del programa. Una causa de ellos puede ser el intentar acceder a una zona de memoria de CESIUS que no está en los límites de la memoria reservada por el programa.

A continuación se muestra la lista de mensajes de error que el programa ejecutor reconoce:

Desbordamiento en la UAL. Ocurre cuando el resultado de una operación es un número que no puede ser almacenado en el acumulador.

Acceso a memoria fuera de rango. Se ha intentado leer o escribir en la memoria en una zona no reservada por el programa.

Instrucción no válida. El contenido del registro I no es una instrucción. Esta situación puede ocurrir cuando el registro P invade la zona de datos o si se ha escrito un dato sobre una instrucción.

Salto no permitido. Se ha intentado realizar un salto a una dirección fuera del rango reservado por el programa; es decir, P ha tomado un valor fuera de dicho rango.

1.4.3. Ejemplo de uso

Continuando con el ejemplo del apartado anterior (véase el cuadro 1.1) la ejecución se pone en marcha mediante

```
ejecutar prog.ece
```

Se muestra en pantalla el mensaje

```
introduzca dato:
```

como consecuencia de haber ejecutado una instrucción de lectura de CESIUS, correspondiente a la sentencia **LEE A** del código fuente. Es preciso introducir por teclado un valor numérico y presionar la tecla Intro. De la misma manera se procede para el dato B, debiendo el usuario proporcionar un número.

Finalmente aparece en pantalla el resultado de una operación de escritura de CESIUS

```
escritura : 18
```

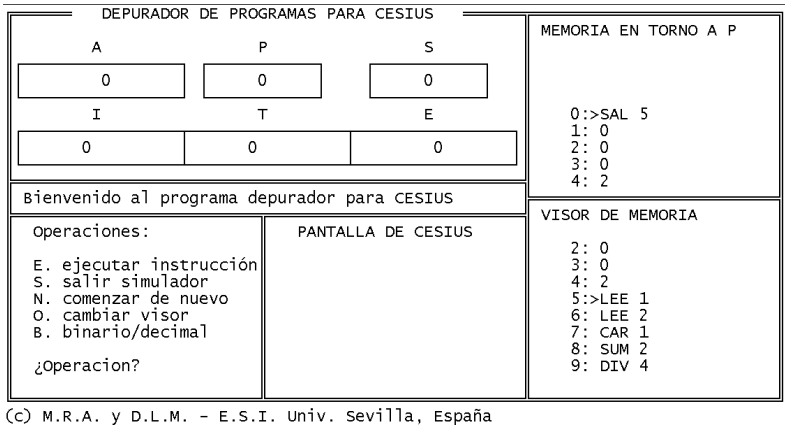
y la indicación de que el programa CESIUS ha llegado a su final

```
Fin del programa CESIUS
```

1.5. El depurador

El programa depurador permite ejecutar los programas de CESIUS instrucción a instrucción, mostrando al mismo tiempo el contenido de los registros y la memoria de CESIUS. La utilidad del depurador es doble: por una parte sirve para buscar errores en el programa, y por otra ayuda a comprender el funcionamiento de CESIUS.

El depurador es un programa parecido al ejecutor, que se ha comentado anteriormente. Ambos se basan en la misma idea: realizar las transferencias elementales entre registros correspondientes a cada instrucción del programa CESIUS. El depurador, además, permite observar el contenido de tales registros. También incorpora otras



(C) M.R.A. y D.L.M. - E.S.I. Univ. Sevilla, España

Figura 1.3: Aspecto de la pantalla del depurador de CESIUS.

funciones, como la posibilidad de abortar la ejecución o de inspeccionar la memoria de CESIUS.

1.5.1. Información en pantalla

La pantalla del depurador se divide en varias zonas como muestra la figura 1.3. En la superior se muestran los registros A, I, P, S, T y E y la pantalla de CESIUS. A la derecha se muestra la memoria en dos partes. Finalmente, en la zona inferior se muestran mensajes del depurador y se piden las opciones al usuario.

La pantalla cambia de estado cada vez que se realiza una operación del depurador. En particular, cada vez que se ejecuta una instrucción de CESIUS se ha de refrescar el contenido de los registros, la pantalla de CESIUS y la memoria. Es preciso tener en cuenta que la actualización de la pantalla se realiza justo al terminar la ejecución de una instrucción, por lo que el registro P apunta a la instrucción siguiente, aunque

ésta no ha sido leída aún de memoria.

La zona de memoria está dividida en dos partes. En la primera de ellas se muestran los registros que están próximos al apuntado por P, es decir, la zona cercana a la instrucción que se está ejecutando.

La parte inferior muestra la memoria próxima a un valor seleccionado por el usuario llamado objetivo del visor de memoria. Cambiando este valor es posible recorrer la memoria entera para su inspección. La posición del objetivo del visor se señala con un signo >. Las direcciones de memoria se indican en decimal comenzando la numeración desde cero de arriba a abajo.

La información contenida en los registros se muestra en forma de números decimales para facilitar la observación. Si se trata de una instrucción se muestra el código mnemónico correspondiente.

1.5.2. Opciones

El depurador funciona interactivamente, esto es, solicita a cada paso una opción al usuario, realiza la operación correspondiente y vuelve a pedir indicaciones al usuario.

Las operaciones que el depurador es capaz de realizar son las siguientes:

- E. Ejecutar siguiente instrucción
- S. Salir del depurador
- N. Comenzar de nuevo la ejecución del programa CESIUS, sin terminar la actual
- O. Cambiar objetivo del visor de memoria
- B. Cambiar base de representación de binario a decimal y viceversa

1.5.3. Mensajes de error

Pueden aparecer errores durante la ejecución del programa ECE, los mensajes de error que se generan son los mismos que proporciona el ejecutor y que han sido comentados anteriormente.

1.6. Acceso al simulador

Los programas necesarios para ejecutar el simulador pueden obtenerse a través de la red Internet. Los autores (Manuel Ruiz Arahal, José Julio Hernández Fernández y Daniel Limón Marruedo) autorizan el uso del programa bajo los términos de la GNU/GPL2.

El manual para el uso del simulador también está disponible en Internet y puede ser reimpresso con fines no comerciales, citando

siempre la fuente de donde ha sido obtenido.

Existe una página de Internet que contiene información actualizada sobre CESIUS y el simulador, cuya dirección es:

<http://www.esi2.us.es/~arahal/CESIUS/cesius.html>

1.7. Conclusiones

La programación de una herramienta educativa como CESIUS en un lenguaje estándar como C nos ha permitido disponer de una aplicación fácilmente transportable, ejecutable en entornos muy variados, y sin más requisitos mínimos que la existencia de un compilador de C. Al haber sido ideada para su ejecución sobre software libre y en línea de órdenes nos ha permitido una gran flexibilidad en su despliegue a la hora de ponerla a disposición de los alumnos, ya que con un simple telnet o SSH a los servidores del Centro de Cálculo disponen de todos los elementos necesarios para realizar sus prácticas. El nivel de aceptación de la aplicación por parte de los alumnos que en los últimos años han sido usuarios de la misma nos hace apostar por este tipo de desarrollos para futuros proyectos educativos, y nos permite simultáneamente servirles a muchos de ellos como primera toma de contacto con el software libre.

Bibliografía

- [1] Fernández, G. y Sáez Vacas, F. *Fundamentos de los ordenadores*. E.T.S.I. de Telecomunicación, Madrid, 1985.
- [2] de Miguel Anasagasti, P. *Fundamentos de los computadores*. Paraninfo, Madrid, 1990.
- [3] Brassard, G. y Bratley, P. *Fundamentos de algoritmia*. Prentice Hall, 1997.
- [4] Prieto, A., Lloris, A. y Torres, J.C. *Introducción a la informática*. McGraw-Hill, Madrid, 1995.
- [5] Beekman, G. *Computación e informática hoy*. Addison-Wesley Iberoamericana, Wilmington, Delaware, 1995.
- [6] Fernández, G. *Conceptos básicos de arquitectura y sistemas operativos. Curso de ordenadores*. Sistemas y servicios de comunicación, Madrid, 1994.

Navegación web Segura

Free Security Suite. Una solución basada en software libre¹

José Briebea Sánchez

jose@briebea.com

Departamento de Informática. Universidad de Extremadura.

Montaña Castuera Toro

moncastoro@yahoo.es

Departamento de Informática. Universidad de Extremadura.

José Luis Gonzalez Sánchez

jlgls@unex.es

Departamento de Informática. Universidad de Extremadura.

Palabras Clave: Software Libre, Código de fuente abierta, Seguridad, Privacidad, Suites.

Resumen

La situación actual de inseguridad en la navegación por Red lleva a que los usuarios inexpertos sean blanco perfecto para todo tipo de engaños y ataques. FSS proporciona

a los usuarios las herramientas que les permiten llegar a niveles de seguridad aceptables sin tener que invertir grandes cantidades en soluciones propietarias. El hecho de contar con un desarrollo basado en software libre hace que sea una solución flexible, renovable y fiable para la comunidad.

¹Este proyecto ha sido patrocinado en parte por la Junta de Extremadura (Expediente 2PR03A09) <http://gitaca.unex.es/agila>

2.1. Antecedentes

En el mundo en que vivimos las comunicaciones se están globalizando e Internet ha dejado de ser coto privado de expertos informáticos entrando en el Medio miles, millones, de usuarios inexpertos. El perfil del usuario medio, hoy por hoy, no tiene nada que ver con el de hace unos años. En ese tiempo, las conexiones eran más restringidas y conectarse a la red requería unos conocimientos avanzados de los servicios que se fueran a utilizar tales como gopher, ftp, finger, etc. En los últimos años se han ido perfeccionando los sistemas de navegación web, permitiendo integrar diferentes servicios en los mismos programas y haciendo su uso accesible al público en general.

La popularización de las conexiones de banda ancha en los hogares ha propiciado un claro salto cuantitativo de los usuarios que se conectan a la Red. Sin embargo, no se ha producido una concienciación de los usuarios en términos de seguridad, para enfrentarse a un medio “hostil” como es la Red. El número de usuarios que se han incorporado al Medio en los últimos tiempos y no sabe a lo que se enfrenta es realmente alto. Los profesionales de la informática debemos ser capaces de concienciarles de la necesidad de usar una serie de técnicas que les permitan conectarse y navegar por la Red con unos niveles de seguridad altos.

Todos los días aparecen nuevos problemas en la Red: virus, *phishing*, protección de menores, acceso ilegítimo a sistemas... Aplicando unas mínimas medidas de seguridad podrían paliarse muchos de estos

problemas. Protegerse es responsabilidad del usuario; si él no lo hace difícilmente lo podremos hacer nosotros.

Generalmente, las soluciones integrales de seguridad para la navegación por la Red son propietarias. Este hecho en sí no las hace malas pero les hace estar en desventaja frente a las que están desarrolladas de manera “libre”. Además, están enfocadas a un sector del mercado (el que consume sistemas operativos propietarios) dejando a un lado a todos los usuarios “libres” del mundo.

2.2. Incidencias de seguridad en la navegación web

El objetivo principal de FSS es que un usuario inexperto pueda controlar, de manera fácil, un sistema de navegación web segura. De esta forma, se proporciona al usuario los instrumentos necesarios para obtener un sistema “seguro” de una forma libre y, en este caso, gratuita.

La Web esta viva. Cada día surgen nuevos problemas susceptibles de ser explotados por usuarios especialmente “hábiles” en su propio beneficio. Por lo tanto, los sistemas de seguridad deben evolucionar ajustándose a la realidad cambiante de la Red. Si la inseguridad crece, nuestras precauciones deben crecer con ella.

En esta sección veremos las diferentes incidencias de seguridad más comunes y posteriormente veremos qué soluciones propone FSS para cada una ellas. La ISO [2], en la norma 7498, define la seguridad informática como:

«Una serie de mecanismos que minimizan la vulnerabilidad de bienes y recursos, donde un bien se define como algo de valor y la vulnerabilidad se define como la debilidad que se puede explotar para violar un sistema o la información que contiene.»

En España la seguridad informática está legislada a través del Real decreto-Ley 14/1999 (17/Sept) [3], por la Orden ministerial 21/Feb/2000 que aprueba el Reglamento de acreditación de prestadores de servicios de certificación y algunos productos de firma, y, además, por otras leyes como son las del DNI electrónico o la Ley de firma electrónica.

Centrándonos en la suite que nos ocupa, veremos cómo FSS va a permitir a un usuario inexperto, siguiendo una serie de pautas o pasos, disfrutar de una navegación web segura.

En concreto, FSS está desarrollada para usuarios de sistemas operativos basados en GNU/Linux pero los problemas que palia son comunes a otras plataformas.

En los últimos años han proliferado los problemas relacionados de alguna manera con la Red. Es el caso de los virus, los cambios de tarificación, defensa de los menores frente a los contenidos de la web, phishing, spam, uso indebido de contraseñas... Para cada uno de estos problemas existen soluciones particulares.

Un **dialer** es un programa capaz de hacer que el sistema se conecte a la Red a tra-

vés de números de tarificación adicional. Estos programas se usan como medio de pago en algunos tipos de páginas y su uso no es ilegal mientras no se haga de manera “oculta”. Permite a los usuarios efectuar suscripciones a estas páginas sin necesidad de rellenar complejos formularios ni dar números de tarjeta, de manera que las empresas propietarias obtienen beneficios por el cobro de las llamadas telefónicas.

Hace unos años, muchos usuarios españoles se enfrentaron a grandes facturas telefónicas por cambios no autorizados en la tarificación de sus conexiones. Hoy por hoy no existen estos problemas debido a la popularización de las conexiones ADSL y a la modificación de la legislación que regula los números de tarificación adicional.

El **correo electrónico** es uno de los servicios más populares vinculados a Internet; en muchos casos incluso reemplazando al correo tradicional. El gran problema de este servicio es que los usuarios creen que, “por defecto”, sus comunicaciones son privadas e inviolables. Sin embargo, los mensajes de correo electrónico pueden ser interceptados, leídos, cambiados, copiados... y todo ellos sin que el usuario emisor ni el destinatario lo noten.

Lo que evaluamos no es el hecho de que mandemos o no informaciones importantes por medio de nuestro correo, sino si se atenta contra nuestro derecho a la intimidad en las comunicaciones privadas. Si no tomamos las precauciones pertinentes, el derecho a la privacidad en las comunicaciones es perfectamente violable. Es igual de grave que se haga con datos poco o muy

importantes, en cuyo caso podría derivar en muchos más problemas.

El abuso de correo electrónico puede definirse como *las distintas actividades que perjudican a los servicios habituales de los servidores de correo*. Podemos encontrar muchos tipos de abusos: *spamming, mail bombing, unsolicited bulk email (UBE), unsolicited commercial email (UCE), junk mail...* Para evitar este tipo de problemas existen herramientas de cifrado, *remailers*, etc.

Otra problemática importante es la que nos plantea el control de los **contenidos no deseados**. Los *filtros de contenido* son elementos que pueden servir, como su propio nombre indica, de filtro para que algunos contenidos, de la naturaleza que sea, no sean accesibles durante la navegación por la Red. De esta manera, podríamos controlar los contenidos potencialmente peligrosos, ya sea desde la perspectiva de la de seguridad informática o desde el punto de vista moral.

Los **virus** y los **troyanos** son males que han encontrado en la Red un caldo de cultivo perfecto. La forma de trabajo del Medio y la gran cantidad de sistemas inseguros han hecho que éstos proliferen de una manera asombrosa en los últimos años. Podemos encontrar *backdoors*, troyanos, virus, gusanos... y todos ellos evolucionando rápidamente. Por lo tanto, es totalmente recomendable que usemos sistemas antivirus que sean capaces de responder a este sistema cambiante con la misma rapidez.

Todo sistema seguro debe ser capaz de gestionar y tener un **control de accesos**

al sistema. Los programas de Firewall son los encargados de proteger nuestro sistema de entradas ilegítimas ya sean humanas o automáticas (troyanos). Son difíciles de configurar adecuadamente, pues el usuario debe tener ciertos conocimientos de los servicios que necesita usar, puertos a utilizar, controles de acceso... Por lo tanto, si un usuario configura un firewall de manera incorrecta puede tener problemas con el funcionamiento de muchas de sus aplicaciones.

Las **contraseñas** son uno de los pilares fundamentales de cualquier sistema seguro. Teniendo una serie de precauciones en la creación se pueden solucionar bastantes problemas. La autenticación de usuarios es una de las bases de los sistemas seguros. Normalmente se basa en mecanismos que permiten identificar al usuario de una manera única y, habitualmente, hacen uso de contraseñas para la identificación.

El acceso a la Web es ahora más sencillo que hace unos años. En la actualidad, los navegadores son sencillos de usar y permiten que los usuarios controlen muchos servicios de una manera integrada y transparente. Como el instrumento habitual de navegación por la Red son los navegadores, el ámbito de trabajo en el que se mueve FSS es, precisamente, uno de los navegadores que existen en el mercado.

2.3. FSS a fondo

FSS es un proyecto que desarrolla totalmente desde el grupo de investigación del proyecto AGILA (Acceso Generalizado a Internet desde LinEx Avanzado [1]),

de la Universidad de Extremadura. AGILA es un proyecto de investigación englobado dentro del *II PLAN REGIONAL DE INVESTIGACIÓN, DESARROLLO TECNOLÓGICO E INNOVACIÓN DE EXTREMADURA* y ha sido financiado, en parte, por la Consejería de Educación, Ciencia y Tecnología de la Junta de Extremadura (*Expediente 2PRO3A09*).

FSS integra soluciones específicas para cada una de las problemáticas planteadas anteriormente. Podría haberse hecho un sistema más completo, pero los mecanismos que se han incluido en FSS son suficientes para asegurar unos niveles de seguridad altos para una navegación “segura” por la Red.

2.3.1. Anti-Dialers

La proliferación en España de las conexiones ADSL ha hecho que las conexiones vía módem telefónico vayan desapareciendo paulatinamente. Aun así, existen muchas regiones en las que la conexión vía módem telefónico es la única opción posible y, por lo tanto, los usuarios de estas zonas están expuestos al fraude de los cambios de tarificación.

FSS integra un programa que controla la posible modificación de archivos, véase Figura 1, que puedan afectar a las conexiones de red. De este modo, si algún programa intentara cambiar estas configuraciones, el anti-dialer registrará esta situación y permitirá al usuario reaccionar de manera efectiva contra este tipo de ataque, restaurando las configuraciones originales directamente.



Figura 1. Pantalla Dialers

2.3.2. Cifrado

Las herramientas de cifrado permiten que un usuario pueda ocultar cierta información a ojos indiscretos, de manera que sólo el destinatario, mediante los mecanismos de descifrado adecuados, sea capaz de obtenerla de manera clara.

El ambiente perfecto de uso de este tipo de herramientas es el relacionado con las comunicaciones privadas. FSS lo aplica a las comunicaciones de correo electrónico. La herramienta facilitada permite cifrar los mensajes, de tal manera que podemos garantizar que, si un mensaje es interceptado, será muy difícil, sino imposible, obtener el contenido de éste. Incluso se podría controlar si un mensaje ha sido interceptado, cambiado y reenviado.



Figura 2. Pantalla de cifrado

FSS ha incluido la herramienta GPA; que es una herramienta de encriptación que permite a un usuario gestionar, de una manera sencilla, las prestaciones avanzadas que ofrece GNUpg. Esta herramienta se complementa con el uso de *remailers*, que garantizan el anonimato del remitente en caso de que el correo sea interceptado durante su envío.

El uso de *remailers* en combinación con las herramientas de cifrado nos permitirá, de una manera eficiente, ocultar nuestra identidad como remitentes de un mensaje, además de la ocultación del contenido del mensaje con el cifrado.

FSS nos permite enviar un mensaje (véase Figura 2), cifrado o sin cifrar, a uno de los *remailers* que tenga establecido en la lista de *remailers* disponibles, utilizando el cliente de correo que tengamos marcado en la opción de clientes de correo.

2.3.3. AntiSpam

Paralelamente al correo electrónico, se han desarrollado algunos fenómenos problemáticos. Clara muestra de ello es el SPAM. Muchos usuarios experimentan a diario este problema; reciben correos no deseados de direcciones que desconocen. Este hecho, en grandes cantidades, afecta al correcto control del correo electrónico y puede realmente desesperar a un usuario que se ve desbordado por cientos de correos no deseados.



Figura 3. Pantalla Antivirus y filtrado

FSS nos da la posibilidad de elegir entre dos clientes de correo electrónico libre: Sylpheed y Thunderbird. (Véase Figura3)

Ambos clientes tienen herramientas que nos permitirán un filtrado de correo electrónico, paliando así los efectos del SPAM. ThunderBird es quizás una solución algo más segura al usar un filtro bayesiano que ayuda a gestionar los correos separando lo

que consideraremos correo deseado del no deseado.

Las dos opciones están basadas en software libre y se distribuyen bajo las condiciones de la licencia GPL. Ambas poseen un entorno gráfico intuitivo y son fáciles de usar e instalar para usuarios no avanzados.

2.3.4. Filtrado de contenido

Mediante los filtros de contenido podemos conseguir que un usuario no tenga acceso a páginas que tengan una temática que consideremos nociva. Por lo tanto el usuario puede funcionar como censor cortando el acceso a páginas de tono sexual, de apología del terrorismo...



Figura 4. Pantalla Antivirus y filtrado.
Filtrado

FSS ha optado por usar Dansguardian; que es una herramienta habitualmente usada en servidores, pero que puede ser usada en un pc normal sin problemas. Lo que hace es analizar las páginas y permitir o no el acceso a sus contenidos. FSS sólo gestiona la puesta en marcha, parada y reinicio de Dansguardian a través de una interfaz propia de FSS, siendo la configuración del

filtro independiente. (Véase Figura 4)

Para su uso doméstico, Dansguardian se distribuye bajo licencia GPL sin embargo, si se quisiera usar de manera comercial sería necesario contactar con el desarrollador.

2.3.5. Antivirus

En la actualidad la Red esta totalmente inundada de virus. Un sistema que no esta protegido tiene grandes posibilidades de tener problemas. Aunque la mayoría de éstos afecta a plataformas propietarias, el resto tampoco esta libre de virus. La plataforma elegida en este caso, GNU/Linux, hace que el sistema sea mas seguro frente a los virus que otras opciones de mercado, como puedan ser los sistemas Windows.



Figura 5. Pantalla Antivirus y Filtrado.
Antivirus

Existen virus, gusanos... que afecta a los sistemas GNU/Linux. Debemos tener en cuenta que el funcionamiento de los ataque es siempre el mismo: buscar una debilidad y atacarla. Se caería en un error si se dejase ese flanco al descubierto por el mero hecho de pensar que hay poca incidencia de virus

en esta plataforma.

FSS usa Clam, distribuido bajo licencia GPL, como antivirus. Se trata de un antivirus que emplea la base de datos de OpenAntivirus (también es un proyecto libre).

Clam está desarrollado en C y tiene características como: multihebra, soporta Amavis, escanea en archivos comprimidos, se actualiza automáticamente y aporta muchas otras atractivas características. Clam funciona, con las versiones actuales, en los sistemas operativos Linux, Solaris, FreeBSD y Windows (con cygwin).

Para hacer más sencillo el uso de Clam se ha desarrollado un entorno visual sencillo y portable, que hace uso de todas las opciones disponibles para el antivirus vía línea de comandos. (Véase Figura 5)

2.3.6. Firewall

Mediante el uso de Firewalls podemos controlar el tráfico entrante y saliente de una máquina. Su uso permite controlar quién y con quién se establecen conexiones además de a través por qué puertos se establece la conexión y los protocolos de comunicación usados. De esta manera, se puede determinar qué comunicaciones se desean permitir y qué comunicaciones se desean prohibir. Para esta labor es interesante disponer de una herramienta de escaneo de puertos. Dicha herramienta nos permitirá controlar qué puertos están abiertos y qué procesos los gobiernan, ayudando así a la localización de posibles brechas en el sistema.

Los firewalls son útiles, pero requieren algo de conocimiento de la arquitectura de red y de las formas de trabajo para configurarlos de una manera correcta. Una configuración errónea del firewall puede llevar a un mal funcionamiento de otros procesos que estén corriendo en la máquina.

FSS ha optado por incluir un firewall muy sencillo, Firestarter, que se puede ejecutar tanto en el entorno KDE como en GNOME. Como complemento ha incluido el mejor escaneador de puertos del mercado: Nmap [4]. Esta herramienta nos permitirá escanear las posibles debilidades de la máquina y el firewall cortar esos flujos de comunicación. (Véase Figura 6)

Las operaciones del firewall son acciones que afectan en gran medida al sistema, por lo tanto se necesitan derechos de administración para realizarlas. FSS avisa al usuario de esa situación.



Figura 6. Pantalla Firewall.

2.3.7. Contraseñas seguras

Las contraseñas son uno de los pilares de cualquier sistema seguro. Tener contraseñas débiles puede provocar múltiples problemas (p.e. suplantación de identidad) y por lo tanto elegir la clave adecuada es una labor a tener muy en cuenta. De la misma manera que es importante la creación de la clave, es importante su almacenaje, pues de nada vale una clave compleja que se guarda un medio de almacenamiento inseguro (p.e. un post-it pegado en el monitor).

Existen muchos ataques diferentes para la obtención de claves: ataque por fuerza bruta, diccionario, ingeniería social... Hay que tener en cuenta que una clave identifica a un usuario unívocamente; si se consigue dicha clave se tendrá acceso a lo que se está protegiendo con ella y se podrá suplantarse a ese usuario con todas sus consecuencias.

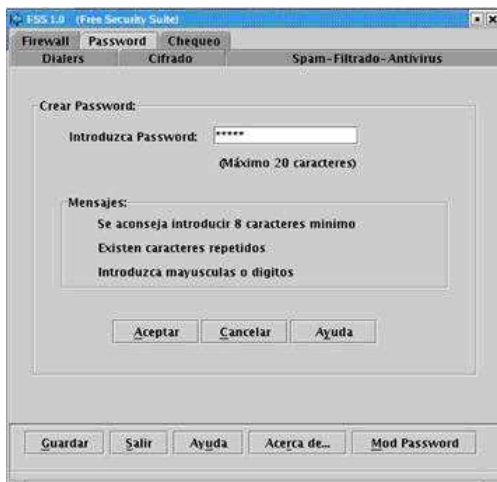


Figura 7. Pantalla creación claves seguras

FSS proporciona un asistente que guía al usuario en la creación de una clave segura. Permite gestionar las claves del sistema del usuario, informándole de si su contraseña tiene unos niveles mínimos de seguridad y qué condiciones tiene una clave “segura”. El usuario no está obligado a generar una clave segura, si quiere usar una clave insegura se le indicará la calidad de esta, pero se le permitirá usarla. (Véase Figura 7)

2.3.8. Chequeo

En algunas ocasiones es interesante conocer qué aplicaciones están corriendo en la máquina y así hacerse una idea de qué puertos deben estar ocupados y qué puertos deberían estar libres (p.e. si se está usando un cliente de correo deben estar abiertos los puertos estándar de correo).

Para este cometido FSS ofrece una aplicación que ayudará a controlar si está ejecutando alguna de las aplicaciones que hemos definido en una lista. De esta manera, el usuario podrá saber si se está o no ejecutando las aplicaciones que le interesa controlar. (Véase Figura 8)

En lo que a **navegadores web** se refiere existen varias opciones. En el desarrollo de FSS se hizo un estudio de los navegadores disponibles basados en licencias libres y se determinó que la opción ideal era Mozilla. Debido a su facilidad de uso, además de ser un navegador con proyección de futuro, dispone de un conjunto de usuarios amplio.

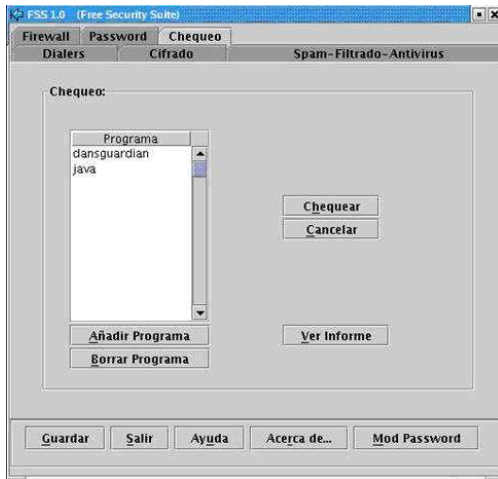


Figura 8. Pantalla chequeo de procesos

Mozilla tiene una arquitectura que permite añadir fácilmente aplicaciones externas que implementen funcionalidades específicas. Aprovechando esta característica se desarrolló FSS como un complemento que ayuda a los usuarios a navegar de manera segura.

2.4. ¿Que debemos hacer para usar FSS?

Hay que seguir unos sencillos pasos: búsqueda, instalación y restauración. La instalación tiene como requisito tener instalado previamente el navegador web Mozilla.

2.4.1. Búsqueda

El primer paso consiste en localizar el fichero de instalación: descargando el archivo de instalación de la red a una carpeta local u obteniéndolo de otro medio. El objetivo es localizar la carpeta local en la que

está el fichero de la extensión. (Véase Figura 9)

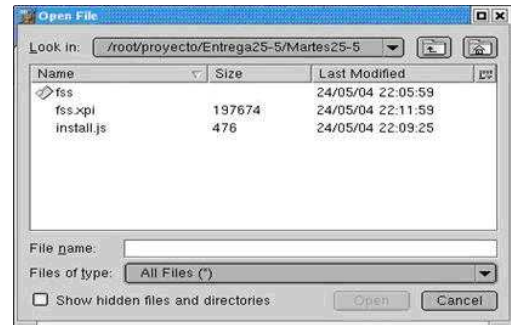


Figura 9. Pantalla de búsqueda

2.4.2. Instalación

Una vez seleccionado y ejecutado el archivo de instalación, el usuario seleccionará la extensión que va a instalar en el navegador web, en este caso FSS. (Véase Figura 10)

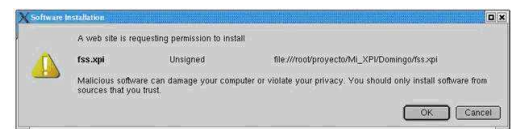


Figura 10. Selección de la extensión

2.4.3. Restauración

Cuando concluya el proceso de instalación el usuario debe reiniciar las aplicaciones relacionadas con Mozilla.

Una vez reiniciadas, el usuario podrá acceder a la extensión por los métodos normales de acceso a cualquier extensión de Mozilla: mediante la barra de estado (con un icono) o a través del menú de herramientas de Mozilla. (Véase Figura 11)

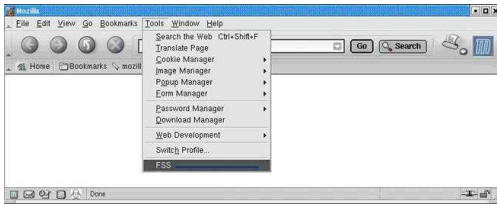


Figura 11. Acceso a la extensión

2.5. Conclusiones

La seguridad en la navegación web es una asignatura pendiente tanto para los usuarios domésticos como para las PYMES. FSS aparece en el horizonte como una opción libre que permite, de manera gratuita, establecer altos niveles de seguridad en la navegación web.

Viene a cubrir el déficit de soluciones integrales de seguridad, basadas en software libre, que permitan a los usuarios inexpertos obtener unos niveles de seguridad suficientemente altos como para navegar por la Red sin excesivas preocupaciones. Se solucionan en gran medida los problemas, relacionados con la navegación web, de los usuarios en la plataforma GNU/Linux, independientemente de la distribución.

FSS es sencillo en instalación y uso. Estar basado en software libre hace de esta suite una solución flexible, ajustable y actualizable fácilmente, permitiendo así que los usuarios completen su funcionalidad con ajustes personales.

Bibliografía

- [1] Sitio web del proyecto AGILA:
<http://gitaca.unex.es/agila>
- [2] Website ISO
<http://www.iso.org>
- [3] El REAL DECRETO-LEY 14/1999,
de 17 de septiembre, sobre firma
electrónica:
*[http://www.setsi.mcyt.es/
legisla/internet/rdley14_
99.htm](http://www.setsi.mcyt.es/legisla/internet/rdley14_99.htm)*
- [4] Howlett, Tony
Software libre. Herramientas de seguridad
ANAYA. ISBN 84-415-1835-1.

Implantación de LDAP como sistema de autenticación centralizada

Caso práctico de uso de software libre en la Universidad

Javier Sánchez Monedero jsanchezmonedero@yahoo.es

Escuela Técnica Superior de Ingeniería Informática. Universidad de Granada

Luis Meléndez Aganzo luism@uco.es

Área de Sistemas. Servicio de Informática. Universidad de Córdoba.

Sebastián Ventura Soto sventura@uco.es

Departamento de Informática y Análisis Numérico. Universidad de Córdoba.

Resumen

Este artículo constituye una guía para el diseño e implantación de un sistema de autenticación centralizada basado en LDAP en entornos de red típicos de sistemas informáticos de universidades o grandes empresas. Estos entornos se caracterizan por tener un gran número de usuarios a los que

debe dar una serie de servicios de red (correo electrónico, FTP...), así como posibilitar el acceso identificado a cuentas de usuario tradicionales. En concreto, el artículo se basa en el proyecto de fin de carrera *Implantación de LDAP como servicio de autenticación centralizada* [1], que en la actualidad se encuentra en funcionamiento en entorno de producción en el Servicio de Infor-

mática de la Universidad de Córdoba.

Así pues, además del diseño del servicio de directorio, resulta imprescindible analizar los parámetros que afectan al rendimiento y seguridad del mismo en función del software que utilicemos. En nuestro caso, después de evaluar una serie de productos de software privado elegimos OpenLDAP y justificamos esta elección mediante un análisis de las posibilidades que ofrece y pruebas de rendimiento e integridad diversas.

Finalmente, como complemento a este proyecto se desarrolló una guía completa de migración de NIS a LDAP utilizando OpenLDAP como servidor de directorio [2].

3.1. Introducción

En los Centros de Informática se proporcionan una serie de servicios que requieren que el usuario que necesita acceder a ellos demuestre su identidad, normalmente mediante la introducción de un nombre de usuario y una clave secreta. Puede tratarse de servicios restringidos a determinados colectivos o del acceso a información privada del propio usuario (como ficheros o mensajes de correo electrónico).

La información que almacena el sistema sobre los usuarios y las claves permitidas puede estar en el ordenador al que se accede para obtener el servicio, o residir en un sistema distribuido en el cual todos los sistemas comparten copias de esa misma información; puede existir un repositorio central al que los sistemas consulten la

validez de la información de autenticación aportada por el usuario, etc..

Con la tendencia actual de que no exista un gran ordenador central que proporcione todos los servicios, sino que estos sean aportados por una serie de ordenadores especializados (por razones de coste, fiabilidad al poder replicar los servicios, etc.), así como la variedad de servicios que actualmente se proporcionan (cuentas interactivas en los ordenadores, correo electrónico, acceso a páginas web restringidas, etc.), es especialmente importante implantar un modelo de información sobre datos de identidad de los usuarios que pueda ser accesible desde todos ellos. Este modelo de información deberá tener una gran fiabilidad y estabilidad y deberá ser fácil de gestionar.

El uso de los *servidores de directorio* apoyados en la tecnología LDAP¹ para cumplir estos objetivos es cada vez más generalizado.

3.2. Soluciones al problema

3.2.1. Solución basada en NIS

El sistema actual², basado en NIS, soluciona en parte la descentralización de la autenticación haciendo accesibles a una serie de máquinas una serie de ficheros, en lugar de que cada máquina tenga una copia de estos ficheros. Además, generalmente, la filosofía al utilizar NIS era asociar el usuario de un servicio al usuario del sistema operativo: antes, un usuario se conectaba a un servidor central para trabajar, leer su correo

¹Protocolo Ligero de Acceso a Directorios (*Lightweight Directory Access Protocol*)

²Llamaremos *actual* al sistema con NIS candidato a ser migrado a LDAP

y demás; en la actualidad, el usuario utiliza estos recursos a través de procesos servidores independientes, no desde el propio sistema operativo.

Sin embargo, la asociación de *usuario de servicio* a *usuario de sistema operativo* origina problemas a nivel de información, puesto que NIS sólo está preparado para almacenar información tradicional de sistemas UNIX, y a nivel de seguridad: cada usuario del sistema operativo es un punto de ataque potencial.

Además, NIS presenta otros problemas:

- Normalmente se necesita al menos un servidor NIS por cada subred.
- Existen limitaciones en las modificaciones de datos, que sólo se permiten desde el servidor principal y exigen la reconstrucción completa de los mapas NIS, que se traduce en un aumento de la carga de red y CPU. Por este motivo, las sincronizaciones no pueden ser inmediatas.
- No se pueden delegar privilegios de administración.

3.2.2. Solución basada en LDAP

Frente a los problemas mencionados, el uso de servidores de directorios LDAP para dar solución a estas necesidades es cada vez más generalizado. En torno a esta solución se plantean una serie de objetivos:

- Puesta en funcionamiento de varios servidores LDAP fiables mediante la elección, configuración y optimización de software de servidores de directorio LDAP

- Conseguido lo anterior, se necesita realizar pruebas de rendimiento e integridad de datos sobre la estructura de servidores implementada.
- Respecto a los clientes, es necesario un estudio para determinar y configurar el software necesario según las necesidades de los mismos.
- Finalmente debe diseñarse con cautela el proceso de migración de la tecnología NIS a LDAP de la manera más transparente y fiable posible.

Por referirse más a cuestiones directas de administración, los dos últimos puntos se desarrollan en un manual de administración [2] que no forma parte de este artículo.

3.2.3. NSS y PAM

Sabiendo que LDAP no es un más que un protocolo, ¿cómo funciona la autenticación basada en LDAP?

En los sistemas tipo UNIX modernos la autenticación se realiza mediante PAM (*Pluggable Authentication Module*, módulos de autenticación conectables). La empresa PADL ha desarrollado, entre otras herramientas de código abierto, el módulo `pam_ldap`, que integra LDAP con sistemas operativos que tienen soporte para PAM.

En estos sistemas operativos se podrán elegir ficheros de texto sencillo, NIS, NIS+ o LDAP para cada uno de los orígenes de datos que un sistema necesita a través del conmutador de los servicios de nombres (*NSS, Name Service Switch*).

Será necesario configurar NSS y PAM adecuadamente para que los futuros clientes del directorio sepan dónde encontrar qué tipo de información. Como las alternativas no son excluyentes, parece lógico que siempre se mantengan una serie de usuarios fuera del directorio en ficheros tradicionales. De este modo, si el servidor de directorio cae, siempre será posible acceder a los servidores vitales de la red.

3.3. Restricciones

3.3.1. Factores dato

Concretando ya en nuestro entorno real, identificamos una serie de limitaciones por parte del *cliente*³:

- El sistema operativo en producción instalado en los servidores es Linux Red Hat Enterprise.
- Existe una necesidad de economizar en consumo software y hardware.
- En todo momento deben tener preferencia los planos de seguridad e integridad de datos aun a costa de perjudicar otros planos.
- Sin que decaigan los aspectos anteriores, se debe optimizar el rendimiento del sistema completo.

3.3.2. Factores estratégicos

Identificadas las restricciones de nuestro proyecto debemos tomar ahora una serie de

decisiones que son las que marcarán la evolución del trabajo.

En primer lugar, primaremos el uso de software libre por una serie de razones estratégicas ya identificadas en la bibliografía [4].

Dentro de nuestro sistema de autenticación centralizada, la pieza más importante va a ser el servidor de directorio LDAP. Aunque no existen unas pautas preestablecidas que organicen la mayor parte de los pasos para la implantación del servidor de directorio, en la bibliografía relacionada con el mundo de los servidores de directorio encontramos unas recomendaciones para la implantación de servicios de directorio LDAP. Timothy A. Howes [3] divide el ciclo de vida del directorio en las fases de *diseño, desarrollo y mantenimiento*. Estas fases son las que seguiremos y nos llevan desde la especificación de las necesidades que va a tener este hasta la elección del software de directorio y su puesta en funcionamiento

Así pues, la elección del software de directorio será fundamental. En nuestro caso determinamos como características relevantes el coste, cumplimiento de estándares, rendimiento y escalabilidad, herramientas de administración disponibles y, finalmente, adopción en RedIris y otras comunidades⁴.

Independientemente de la opción elegida, será imprescindible someterla a pruebas de rendimiento y estabilidad que hagan trabajar al servidor, o conjunto de servidores, en condiciones similares a las del en-

³El Servicio de Informática de la Universidad de Córdoba.

⁴Puesto que el entorno de implantación es universitario, la comunidad RedIris supone un soporte técnico al que aportar y del que recibir soluciones que no se debe menospreciar.

torno real (o incluso superen con creces a la futura carga real). Encontramos un software privado para *benchmarking* de servidores LDAP llamado *DirectoryMark*⁵ que no se adaptaba a probar los aspectos del servidor que necesitamos evaluar. Así, decidimos implementar un conjunto de guiones de Perl que satisficieran las necesidades de las pruebas.

Respecto a la migración final de NIS a LDAP vamos a utilizar las herramientas libres *MigrationTools* de PADL Software. Para hacer esta migración de manera progresiva, y no dejar sin soporte a las máquinas con versiones de Solaris sin soporte para LDAP, utilizaremos una pasarela NIS/LDAP privada y de pago de la misma empresa.

3.4. Diseño del directorio

El primer paso que se debe dar es el diseño del directorio que va a alojar toda la información necesaria para que el sistema global, el conjunto de clientes, funcione correctamente. Así obtendremos una serie de requisitos que describan cómo va a ser el directorio.

Aplicaciones que van a acceder al directorio

Identificamos, por un lado, bibliotecas o librerías del sistema operativo relacionadas con la autenticación y la información para el funcionamiento de los sistemas operativos, que hasta ahora se almacenaba en mapas NIS, y, por otro, servidores de Telnet, SSH, FTP, Web, Correo electrónico y NFS, además de aplicaciones Web.

Datos

A partir de las aplicaciones que acceden al directorio se identifican una serie de datos que almacenar. Prácticamente bastará con migrar los mapas NIS (alias, automount, group, hosts, netgroup, passwd...) a LDAP.

Esquemas LDAP necesarios

El RFC 2307 [8], *An Approach for Using LDAP as a Network Information Service*, define una serie de esquemas LDAP estándar para sustituir a NIS. Además de las recomendaciones del RFC 2307 nos interesa crear un esquema propio (*uco.schema*) que nos va a permitir diseñar un control de acceso a máquinas y privilegios de administración que hasta ahora eran imposibles con NIS.

Espacio de nombres

Diseñar cómo se va a distribuir la información por el directorio es esencial. Lo más adecuado parece seguir la nomenclatura de NIS en las ramas del árbol LDAP obteniendo ramas de la forma `ou=People,dc=uco,dc=es` para los usuarios u `ou=Group,dc=uco,dc=es` para los grupos. Para las entidades relacionadas con la administración, que no coinciden con usuarios o entidades reales, lo mejor será construir una rama o ramas específicas.

⁵<http://www.mindcraft.com/directorymark/>

Topología de la red y replicación

Si queremos obtener una mayor garantía de estabilidad y escalabilidad de los servidores LDAP es obvio que debemos replicar la información entre varios. Parece que una estructura relativamente sencilla, y que proporciona mejoras en cuanto a la seguridad, puede ser la de un servidor maestro y dos de réplica en el que sólo se permitan operaciones de escritura en el servidor maestro a través de interfaces seguras. La mayor parte de la carga de trabajo, las lecturas, se desviará siempre a los servidores esclavos.

Seguridad

El aspecto de la seguridad en el directorio debe abordarse desde dos perspectivas.

En primer lugar, desde la propia seguridad del protocolo LDAP y las restricciones de acceso al directorio. Por la sensibilidad de la información que se transfiere debe utilizarse LDAP sobre TSL/SSL siempre que sea necesario. Además, la jerarquía de acceso a ramas del árbol y atributos de las entradas, como la contraseña, debe definirse claramente con las listas de control de acceso (ACL, *Access Control List*) de forma adecuada y teniendo en cuenta que un mal diseño de las ACL puede impactar negativamente en el rendimiento del servidor.

En segundo lugar, se debe implementar la seguridad en las cuestiones más comunes a cualquier tipo de servicio. La protección frente a ataques de denegación de servicios (DoS) la consideramos a dos niveles: ataques DoS directos a través de LDAP (se con-

trolan mediante la configuración del servidor LDAP) y ataques DoS indirectos, que deben evitarse con una auditoría básica de seguridad. Por supuesto, se debe evitar el acceso de usuarios corrientes a ficheros del servidor LDAP. En todas las máquinas no se debe dejar entrar a todos los usuarios. Una rama LDAP específica o los ficheros tradicionales de UNIX pueden ser un buen lugar para guardar las listas de administradores.

3.5. Elección de software de directorio para producción

Una vez analizado qué necesitamos a nivel de servidor LDAP, debemos decantarnos por una alternativa. Según fuentes bibliográficas [5] y [6] y de la consultora METAspectrum⁶, el software de directorio LDAP más utilizado es OpenLDAP, Microsoft Active Directory, iSunONE Directory Server y Novell eDirectory. Respecto a temas como el cumplimiento de estándares e interoperabilidad encontramos el *Directory Interoperability Forum*⁷ del *Open Group* que aporta información interesante. Aunque algunos de estos estudios están obsoletos o son poco objetivos, nos sirven de base para examinar las posibilidades de cada uno de estos productos.

El análisis realizado se ha basado en el examen y comparación de las prestaciones ofrecidas en las respectivas páginas web de cada producto para actualizar o completar los estudios encontrados.

Seguramente OpenLDAP no sea el mejor software de directorio; no obstante creemos que para las necesidades identificadas

⁶<http://www.novell.com/products/edirectory/metaspectrum.html>

⁷<http://www.opengroup.org/directory/>

en el apartado anterior no notaremos diferencia con las otras alternativas. Determinadas carencias, como la falta de herramientas de administración pueden suplirse con software de gestión de directorios independiente del servidor de directorio.

3.6. Parámetros que afectan al rendimiento y la estabilidad de OpenLDAP

Elegido OpenLDAP como software de servidor de directorio, pasamos a una segunda fase en la que realizamos un examen de sus características con el fin de optimizar su rendimiento.

Parece que el primer estudio debe centrarse sobre qué *backend* de base de datos debemos elegir. Según la experiencia en RedIris y diversas conversaciones en las listas de correo de OpenLDAP⁸, a primera vista la estabilidad de LDBM (rutinas de bases de datos del proyecto GNU) es buena. Podemos considerar este *backend* anticuado pero seguro, aunque no muy eficiente comparado con BDB. No obstante, los propios desarrolladores de OpenLDAP indican en las mismas listas que ya no trabajan con LDBM y que el proyecto se encamina hacia el uso de BDB (Berkeley DB)⁹. BDB ofrece ciertas mejoras como consistencia en las transacciones, durabilidad, bloqueo de granularidad fina. . . aunque la experiencia de uso de OpenLDAP con BDB en ciertos entornos es de inestabilidad del servidor, sobre todo en las escrituras, en comparación con LDBM. Vemos pues que el uso de BDB debemos complementarlo con pruebas propias que

nos aseguren su estabilidad y, en caso de resultados negativos, optar por LDBM. A esto hay que añadir que LDBM es el *backend* por defecto en los paquetes de instalación de OpenLDAP en varias distribuciones como Red Hat.

Indexar lo necesario partiendo de las fuentes de información

En todo caso, hay muchos parámetros que afectan al rendimiento de OpenLDAP (y al de otros servidores) con independencia del software de gestión de la base de datos interna, como es el caso de la indexación de atributos. Los servidores de directorio trabajan con bases de datos indexadas para optimizar las lecturas. Está claro que cuantos más atributos indexemos más eficiente será el acceso a entradas del directorio a través de estos atributos. Sin embargo, la indexación de atributos no se puede tomar a la ligera, ya que impacta negativamente sobre los tiempos de inserción y modificación de datos, puesto que es necesaria la reconstrucción de índices.

La mejor manera de estimar qué atributos indexar, al menos inicialmente, es partiendo de la información que nos da el RFC 2307, el conjunto de aplicaciones y servicios que acceden al directorio y, finalmente, la información almacenada en los archivos de registro de los servidores LDAP. Este último caso deberá observarse especialmente en los primeros períodos de implantación del sistema. Basta con buscar líneas similares a:

```
bdb_equality_candidates:
```

⁸<http://www.openldap.org/lists/>

⁹<http://www.sleepycat.com/>

(`homeDirectory`) `index_param` failed en los ficheros y contar sus ocurrencias para orientarnos respecto a parámetros no indexados que se estén demandando al servidor con más o menos frecuencia.

Optimización de OpenLDAP y BDB

Respecto al *backend* de base de datos es conveniente revisar todas directivas de tratamiento de transacciones, caché, comprobación de errores, etc. de OpenLDAP y BDB. En este sentido desecharemos todas las directivas cuyo uso entre en conflicto con las premisas de seguridad y estabilidad del servidor. Descartadas, pues, las directivas que nos creen estos conflictos, obtenemos las siguientes:

- `cachesize/set cachesize`: fija tamaño de la base de datos en caché de memoria en número de entradas.
- `checkpoint`: frecuencia con que se vacían los *buffers* de los ficheros de transacciones.
- `dbnosync` y `dbsync/set (flags DB_TXN_NOSYNC)`: opciones de sincronización de la memoria y almacenamiento inmediata o no.
- `dirtyread`: lecturas sobre transacciones no confirmadas.
- `set lg regionmax, set lg bsize, set lg dir`: opciones de ficheros de transacciones de la base de datos y su ubicación física.

Uno de los aspectos fundamentales en la optimización va a ser la *cantidad* (referida en número de entradas o índices) de

la base de datos que se va a alojar en caché de memoria principal. Al respecto, encontramos en la sección de soporte de OpenLDAP el artículo *How do I determine the proper BDB/HDB database cache size?* [7], en el que se hace una serie de recomendaciones para calcular tamaño mínimo de caché para optimizar OpenLDAP. Entre ese tamaño mínimo y elegir tamaños de caché mucho más grandes, parece ser que no existen apenas diferencias de rendimiento. La justificación a esto se basa en el funcionamiento de los árboles B+ en los que están basados tanto LDBM o BDB. En estas estructuras lo que realmente importa es encontrar el índice que apunta a una entrada lo más rápido posible, de modo que interesa guardar una copia de los índices en memoria principal. Si la cantidad de entradas y los recursos hardware lo permiten, lo ideal sería ubicar toda la base de datos en memoria principal.

Access Control Lists (ACL)

Otro aspecto de la optimización es el referente a las ACL de control de acceso, que, como hemos comentado, deben simplificarse al máximo.

Name Service Caching Daemon (NSCD)

Finalmente, aunque no forme parte de la optimización del servidor, que sí del sistema global, debemos tener en cuenta la gran reducción en la carga de red y del servidor que supone por lo general la instalación de un servidor NSCD en los clientes. El *Demonio o Servicio de Caché de Servicio de Nombres (Name Service Caching Daemon)* guarda durante un tiempo las consultas de de-

terminados datos que le indiquemos en el cliente para evitar que tengan que volver a ser solicitadas al servidor.

3.7. Pruebas de rendimiento e integridad de OpenLDAP

Con el fin de determinar el efecto que suponen los distintos parámetros de OpenLDAP y BDB que afectan al rendimiento, debemos realizar pruebas que nos aseguren el buen funcionamiento del sistema. Los objetivos que perseguimos con las pruebas de rendimiento e integridad son comprobar el impacto de algunas directivas del *backend* de Berkeley, comprobar la estabilidad de la base de datos ante operaciones de escrituras masivas (modificación de atributos e inserción y borrado de entradas), comprobar el impacto de la indexación de nuevos atributos y obtener datos de eficiencia similares a los que se esperan en producción.

Después de medir el tamaño total de la base de datos, vemos factible fijar un tamaño de caché de memoria que pueda alojar en un momento dado a todas las entradas del directorio en memoria principal. Las pruebas se realizarán con este parámetro fijo.

Herramientas para las pruebas

Para la realización de estas pruebas se ha desarrollado un conjunto de scripts de Perl utilizando los módulos necesarios para utilizar el protocolo LDAP (`Net::LDAP`) y módulos para mediciones de tiempo de alta precisión (`Time::HiRes`). Estos automatizan las siguientes operaciones:

- Configuración, llenado e indexado

inicial del servidor.

- Realizar una cantidad considerable de operaciones de lecturas y escrituras por separado.
- Simular un conjunto de operaciones similares a las reales en las que pueda ajustarse la cantidad, porcentaje y contenido de operaciones de cada tipo.
- Medida de los tiempos de cada tipo de operación y generación de estadísticas, tablas y gráficas ilustrativas.

Pruebas de carga masiva de datos

El objetivo de estas pruebas es comprobar los límites de adición masiva de datos con el servidor en línea, así como el impacto de las opciones de configuración del *backend* BDB en estas operaciones.

Los resultados dieron pie a la aparición de problemas de integridad de datos al realizar adiciones masivas con el servidor en línea. Con el servidor parado, y configurando BDB adecuadamente, comprobamos que en pocos segundos se puede poner en marcha el servidor a partir de un fichero de texto sencillo de copia de seguridad.

Pruebas de operaciones masivas

Estas pruebas tienen como objetivos determinar la estabilidad del servidor ante cargas de trabajo masivas y comprobar que el servidor OpenLDAP no empeora su rendimiento ni consumo de memoria ante cargas de trabajo superiores a las de producción. En este tipo de pruebas forzamos un número de escrituras superior al normal en

directorios para tratar de forzar posibles fallos del *backend*, previo conocimiento que estos podían ser los únicos fallos que tuviese.

Como batería de pruebas se lanzaron 10 clientes durante 20 horas desde los que se hacían operaciones de lectura y modificación de atributos aleatorios. Sin detener estos clientes se incrementó el número de entradas y posteriormente se borraron aleatoriamente varias de ellas mientras se mantenían las operaciones incluso sobre entradas ya inexistentes. En ningún momento hubo problemas de integridad de datos.

Pruebas de simulación de entorno real

Finalmente lanzamos unas pruebas que se parezcan más a lo que puede ser un entorno en producción con la intención de obtener datos reales sobre el impacto de la indexación de atributos para tipo de operaciones similares sobre el servidor. Paralelamente se busca obtener datos del impacto del aumento de entradas en la base de datos sobre varios tipos de operaciones distintas.

Las pruebas consistieron en probar diversas configuraciones de OpenLDAP sobre todo en cuanto a indexación de atributos y sus opciones, por ejemplo, algunas de las combinaciones de índices que se utilizaron se muestran en la tabla siguiente. Durante la ejecución de pruebas se fueron desechando las combinaciones que obtenían los peores resultados al aumentar el número de entradas progresivamente.

Combinación	Líneas del fichero <i>slapd.conf</i>
index01	index objectClass eq
index02	index objectClass eq index uid eq
index03	index objectClass eq index uid,uidNumber, gidNumber eq
index04	index objectClass eq index uid,uidNumber, gidNumber eq,pres

Conclusiones generales sobre las pruebas

En primer lugar, se ha comprobado cómo puede ser interesante activar temporalmente los parámetros `DB_TXN_NOSYNC` y `DB_TXN_NOT_DURABLE` para las cargas e indexaciones ya que disminuye el tiempo de construcción de la base de datos e índices muy considerablemente. En estos casos no se han observado inconsistencias en los datos.

Tras someter al servidor a cargas de operaciones muy superiores a las que tendrá en su fase de producción, no se han observado tampoco ningún tipo de corrupción de datos, ni siquiera forzando miles de operaciones de escritura simultáneas. Parece que la aplicación de los parches oficiales para la versión 4.2.52 de Berkeley DB es suficiente para solventar los problemas de corrupción de datos de los que se tenían noticias a través de diversas listas de correo.

Al realizar pruebas con tamaños de caché de memoria pequeños, por ejemplo de 10 MB, el proceso de inserción de entradas falló con el servidor en marcha. Las soluciones que se encuentran cuando no se quieren utilizar grandes cantidades de caché de memoria pasan por dividir la inserción de datos en ficheros pequeños, de manera que

la carga de datos se haga de manera gradual.

Respecto al uso de herramientas de trabajo *online* de OpenLDAP para cargas masivas de datos lo desaconsejamos totalmente por varios motivos:

- El consumo de memoria del servidor crece hasta incluso duplicar el consumo habitual sin que se aprecie una liberación de la misma a corto plazo, entendiendo por corto plazo el transcurso de entre 30 minutos y una hora. En estos casos merece la pena reiniciar el servidor para liberar memoria.
- Los ficheros de registro de las transacciones ocupan un espacio de disco que supera al tamaño habitual de la base de datos. Se trató de eliminar los ficheros de registro con las herramientas de administración de Berkeley DB¹⁰ sin éxito. Hasta que no se reinició el servidor, los ficheros de transacciones siguieron bloqueados.

En cuanto a la indexación de atributos, para nuestro entorno y el uso que se le va a dar al servidor podemos incluir tantos índices como queramos. Las pruebas demuestran que el tiempo de respuesta del servidor para devolver resultados de búsquedas en atributos indexados no empeora aún al incrementar el número de entradas en la base de datos. En este punto recordamos que una copia prácticamente completa de nuestra base de datos siempre ha sido alojada en la memoria RAM del servidor. Dado que en nuestro entorno hardware la memoria no va a ser un problema para el tama-

ño de la base de datos de OpenLDAP, no se han realizado pruebas con tamaños inferiores de caché de memoria. Además, también es conveniente destacar que el tipo de búsqueda que se ha realizado ha sido del tipo “sub”, que es la más corriente en el protocolo LDAP. Así mismo, como las operaciones de escritura reales sobre atributos casi siempre se harán sobre el campo de la contraseña, que no está indexado, el tiempo de operaciones de modificación general se mantiene constante independientemente del tamaño de la base de datos.

El tiempo de búsqueda de atributos no indexados crece muy rápidamente conforme la base de datos tiene más entradas. Por tanto debe vigilarse la demanda de atributos no indexados para plantear su indexación, sobre todo cuando estos atributos no vayan a ser modificados muy a menudo. Esto puede hacerse analizando los ficheros de registro generados por el proceso `slapd` en casi todos los niveles de depuración.

Los ficheros de índices no ocupan demasiado. Al indexar uno de los campos de mayor tamaño, como por ejemplo el campo *cn*, el fichero de índices de este campo apenas ha sobrepasado 1 MB de tamaño en disco para una base de datos de 70 MB con un campo *cn* en todas o casi todas sus entradas. Estos datos se refieren siempre al tipo de índices utilizados en las pruebas, *eq* y *pres*.

3.8. Conclusiones

A partir de los resultados de las pruebas, concluimos que se ha acertado al elegir

¹⁰Estas herramientas comprueban qué ficheros de transacciones se pueden borrar y cuáles no.

OpenLDAP como software servidor de directorio. En conjunto, se ha creado un sistema de alta disponibilidad, fiabilidad y rendimiento junto con una guía completa de migración de NIS a LDAP desde la instalación de los servidores LDAP hasta el reemplazo progresivo de LDAP con NIS apoyándose en la pasarela NIS/LDAP

Como futuro trabajo sería interesante comprobar el impacto de la encriptación de comunicaciones sobre el rendimiento global del sistema. También como trabajo futuro, y según se vayan definiendo directrices y acuerdos en RedIris será necesario adaptar los esquemas utilizados o cualquier otro aspecto del directorio.

Bibliografía

- da edición 2003, ISBN: 0-672-32316-8
- [1] Javier Sánchez Monedero, *Implantación de LDAP como sistema de autenticación centralizada. Manual Técnico.*, Directores: Sebastián Ventura Soto. Universidad de Córdoba. Departamento de Informática y Análisis Numérico, Luis Meléndez Aganzo. Universidad de Córdoba. Servicio de Informática. Área de Sistemas, 2004.
 - [2] Javier Sánchez Monedero, *Implantación de LDAP como sistema de autenticación centralizada. Manual de Administrador.*, Directores: Sebastián Ventura Soto. Universidad de Córdoba. Departamento de Informática y Análisis Numérico, Luis Meléndez Aganzo. Universidad de Córdoba. Servicio de Informática. Área de Sistemas, 2004.
 - [3] Timothy A. Howes Ph.D., Mark C. Smith, Gordon S. Good, *Understanding and Deploying LDAP Directory Services*, AddisonWesley, Segunda edición 2003, ISBN: 0-672-32316-8
 - [4] Monera Daroqui, *Modelos de negocio basados en software libre*, http://www.opensistemas.com/Articulo_completo.644+M54a708de802.0.html, Propietario: OpenSistemas. Fecha de Consulta: octubre 2004
 - [5] Brian Arkills, *LDAP Directories Explained: An Introduction and Analysis*, Addison Wesley, 2003, ISBN: 0-201-78792-X
 - [6] Norbert Klasen, *A Comparative Performance Analysis of 7 Lightweight Directory Access Protocol*, <http://www.daasi.de/staff/norbert/thesis/>, Propietario: DAASI International. Fecha de Consulta: octubre 2004
 - [7] Kurt Zeilenga, *How do I determine the proper BDB/HDB database cache size?*, http://www.openldap.org/faq/index.cgi?_recurse=1&file=190, Propietario: The OpenLDAP Foundation. Fecha de Consulta: octubre 2004
 - [8] L. Howard, *RFC 2307: "An Approach for Using LDAP as a Network Information Service"*, <http://www.ietf.org/rfc/rfc2307.txt>, Propietario: The Internet Society. Fecha de Consulta: octubre 2004

MOVICUO: Comunicaciones móviles y software libre para la ubicuidad

Javier Domingo Carmona Murillo `jcarmur@unex.es`

Depto. Informática. Universidad de Extremadura.

José Luis González Sánchez `jlgzs@unex.es`

Depto. Informática. Universidad de Extremadura.

Alfonso Gazo Cervero `agazo@unex.es`

Depto. Informática. Universidad de Extremadura.

Lorenzo Martínez Bravo `lorenzom@unex.es`

Depto. Informática. Universidad de Extremadura.

Resumen

Las comunicaciones móviles se encuentran en un proceso de evolución constante. Desde la aparición de GSM (2G), que ofrece servicios de voz a través de una red de circuitos conmutados, distintos cambios en la tecnología han permitido adaptarse a las

necesidades de los usuarios. GPRS (2.5G) complementa el diseño de GSM con una red de conmutación de paquetes para el tráfico de datos, además de aportar QoS. Sin embargo, las tasas de velocidad alcanzadas en GPRS (171,2 kbps en situaciones ideales) no son adecuadas para determinados servicios como el tráfico multime-

dia. UMTS (3G) utiliza un nuevo método de acceso al medio, CDMA (Code Division Multiple Access), que añade complejidad pero permite velocidades más altas (hasta 2Mbps).

Partiendo de esta situación surge *MOVICUO*¹, un proyecto de investigación centrado en el estudio de las tecnologías de comunicaciones móviles más utilizadas (GSM, GPRS y UMTS), y su posible desarrollo en sistemas libres como GNU/LinEx. El resultado de este desarrollo es la aplicación libre GNOME-GPRS, una herramienta que soluciona el vacío de conectividad a Internet a través de GPRS.

4.1. Introducción

Más de mil millones de personas en el mundo (uno de cada cinco habitantes) y más de 200 países, utilizan teléfonos móviles GSM (Global System for Mobile communications). Esto ha supuesto que sea la tecnología de referencia para la telefonía móvil al ser la elección de más del 80 % de los nuevos clientes, haciendo que la cantidad de terminales móviles superen a las líneas telefónicas fijas.

A pesar de estas cifras, ya han aparecido varias tecnologías que ofrecen otro tipo de servicios que GSM no permite por sus características de diseño. Evoluciones como GPRS (General Packet Radio Service), que permiten el acceso a redes de datos, en lugar de las tradicionales redes de voz, han

supuesto que el acceso a Internet tenga otra lanzadera con millones de usuarios potenciales, que cada vez demandan tecnologías que ofrezcan más movilidad.

Considerando este proceso de evolución, Movico es una base en la que apoyarse para el desarrollo de sistemas de comunicaciones móviles de última generación, ya que en este proyecto se detallan todos los aspectos tecnológicos principales de las tecnologías GSM, GPRS y UMTS (Universal Mobile Telecommunications System), además de aportar una solución libre para la implantación de GPRS en un sistema como GNU/LinEx. Los avances del proyecto pueden consultarse desde su página web [1].

4.2. Desarrollo e implementación de conexiones GPRS en GNU/LinEx

4.2.1. Introducción

Para el desarrollo de conexiones GPRS en GNU/LinEx, nos hemos basado en las recomendaciones de los estándares publicados por la 3GPP [2], lo que permite la compatibilidad del desarrollo con todos los dispositivos que sigan esta filosofía. La velocidad máxima alcanzada de GPRS es, teóricamente, 171,2 Kbps. Este valor es una referencia, ya que se alcanzarían si un mismo usuario utilizara los ocho time-slots² o ranuras de tiempo del canal, si ningún otro usuario estuviera compartiéndolo, y si no

¹Este proyecto ha sido patrocinado en parte por la Junta de Extremadura a través de los proyectos siguientes: Proyecto Ágila (Expediente 2PR03A090) y Campus Ubicuo (Expediente PDT05A041) <http://gitaca.unex.es/agila>

²Cada una de las 8 franjas de tiempo en el que TDMA (Time Division Multiple Access) divide un canal de 200 KHz de frecuencia.



Figura 4.1: Estructura del sistema

existieran códigos de protección ³. En realidad, el terminal también limitará esta velocidad. Hay definidas 29 configuraciones que indican cuantas ranuras de tiempo pueden usarse para subida y bajada. La capacidad de cada una de estas franjas de tiempo varía (de 9 a 20 Kbps) dependiendo de la codificación que se utilice. Para los teléfonos GPRS, donde la limitación de consumo de potencia es un factor clave, las configuraciones más comunes son del tipo 4+1, que significa que se usan cuatro ranuras de tiempo para la bajada y una para la subida. Los dispositivos consumen más energía cuanto mayor sea el bit rate, aunque mucha de la energía generada por la batería es perdida en forma de calor. Por eso, es difícil construir un terminal que utilice las 8 rodajas de tiempo para la subida y las 8 para la bajada, ya que el consumo de energía sería enorme y el calor desprendido insoportable. También hay otros tipos de limitaciones físicas debido al uso de los transceiver ⁴, pero la explicada anteriormente es la más importante. [3]

4.2.2. Componentes de un sistema móvil

En el proyecto Movicio, el acceso a Internet a través de GPRS se realiza utilizando el teléfono móvil como módem, mientras que el sistema GNU/LinEx es el sistema final que actúa de interfaz con el usuario. Siguiendo la nomenclatura utilizada por la 3GPP [2], los elementos que componen el sistema son [4]:

- TE (Terminal Equipment): Dispositivo en el que están las aplicaciones y el interfaz con el usuario (ordenador).
- MT (Mobile Terminal): Es la parte que conecta a la red (teléfono móvil).

A la hora de establecer una conexión, lo primero que tendremos que hacer es conectar los dispositivos en la capa física. Esta conexión puede realizarse bien con un cable USB, cable serie, Bluetooth, Infrarrojos, etc. Será necesario tener instalado el módulo que el sistema operativo (en nuestro caso GNU/LinEx) tiene que cargar para re-

³El interfaz aéreo introduce muchos errores, por esto, se definen cuatro códigos de protección que limitan la velocidad.

⁴En comunicaciones se define como un transmisor/receptor de señales de radio frecuencia. Se utiliza conectar dispositivos por enlaces inalámbricos.

conocer dicha conexión física entre el TE y el MT.

El siguiente paso consiste en establecer una conexión punto a punto (PPP - Point to Point Protocol) entre los dispositivos. PPP está implementado en la mayoría de los sistemas operativos, entre ellos Debian (y GNU/Linux). PPP crea un interfaz de comunicación hacia las capas inferiores y permite a la capa IP comunicarse con la red de forma transparente. Además realiza otras optimizaciones, como evitar el envío de información redundante. En la figura 4.2, aparece la pila de protocolos resultante tras la activación de PPP. [5]

4.2.3. Acceso a capas inferiores. Comandos AT+

Una de las características más interesantes en la comunicación entre el MT y el TE, es el acceso a las capas inferiores mediante los comandos AT. Para proporcionar a los desarrolladores el acceso al hardware del terminal GPRS y a las propiedades de la red directamente, el estándar GPRS define una serie de comandos AT (Attention commands). Los comandos AT han sido (y son) desarrollados con la recomendación ITU-T⁵ V.25ter (Serial asynchronous automatic dialing and control), con el objetivo de dar un conjunto de comandos que el sistema de comunicaciones debe ofrecer a las capas superiores.

Por lo tanto, muchos de los comandos AT son también usados en GSM y UMTS, así como en los módems tradicionales. El conjunto de comandos AT que se han añadido al de comandos tradicionales de control de

módem, se denominan AT+ o comandos AT extendidos. El estándar [6] describe el conjunto de comandos AT+ para los terminales GSM, incluyendo GPRS.

4.2.4. Conexión a la red

Veamos que ocurre cuando un teléfono móvil (GSM con funcionalidad GPRS), se conecta a la red. En primer lugar, el dispositivo necesita decirle a la red que puede hacer y recibir tanto conexiones GSM como GPRS. Este procedimiento se llama Attach y es similar para GSM y GPRS. Lo contrario de un Attach GPRS lo llamaremos Detach GPRS, que elimina el terminal GPRS de la red. Esto ocurre cuando se apaga el teléfono, al igual que el Attach se produce al encenderlo. Para ejecutar el Attach (o detach) a la red, se utiliza el comando AT+CGATT.

Una vez realizado el Attach, el móvil necesita una dirección IP y otros parámetros de conexión. Esta tarea es realizada mediante la activación del contexto PDP (Packet Data Protocol). El contexto PDP puede ser visto como un registro software que mantiene parámetros que son relevantes para una conexión concreta. Esta información incluye los protocolos que serán utilizados (IPX25, PPPetc), la dirección IP (si se usa el protocolo IP), la QoS [7] (o mejor dicho, su perfil), e información de compresión de datos. La activación del contexto PDP hace que el dispositivo GPRS sea visible para el GGSN (GPRS Gateway Support Node), que hace posible conexiones con redes externas a GPRS.

Para activar el contexto PDP, el estándar GPRS define el comando AT+CGDCONT.

⁵Unión Internacional de Telecomunicaciones. ITU es un organismo de las Naciones Unidas encargado de regular las telecomunicaciones, entre las distintas administraciones y empresas operadoras.

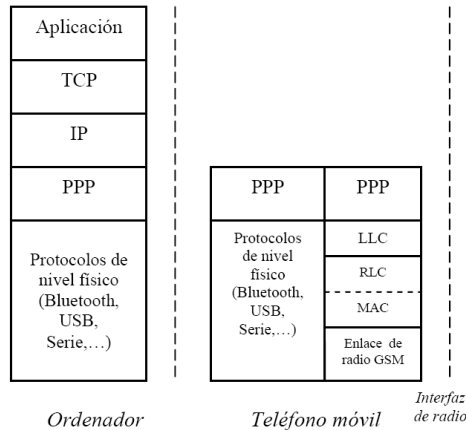


Figura 4.2: Pila de protocolos del enlace PPP

Este comando permite indicarle a la red los valores utilizados para establecer un contexto PDP. En la figura 4.3 se muestra la secuencia de mensajes AT del Attach, seguido por la activación del contexto PDP. Ahora el usuario GPRS estará preparado para enviar y recibir paquetes [8].

Esta primera parte ha dado un conocimiento general de cómo activar una conexión GPRS a través de comandos AT+. En el proyecto Movico, esta activación se ha realizado desde el sistema operativo GNU/Linux, aunque la metodología seguida es similar para el resto de sistemas Linux. Los comandos AT extendidos utilizados en el proyecto, aparecen en los estándares del 3GPP [6].

4.3. Solución de conexión GPRS sobre GNU/Linux. GNOME-GPRS

4.3.1. Introducción

Partiendo de la investigación presentada en los puntos anteriores, desde el proyecto Movico quisimos plasmar este esfuerzo en un desarrollo que fuera útil a los usuarios de Linux y que siguiera los objetivos de accesibilidad propuestos en el proyecto.

Este desarrollo es la herramienta GNOME-GPRS, una aplicación para GNU/Linux (y otros sistemas Linux) que permite realizar conexiones a Internet a través de GPRS usando como módem un teléfono móvil. GNOME-GPRS se ha implementado utilizando programación GNOME [9], por lo que la aplicación tendrá el mismo aspecto que el resto del sistema.

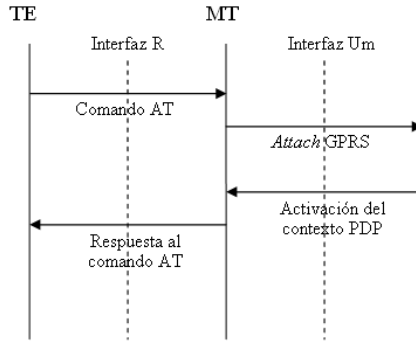


Figura 4.3: Activación del contexto PDP con comandos AT+

Ventana principal de la aplicación

La ventana principal del programa tiene un aspecto sencillo (ver Figura 4.4). Tras ejecutar el programa por primera vez, o en general cuando GNOME-GPRS no este correctamente configurado, muchos de sus componentes estarán desactivados, lo que significa que el programa no conoce su valor. Así, los componentes *Proveedor* y *Puerto*, que son entradas de texto no modificables, estarán desactivadas hasta que no se configure el programa. De la misma forma, los botones *Conectar* y *Analizar gráficas* están también desactivados. En la parte inferior de la ventana, aparecen tres botones, que sí están activados, éstos son: *Ayuda*, *Configuración* y *Salir*. En este documento tan sólo vamos a explicar la opción de configuración de la aplicación, ya que los otros dos botones tienen funciones obvias.

Configuración

Al pulsar sobre el botón *Configuración* en la pantalla principal, accederemos a una de las partes fundamentales de GNOME-

GPRS, ya que para poder realizar la conexión, es necesario establecer algunas opciones. Lo primero que veremos al acceder a esta sección del programa será una ventana con cuatro pestañas, que definen cuatro tipos de opciones bien diferenciadas. Veamos cada una de ellas por separado.

- PARÁMETROS OBLIGATORIOS:** Permite establecer las opciones mínimas sin las cuales no se puede conectar. Además, tan sólo configurando correctamente estas opciones la conexión se puede realizar sin ningún problema, aunque si hacemos eso, no estaremos aprovechando todas las posibilidades que permite GNOME-GPRS. Esta ventana se divide en dos partes, la parte superior en la que se configura el dispositivo que tenemos conectado, y la parte inferior en la que se configura el operador con el que se va a conectar. Para configurar el teléfono, debemos rellenar tres campos, que pueden ser configurados automáticamente por GNOME-GPRS utilizando el botón *Detectar*.

Una vez configurada la parte del dispositivo en los parámetros obligatorios, es necesario que se den valores relacionados con el proveedor del servicio al que se está suscrito, y que ofrece el servicio GPRS.

- **PARÁMETROS AVANZADOS:** Tras rellenar los parámetros obligatorios, las siguientes opciones a configurar son los parámetros avanzados. La ventana ahora se divide en cinco partes (*DNS*, *Aspecto al conectar*, *QoS*, *Opciones de las gráficas* y *Otros parámetros*). En la parte superior de la ventana está el apartado de configuración de los servidores DNS. *Aspecto al conectar* contiene dos casillas de activación, que establece el comportamiento de GNOME-GPRS una vez que se conecta. *Integrar en el área de notificación* permite que al conectar, las ventanas de GNOME-GPRS desaparezcan, quedando únicamente un pequeño icono en el área de notificación del panel del escritorio de GNOME. De esta forma, no queda ninguna ventana abierta de la aplicación, y tan sólo tendremos un pequeño botón que irá cambiando según estemos enviando datos, recibidos o ambas cosas a la vez. Una vez configurado el aspecto, aparece el apartado dedicado a la calidad del servicio (QoS). Únicamente tendremos que decirle al programa si queremos activar la QoS o no. Si decidimos activarla, tendremos que configurar las últimas dos pestañas de la ventana: *QoS solicitada* y *QoS mínima*. Si no, los valores de estas dos últimas pes-

tañas serán ignorados. Tras la QoS, GNOME-GPRS nos da la posibilidad de configurar las *Opciones de las gráficas*. La aplicación nos permite ver gráficamente la velocidad de subida y bajada en cada momento.

- **QoS SOLICITADA:** La QoS en GPRS se define a través de cinco parámetros: Prioridad, retardo, fiabilidad, throughput máximo y throughput medio. Tanto en este apartado como en el de QoS mínima, se pueden elegir varios valores para cada uno de los parámetros.
- **QoS MÍNIMA:** Con la QoS solicitada, le indicamos a la red cuales son los valores de QoS que nos gustaría tener. Luego la red puede darnos o no esos valores requeridos en función de la disponibilidad de recursos que tenga, del número de usuarios conectados y muchos factores más de los que depende la negociación. Sin embargo, la QoS mínima va un paso más allá y establece los cinco valores de QoS mínimos sin los cuales no se podrá realizar la conexión.

Conectar

El botón *Conectar* inicialmente se encuentra deshabilitado, y no se puede pulsar. Se mantendrá así hasta que GNOME-GPRS esté configurado correctamente. Cuando se pulsa el botón, comienza el proceso de conexión. El programa negocia con la red la configuración elegida para llegar a un acuerdo. Podemos conocer el estado en el que se encuentra GNOME-GPRS a través de los mensajes que se muestran en la ventana

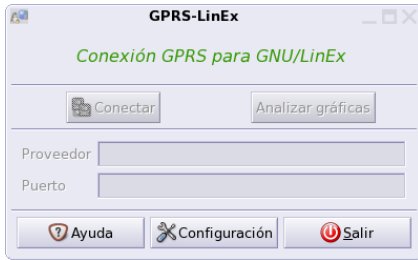


Figura 4.4: Aspecto inicial de GNOME-GPRS

Conectando. Si la negociación ha sido satisfactoria, desaparece *Conectando* y aparece la ventana *Conectado* (figura 4.5). A través de ella, podemos dar por finalizada la conexión y salir, o bien acceder a la información de la actual conexión (Situación de la conexión, Velocidad de bajada, velocidad de subida, dirección IP, datos recibidos, datos enviados, etc). Si se ha elegido el soporte de gráficas, en *Información* aparece un gráfico que indica constantemente la velocidad de la conexión.

4.4. Estudio práctico. Resultados

La aplicación GNOME-GPRS ha sido diseñada no sólo para ofrecer una solución para los usuarios finales, sino también para evaluar la red GPRS. Así, las pruebas evalúan tanto la aplicación, como la propia red GPRS.

El módem (teléfono) utilizado para las pruebas es un Motorola c353, un teléfono multislot clase 8 (4+1), por lo que utiliza 4 ranuras de tiempo del canal para la de bajada y una para la subida. Los dispositivos que se comercializan en la actualidad son de este tipo. A continuación se explica cada una de las cuatro pruebas y el resultado

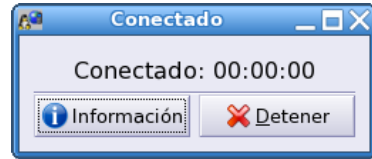


Figura 4.5: Aspecto al conectarse

obtenido en cada una de ellas.

4.4.1. Caso 1. Tráfico constante sin QoS

El primer caso que se plantea en este estudio práctico es el análisis de una conexión con un tráfico lo más invariable posible. En principio, la transferencia de un fichero de gran tamaño, es uno de los casos en los que la velocidad se muestra más constante. En esta conexión, se ha descargado un fichero, con un tamaño aproximado de 1 MB. La gráfica resultante de esta conexión se muestra en la figura 4.6. El eje Y indica el valor de la velocidad medida en kbps, mientras que el eje X es el tiempo. La velocidad de bajada mantiene una situación más o menos estable alrededor de los 45 kbps, con subidas hasta los 56 kbps, mientras que la velocidad de subida se mantiene durante toda la conexión por debajo de los 5 kbps.

Nos fijaremos primero en la velocidad de subida que es mucho menor con respecto a la de bajada. Por las características del terminal, esto es lógico, ya que el teléfono que se utiliza para las pruebas utiliza tan sólo una rodaja de tiempo en la subida frente a cuatro en la bajada.

Con respecto a la velocidad de descarga, las variaciones que se observan, son propias del tráfico *a ráfagas* de Internet, que también se verán acentuadas al transmitir sobre una red GPRS, veamos porqué. Las características del interfaz aéreo de GPRS hacen que el tráfico de varios usuarios puedan compartir los recursos de una misma ranura de tiempo del canal, algo que no ocurre en GSM. Además, el tráfico generado por las llamadas (GSM) es priorizado frente al de datos GPRS, en el sentido de que ocupa en primer lugar los recursos a menos que estos estén reservados explícitamente para GPRS.

Para los usuarios de GPRS, las rodajas de tiempo del canal que no estén siendo utilizadas por conexiones GSM, son recursos disponibles que podrán compartir. Esto hace que la velocidad alcanzada por una sesión de datos GPRS dependa mucho del resto de usuarios que tienen necesidad de transmitir en la misma célula de cobertura, sobre todo los que lo hacen a través de GSM. La velocidad puede variar mucho en conexiones GPRS, ya que en un momento dado, cuando hay muchos usuarios de GSM conectados, la velocidad que alcanzaremos será baja debido a que las ranuras de tiempo del canal están ocupadas en esas llamadas. Sin embargo, en un momento inmediatamente posterior, una de esas llamadas termina liberando así una de las ranuras ocupadas, que será utilizada por la conexión de datos GPRS que podrá aumentar la velocidad, hasta que se produce otra llamada GSM, que *coge* de nuevo la ranura de tiempo del canal, reduciendo otra vez la velocidad de GPRS, y produciendo la misma situación.

La velocidad alcanzada en esta prueba,

está en torno a los 45 kbps (con picos hasta los 56 kbps). Antes se comentó que GPRS podía llegar a los 171,2 kbps. Esta velocidad se consigue si no hubiera otros usuarios en la misma célula con necesidades de conexión, si las 8 rodajas de tiempo del canal son monopolizadas por el mismo usuario, y si no se usa un esquema demasiado protector.

Esquema	Bit Rate	Datos usuario
CS-1	9,05 kbps	8 kbps
CS-2	13,4 kbps	12 kbps
CS-3	15,6 kbps	14,4 kbps
CS-4	21,4 kbps	20 kbps

Para el terminal que se está utilizando, cada ranura de tiempo del canal tiene una capacidad aproximada de 14,4 kbps ya que se está utilizando el esquema CS3. Así, la máxima capacidad del dispositivo es de 57,6 kbps de bajada y 14,4 kbps de subida. Según la gráfica alcanzamos una velocidad de 56 kbps, por lo que se puede decir que la aplicación realiza bien su trabajo y que la red nos ofrece el servicio que se esperaba.

4.4.2. Caso 2. Tráfico constante con QoS

Veamos si la activación de la QoS [10] influye para mejorar las prestaciones de la conexión comparando los resultados con el caso anterior. La situación es similar, es decir, se realiza la descarga del mismo fichero y la configuración únicamente cambia con la activación de la QoS. En la la tabla 4.8 se indican la QoS solicitada y la QoS mínima negociadas.

La gráfica obtenida en este caso (figura 4.7) es similar a la del caso anterior, con la

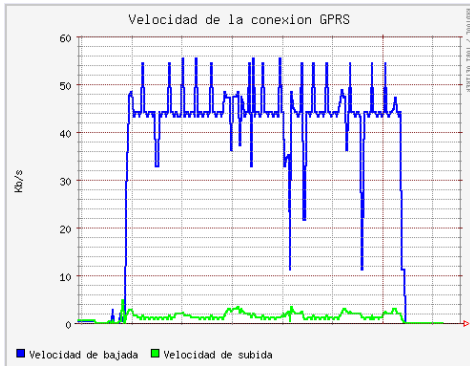


Figura 4.6: Gráfica resultante para el Caso 1

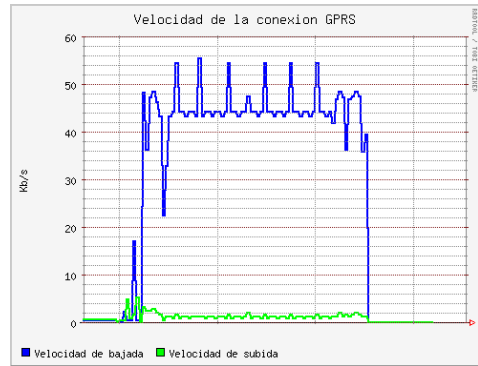


Figura 4.7: Gráfica resultante para el Caso 2

	QoS mínima	QoS solicitada
Prioridad	Clase1	Clase1
Retardo	Clase3	Clase1
Fiabilidad	Clase3	Clase1
Thr. máximo	Max 256 kbps	Max 2048 kbps
Thr. medio	111 kbps	111 kbps

Figura 4.8: QoS solicitada y QoS mínima negociadas

diferencia de que los picos bajos se reducen. Esto es lógico, ya que según la QoS negociada, la red nos puede ofrecer una velocidad media de 111 kbps, por lo que durante la conexión, tendremos los recursos del canal necesarios para alcanzar esta velocidad. Aunque se ha negociado un throughput medio de 111 kbps y una velocidad de pico de hasta 256 kbps, el dispositivo no permite alcanzar esa velocidad. Esto hace que no se pueda llegar a ninguna conclusión acerca del efecto de la QoS en la velocidad, ya que por mucho que la red nos ofrezca, no pasaremos de los 57,2 kbps que tiene de límite el móvil. El límite de la velocidad viene impuesto por el dispositivo y no por la red GPRS.

4.4.3. Caso 3. Tráfico a ráfagas sin QoS

Una vez analizado el caso de una conexión con tráfico *constante*, veamos que ocurre el tráfico *a ráfagas* típico de Internet. La gráfica obtenida aparece en la figura 4.9. Esta es distinta a las anteriores, ya que tanto el tráfico de bajada como el de subida, se muestran muy cambiantes, teniendo los picos en los momentos en los que se carga la página web, y descansando en los momentos siguientes, para volver a otro pico que significa que otra web es solicitada. La velocidad de subida es mucho más alta que en los dos primeros casos. Esto es porque entonces, tan sólo había que enviar a la red asentimientos y algún mensaje de control. En este caso transmitimos otros datos como los nombres de usuario y contraseñas del correo electrónico (que si van cifradas ocupan más tamaño), mensajes que se escriben en foros, datos de formularios, juegos online, etc. Es decir, al tener que interactuar en

muchas páginas web, la información que se envía a la red es mayor que en los casos anteriores.

4.4.4. Caso 4. Tráfico a ráfagas con QoS

Ahora se genera el mismo tráfico que en el caso anterior, pero se activa la QoS con los valores de la tabla del caso 2.

La gráfica que se muestra en la figura 4.10 es muy similar a la anterior (figura 4.9). Después de varias pruebas con esta configuración se ha comprobado que los picos altos suelen ser más frecuentes. Esto es lógico ya que al tener activada la QoS, la red nos debe *asegurar* los recursos suficientes (ranuras de tiempo del canal) para alcanzar la velocidad media y de pico negociada, que son de 111 y 256 kbps respectivamente. Aún así, como tan sólo podemos alcanzar 57,2 kbps (como se ha ido explicando), las diferencias con el caso anterior no son muy significativas.

4.5. Conclusiones

Desde la aparición de GSM, y el posterior desarrollo de GPRS y UMTS, las tecnologías móviles han ido integrando nuevos servicios a medida que se han ido necesitando, partiendo de la infraestructura existente. Debemos ver el avance en el campo de las comunicaciones móviles como una *evolución* y no *revolución*.

Desde el punto de vista práctico, al no poder explotar todas las capacidades de la red, por no existir dispositivos lo suficientemente potentes, no se ha podido comprobar cual es el límite real que pone la red GPRS.

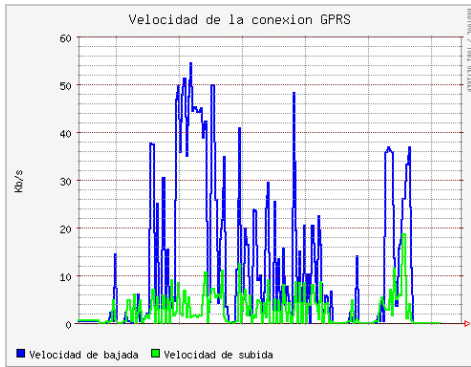


Figura 4.9: Gráfica resultante para el Caso 3

El dispositivo que tenemos, permite un throughput máximo de 57,2 kbps según se explicó en el caso 1 del estudio de resultados. Esta velocidad no permite que *expresamos* las capacidades de la red GPRS para ver cuando comienza a flaquear, que hubiera sido lo deseable, pero se han llegado a unas conclusiones que coinciden totalmente con los aspectos teóricos de la tecnología, y que de alguna forma, han validado la aplicación desarrollada ya que genera los resultados esperados.

A pesar de estos resultados, hay que tener en cuenta la complejidad de la tecnología con la que GNOME-GPRS realiza la conexión a Internet. GPRS depende de muchos factores que dificultan la interpretación de los resultados, y la mayoría de ellos son externos e imposibles de controlar por nosotros. Número de usuarios que están registrados en la misma célula, número de recursos del canal asignados en cada momento, efecto que se produce en los parámetros de la QoS al ir conectándose nuevos usuarios, y otras situaciones incontrolables durante la realización de este proyecto,

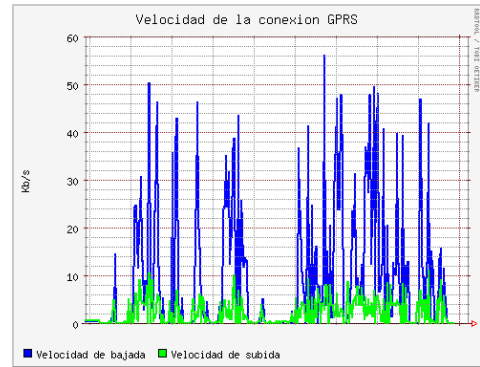


Figura 4.10: Gráfica resultante para el Caso 4

permitirían que el análisis de los resultados fuera mucho más completo.

Esta complejidad hace que un proyecto de este tipo ayude a los desarrolladores a investigar las tecnologías de última generación de comunicaciones móviles. Un proyecto basado en estos sistemas no debe tener fin, ya que estas tecnologías no dejarán de evolucionar, y los sistemas Linux, deben estar dotados de estas posibilidades. Así, tanto la investigación realizada como el desarrollo (GNOME-GPRS) completan un proyecto que puede y debe ser la base para otros estudios posteriores encaminados a UMTS. Como trabajo futuro también se propone realizar nuevos experimentos que puedan aportar diferencias entre el uso de conexiones con y sin QoS, apoyadas con un estudio estadístico de los resultados.

Bibliografía

- [1] Web del proyecto AGILA: <http://gitaca.unex.es/agila>. Página del Grupo de Ingeniería Telemática Aplicada y Comunicaciones Avanzadas de la Universidad de Extremadura.
- [2] 3GPP (3rd Generation Partnership Project) es una organización encargada del mantenimiento y el desarrollo de las especificaciones técnicas de los sistemas de tercera generación (3G) basados en el núcleo de red de GSM y en su sistema de radio enlace. <http://www.3gpp.org/>
- [3] J. Bannister, P. Mather, S. Coope *Convergence Technologies for 3G Networks: IP, UMTS, EGPRS and ATM* John Wiley & Sons (2004)
- [4] ETSI TS 27.007 *Digital cellular telecommunications system (Phase 2+); Universal Mobile Telecommunications System (UMTS); AT command set for User Equipment (UE)*.
- [5] C. Bettstetter, H-J Vögel, J. Eberspächer. *GSM Phase 2+. General Packet Radio Service. GPRS: Architecture, Protocols, and Air Interface*. Universidad de Munich. 1999.
- [6] ETSI TS 07.07 *Digital cellular telecommunications system (Phase 2+); AT Command set for GSM Mobile Equipment (ME)*.
- [7] ETSI TS 27.107 *3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Quality of Service (QoS) concept and architecture*.
- [8] C. Anderson *GPRS and 3G Wireless Applications* John Wiley & Sons (2001)
- [9] R.Moya, A. Del Castillo, R. Majadas, G. Oriol, G. Robles, Y. Moreno, A. Cazorla, G. Poo-Caamaño, A. Sánchez Acosta, R. Pérez Cubero, C. Garnacho, J. Pereira, A.Santiago, A.Valdés, C. Saavedra. *Programación en el entorno GNOME* The GNOME foundation. (2002)
- [10] D. Staehle, K. Leibnitz, K. Tsipotis. *QoS of Internet Access with GPRS*. Universidad de Wurzburg. 2002.

Ruby on Rails y otros entornos MVC de licencia libre: Un análisis práctico

Carlos Alberto Paramio Danta

carlosparamio@gmail.com

Asociación Campogibaltareños Entusiastas del Software Libre

Resumen

Este artículo pretende ser un análisis comparativo de los distintos entornos de programación de licencia libre disponibles que siguen el patrón de diseño *MVC* (Modelo-Vista-Controlador), y que facilitan el desarrollo de aplicaciones web.

Dicho patrón se ha venido utilizando desde 1979, pero tras la aparición del conocido entorno *Ruby on Rails* se ha popularizado enormemente, originando el desarrollo de alternativas que se fundamentan en otros lenguajes distintos a *Ruby*, así como promoviendo indirectamente el uso de otros ya existentes.

5.1. Introducción: El patrón MVC

El patrón de diseño *Modelo-Vista-Controlador* [1] fue descrito por primera vez en el año 1979 por Trygve Reenskaug, de los laboratorios de I+D de Xerox. Su característica primordial es la separación de la aplicación en tres componentes distintos, a saber: el modelo de datos, la interfaz del usuario, y la lógica de control de la aplicación. De esta manera, un cambio en cualquiera de estos componentes (como, por ejemplo, una nueva interfaz) no afecta demasiado al resto de código que conforma la aplicación.

El desarrollo de una aplicación mediante MVC implica por tanto definir tres componentes interconexos:

- Modelo: Representa el estado de la

aplicación, tanto el temporal como el permanente. No incluye sólo los datos, sino que también guarda información acerca de las reglas a cumplir por dichos datos, para que los otros componentes no puedan invalidarlos. Por ejemplo, si en la implementación de una tienda virtual los costes de envío no son aplicables para los artículos en promoción, será el modelo el que contenga esta restricción.

- Vista: Genera la interfaz de usuario. Esta interfaz usa normalmente datos del modelo, como por ejemplo la lista de artículos pendientes de envío que puede verse desde una página de la interfaz administrativa de una tienda virtual. La vista carga los datos del modelo y los muestra con el formato definitivo al usuario de la aplicación.
- Controlador: Contienen la lógica de la aplicación. Actúan recibiendo los órdenes del usuario (normalmente a través de una vista), interactúan con el modelo de datos, y responden mostrando un resultado concreto a través de una vista.

Las ventajas de este diseño frente a la muy usada programación monolítica son:

- La separación en componentes nos permite su implementación por separado, de manera que el producto final es más sencillo de elaborar por un equipo de desarrollo al dividir plenamente las tareas necesarias para llevarlo a cabo.

- La conexión entre componentes se realiza mediante un API (*Application Programming Interface*) bien definido, siendo posible el reemplazo de cualquiera de ellos sin que afecte al resto.

El esquema más típico de una aplicación diseñada con MVC es el que se observa en la figura 5.1. Varios controladores acceden y manipulan un mismo modelo de datos, y existen varias vistas para los datos del modelo que cambian cuando lo hace el estado del modelo.

Conocidos ya los fundamentos de este patrón de diseño, en las siguientes secciones se hará un brevísimo recorrido por algunos entornos de desarrollo MVC de licencia libre para el desarrollo de aplicaciones web, terminando con la revisión técnica de uno de los más populares en la actualidad: Ruby on Rails.

5.2. Apache Struts

Uno de los entornos más veteranos es Struts. Forma parte del colectivo de proyectos de la Apache Software Foundation, y tiene más de siete años de desarrollo y mantenimiento bajo sus espaldas. Su arquitectura está basada en *Model 2*, una variante del patrón MVC. En julio del 2005 había algo más de 2800 personas suscritas a la lista de correo de usuarios de Struts, lo que da una idea bastante buena de su grado de utilización. Está desarrollado en Java, siendo la lógica de la aplicación escrita en un servlet y las vistas en JSP (Java Server Pages).

En la actualidad, Apache Struts está compuesto en realidad de dos entornos de

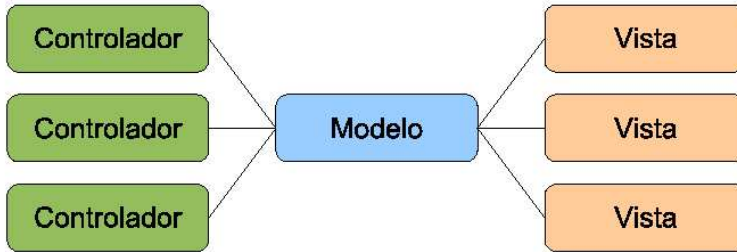


Figura 5.1: Esquema típico de MVC

programación y algunos otros subproyectos: Struts Action Framework y Struts Shale Framework, siendo el primero el que deriva del proyecto original, con una lógica de control basada en las peticiones del usuario; y el segundo una variante que usa JavaServer Faces para construir la interfaz de la aplicación, convirtiéndose en un entorno orientado a la definición de componentes reutilizables. Es conveniente destacar que, efectivamente, son dos entornos separados, cada uno con su propio ciclo de desarrollo, y por tanto no tiene sentido considerar a Shale como una nueva versión de Struts que viene a reemplazarlo.

Al estar escrito en Java, es posible usar cualquier motor de base de datos al que pueda accederse desde JDBC para el almacenamiento del estado permanente del modelo de datos.

Existen varias herramientas de modelado visual para Struts, tales como VisualStruts, Camino, StrutsConsole, etc., que facilitan aún más el desarrollo rápido de aplicaciones.

Por último, comentar que hay disponible muchísima documentación para el desarrollo con Struts, así como publicaciones específicas sobre el tema.

Se puede encontrar mucha más información en la página del proyecto [2].

5.3. Symfony

Este entorno MVC está escrito en PHP5, y es ya lo suficientemente maduro como para considerarlo una alternativa eficaz. Usa fragmentos de código de Mojavi, otro entorno MVC escrito en PHP. Entre sus muchas características destacan la automatización de procesos como la internacionalización, la validación de formularios, la administración de la caché, *scaffolding* (generación automática del modelo y un controlador con las operaciones básicas para operar con el mismo, a partir de un esquema de definición de datos), administración de un carrito de la compra, interacciones con la aplicación mediante AJAX (*Asynchronous JavaScript And XML* [3]), tests de unidad (porciones de código que comprueban el buen funcionamiento de los modelos), tests funcionales (porciones de código que comprueban el buen funcionamiento de los controladores) y muchas otras.

El esquema de datos del modelo se describe en lenguaje XML, y se dispone de he-

ramientas para generar las clases correspondientes a ese modelo, así como un fichero con las sentencias SQL necesarias para construir las tablas donde se almacenarán los datos permanentes del modelo. Symfony es compatible con la mayoría de los principales motores de bases de datos (MySQL, PostgreSQL, Oracle, MSSQL, SQLite, y otras compatibles para Creole, la capa de abstracción de la base de datos).

Los archivos de configuración del entorno están escritos en YAML, un lenguaje de marcas mucho más simple que XML, sencillo de escribir y de analizar sintácticamente.

Es posible definir diferentes entornos de ejecución, de modo que pueden distinguirse las distintas configuraciones según estemos usando la aplicación durante el desarrollo, o bien cuando ya esté en producción, o cuando se están ejecutando los tests de unidad. Cada uno de estos entornos pueden usar, por ejemplo, una conexión a una base de datos concreta, distinta para cada uno de ellos.

Más información en la página del proyecto [4].

5.4. Django

Como no podía ser de otra forma, también existe ya un entorno MVC para el lenguaje de programación Python. Django comparte muchas de las características presentes en Ruby on Rails, tales como *scaffolding*, un vinculador de objetos con la base de datos relacional (ORM), un sistema de plantillas, convención frente a configuración, un pequeño servidor web integrado para probar código durante el desarrollo, etc. Estas características serán descritas

convenientemente en la siguiente sección.

El modelo se describe en su totalidad en Python, y gracias a esto y al ORM, se pueden escribir porciones completas de código sin utilizar una sola sentencia de SQL para el acceso a los datos.

Con el entorno de desarrollo disponemos también de una shell interactiva desde la que invocar funciones del mismo, crear nuevas instancias de alguno de nuestros modelos, y en general poder probar el API de nuestra aplicación.

Posee una comunidad de usuarios bastante holgada, y una excelente documentación.

Una vez más, información completa en la página del proyecto [5].

5.5. Ruby on Rails

Rails se ha convertido en poco tiempo en el entorno MVC que más está llamando la atención de los desarrolladores de aplicaciones web de todo el planeta. El lenguaje de programación en el que se escriben dichas aplicaciones en este caso es Ruby, un moderno lenguaje de *scripting* orientado a objetos. El código escrito en Ruby suele ser conciso y próximo al lenguaje natural, lo cual facilita tanto su escritura como su lectura posterior (por el mismo desarrollador o por otros que conforman el grupo de trabajo).

El diseño de Rails se ha basado principalmente en dos conceptos, a saber:

- **DRY, o *Don't Repeat Yourself***: no necesitamos especificar más de una vez un mismo concepto del sistema, como por ejemplo la definición de datos del modelo que se usará tanto en

las instancias de la clase como en la tabla de la base de datos.

- Convención por encima de configuración. Es decir, existe un comportamiento por defecto para muchas de las características a tener en cuenta durante el desarrollo de nuestra aplicación, de manera que no es necesario especificar dicha configuración en un fichero aparte (si bien es posible hacerlo en caso de querer obviar dichas convenciones para usar unas propias).

Otras características interesantes de Rails son los generadores de código, el soporte integrado para servicios web (*web services*), la mejora de la interacción mediante AJAX, o los tests funcionales y de unidad. Y, en general, es posible tratar con las tareas más típicas en el desarrollo de una aplicación web de una manera muy directa, permitiendo que nos concentremos en la lógica principal de la aplicación.

La bibliografía sobre Rails es, por el momento, escasa, aunque ya se ha publicado un libro [6] dedicado exclusivamente a este tema, y en breve aparecerán al menos otros dos títulos.

Veamos algunos de los componentes de Ruby on Rails que hacen posible esto último.

5.5.1. Active Record

Como se ha visto, lo normal en un entorno MVC es que los datos permanentes del estado del modelo se almacenen en una base de datos relacional. Los modelos usados en la aplicación están escritos en un lenguaje de programación orientado a objetos, y la

conversión de un modelo relacional a otro orientado a objetos (y viceversa) no es en absoluto directa. La manera tradicional de operar con la base de datos es usar funciones de acceso a la misma, utilizando SQL como lenguaje de operaciones. A veces, se usa un envoltorio para el acceso a la base de datos, de manera que la aplicación pueda hacerse independiente del motor de base de datos utilizado en la implementación. Existe, no obstante, una solución mejor, que consiste en proveer de un mecanismo para vincular las tablas de la base de datos a clases en el lenguaje usado. Este mecanismo se denomina *Object/Relational Mapping* (ORM), y es justo lo que utiliza Ruby on Rails.

Supongamos que estamos generando una aplicación con Ruby on Rails para la administración de un *blog* o bitácora personal. En la base de datos, habremos definido una tabla para almacenar los envíos, llamada *posts*. En nuestra aplicación, tenemos una clase que define el modelo de datos llamado *Post*. Pues bien, cada una de las entradas (filas) de la tabla corresponde con una instancia de nuestra clase *Post*. Las propiedades del objeto **Post** corresponden con las columnas de la tabla *posts*. Para poder operar con cada una de estas instancias, y así modificar su estado (o lo que es lo mismo, alterar el contenido de la base de datos), dispondremos de una serie de métodos para obtener y modificar las propiedades de las mismas. Incluso estarán disponibles algunos métodos para operar con la tabla, como puede ser el método **find** para buscar un registro dentro de la tabla. Este método nos devolvería un objeto o un grupo de objetos de tipo **Post**. O el método **save** para guardar los cambios en la fi-

la correspondiente de la tabla. Por ejemplo, podríamos usar las líneas de código que se observan en la figura 5.2 para mostrar los envíos realizados por Pedro, al mismo tiempo que cambiamos el autor de dichos envíos a Ana.

Otra característica importante de ActiveRecord es un soporte sofisticado para la validación de datos provenientes de los formularios. Basta con agregar una línea a la descripción del modelo en cuestión para realizar las comprobaciones más comunes. Algunas de ellas son, por ejemplo:

```
validates_presence_of
validates_numericality_of
validates_uniqueness_of
validates_format_of
validates_length_of
validates_inclusion_of
validates_each
```

Como muestra, si por ejemplo quisiésemos añadir la restricción al modelo anterior *Post* de que no sea posible guardar un envío con los campos *title* o *body* vacíos, y que además los envíos tengan forzosamente un título único, añadiríamos las siguientes sentencias a la clase:

```
class Post < ActiveRecord::Base
  validates_presence_of :title, :body
  validates_uniqueness_of :title
end
```

5.5.2. Action Pack

Este componente se encarga del soporte para las vistas y los controladores en Rails.

Las vistas no son más que plantillas HTML con porciones de código dinámicas escritas en *Embedded Ruby* (ERb). Para no romper con el modelo MVC, es conveniente poner atención en no incluir código en

Ruby dentro de la plantilla que forme parte de la lógica de la aplicación. La extensión de los ficheros plantilla es *rhtml*. Las etiquetas usadas para marcar código en Ruby dentro de las plantillas son:

`< % >` : Ejecuta el bloque de código situado entre las etiquetas.

`< %= % >` : Ejecuta el bloque de código situado entre las etiquetas, e inserta el resultado de la ejecución en esa posición.

Los controladores contienen la lógica de la aplicación. Sirven los datos necesarios a las vistas, y reciben los eventos de las páginas generadas por las vistas. Son por tanto el centro neurálgico del programa, e interconectan al usuario, al modelo y a las vistas. Las acciones definidas por un controlador se vinculan fácilmente con una dirección web, de manera que el usuario puede invocarlas simplemente escribiendo la dirección correspondiente. El formato de dirección típico con el comportamiento por defecto de Action Pack es el que se muestra en la figura 5.3.

El nombre de la acción corresponde con el nombre de un método definido en la clase controlador. En el caso de la URL de la figura, Rails buscaría un método denominado **edit_post** definido en la clase *admin*, que hereda de *ApplicationController*. El código correspondiente se puede observar en la figura 5.4.

5.5.3. Creando un blog

Veamos un pequeño ejemplo al completo de una aplicación sencilla con Ruby on

```

class Post < ActiveRecord::Base
end

Post.find(:all, :conditions => "name='Pedro'") do |post|
  puts post.title
  puts post.body
  post.author = "Ana"
  post.save
end

```

Figura 5.2: Ejemplo de recuperación y modificación de datos con ActiveRecord



Figura 5.3: Formato de URL

```

class AdminController < ApplicationController
{
  def edit_post
    # Código correspondiente a la acción
  end
}

```

Figura 5.4: Ejemplo de definición de acción

Rails. El objetivo es obtener la interfaz administrativa de un *blog* o bitácora, que deberá permitir la introducción de noticias desde un formulario con algunas restricciones.

Lo primero es generar el esqueleto de la aplicación. Rails dispone de un *script* para crear la estructura de directorios donde se almacenarán los componentes de nuestro programa, así como la inicialización de algunos archivos (como la configuración de acceso al motor de la base de datos, o el controlador principal de la aplicación). Basta con ejecutar dicho *script* pasando como parámetro el nombre de la aplicación, y obtendremos el esqueleto:

```
$ rails blog
  create
  create  app/controllers
  create  app/helpers
  create  app/models
  create  app/views/layouts
  create  config/environments
  ...
```

A continuación, vamos a crear una tabla en nuestro motor de base de datos que contendrá los envíos que se realicen al *blog*. Llamaremos a esta tabla *posts*. Guardaremos las instrucciones SQL en **db/create.sql**.

```
drop table if exists posts;
create table posts (
  id int not null auto_increment,
  title varchar(255) not null,
  body text not null,
  primary key (id)
);
```

Pasamos este fichero SQL a nuestra base de datos, y editamos el fichero **config/database.yml** para modificar los parámetros necesarios: adaptador, base de datos, nombre de usuario para acceder a la

misma, y contraseña. Como se podrá observar, existen tres entornos de ejecución distintos: *development*, *test* y *production*. El primero es el que usaremos en el ejemplo, y es utilizado durante el desarrollo para comprobar el funcionamiento de la aplicación; el entorno *test* es usado por los tests funcionales y de unidad, porciones de código que realizan comprobaciones con los modelos y controladores de nuestra aplicación; y el entorno *production* es, como su propio nombre indica, el usado durante la explotación del programa.

A continuación, vamos a pedir a Rails que genere para nosotros un modelo con el que vincular la tabla *posts*. Además, aprovecharemos para pedir a Rails que nos genere un controlador para realizar las operaciones básicas con dicho modelo de datos, es decir, CRUD (*Create, Recover, Uptate and Delete*). Usaremos un *script* generado con el esqueleto de la aplicación para tal fin:

```
$ script/generate scaffold Post Blog
```

Si ahora utilizamos el servidor web incorporado por Rails, WEBrick, para acceder a nuestra aplicación, podremos observar toda una interfaz para comenzar a crear nuestras primeras entradas de *blog*. Arrancamos el servidor web:

```
$ script/server
```

Y accedemos con un navegador a la URL <http://localhost:3000/blog> para ejecutar la acción por defecto del controlador denominado *blog* que acabamos de crear.

Podemos personalizar las vistas generadas por Rails para modificar el aspecto

de la aplicación. Al generar el controlador *Blog*, Rails nos habrá creado una carpeta en **app/views** con el nombre del mismo, y dentro de ella encontraremos una serie de plantillas de extensión `rhtml` que contienen las etiquetas HTML con código ERb que conforman las vistas de las distintas acciones del controlador. Modificando las plantillas, cambiamos la interfaz del usuario.

5.6. Conclusiones

Definitivamente, el patrón de diseño MVC está bastante extendido, y son muchos los entornos de desarrollo que han ido apareciendo basados en el mismo. Ruby on Rails se ha convertido en poco tiempo en uno de los entornos más populares debido a su excelente comportamiento, y a que hace que el desarrollo se agilice muchísimo gracias a sus características de convención frente a configuración, a su ORB, y a sus numerosas funciones de ayuda. Esto permite que pueda emplearse mucho más tiempo de programación a la implementación de tests funcionales y de unidad, por ejemplo, que aseguren un mínimo de calidad en cada uno de los componentes del programa.

Para más información, se puede consultar la página del proyecto [7].

Bibliografía

- [1] Modelo Vista Controlador
<http://es.wikipedia.org/wiki/MVC>
Edición en español de Wikipedia, la enciclopedia libre.
- [2] Apache Struts Project.
<http://struts.apache.org/>
- [3] Asynchronous JavaScript And XML.
<http://es.wikipedia.org/wiki/AJAX>
- [4] Symfony project.
<http://www.symfony-project.com/>
- [5] Django project.
<http://www.djangoproject.com/>
- [6] Dave Thomas. David Heinemeier Hansson. *Agile Web Development With Rails*. The Pragmatic Programmers LLC. (2005)
- [7] Ruby on Rails Framework.
<http://www.rubyonrails.org/>

Implantando GNU/Linux: Una visión desde un Ayuntamiento

Juan Fernando Fernando Sánchez

cpd.fernando@sanroque.es

Departamento de Informática (CPD). Ayuntamiento de San Roque.

Resumen

El Ilustre Ayuntamiento de San Roque tomó la decisión en la legislatura del 2005 de impulsar el *Software Libre* [1]. Entre las diversas iniciativas, un punto fundamental es la implantación de sistemas GNU/Linux en las dependencias del Ayuntamiento. Este artículo explica las diversas soluciones empleadas y los problemas encontrados durante la implantación.

6.1. Usando GNU/Linux

A la hora de buscar alternativas viables basadas en GNU/Linux, hay que distinguir dos bloques fundamentales. El lado servidor y el lado cliente. Mientras que en el primer caso, los productos están muy evolu-

cionados, en el segundo se está empezado a encontrar soluciones parcialmente operativas para el usuario final.

6.1.1. Lado Servidor

En este apartado, el Ayuntamiento tiene completamente funcionales diversas soluciones y con otro pequeño grupo se están realizando pruebas finales para la puesta en marcha en entornos de producción. En concreto, se han implantado servidores de archivos, autenticación centralizada, SGDB, web corporativas, servidores de correo, cortafuegos, proxys y vpns.

6.1.2. Lado Cliente

De cara al usuario final, el funcionario de la administración local en nuestro caso, el perfil estándar en su relación con los procesos informáticos es el siguiente:

- Aplicaciones ofimáticas (Microsoft Office y Corel Wordperfect). En este apartado la mayoría de los documentos son escritos realizados en Word aunque también se encuentran hojas de cálculos y bases de datos access.
- Navegador de internet (Microsoft Internet Explorer)
- Lector de correo (Microsoft Outlook Express)
- Cliente de Terminal Server para acceso al servidor central donde a efectos prácticos para el usuario se trabaja como si fuera localmente con las aplicaciones de gestión: Padrón, Registro de E/S, Contabilidad, etc.

En este perfil estándar se encuentran excepciones con aplicaciones específicas como pueden ser programas de banca electrónica, diseño asistido por ordenador, etc. El planteamiento de trabajo en el CPD ha sido buscar alternativas a los tres primeros puntos.

La migración de las aplicaciones de gestión accedidas a través de Terminal Server, no es viable porque no existen en el mercado alternativas válidas y fiables destinadas a GNU/Linux. En este sentido los Organismos y Administraciones Públicas deberían establecer mecanismos y directrices para promover y fomentar la aparición en el mercado tecnológico de alternativas, no

solamente que funcionen en la plataforma GNU/Linux sino que además posean licencias GPL [3].

6.2. Servidor de archivos. Samba [3]

El objetivo es tener centralizados los documentos de usuarios en una máquina dotada con la capacidad hardware para tolerancia de fallos (sistemas RAID) y copias de seguridad en cinta. El acceso al sistema de archivos desde los puestos clientes debe ser independiente de plataforma (Microsoft Windows/Unix). El protocolo CIFS (antes denominado SMB) de archivos compartidos de Microsoft Windows es una solución estándar bastante simple de implantar. Samba [3] es una implementación robusta y de alto rendimiento para Unix de ese protocolo. La gestión avanzada de los permisos para los usuarios puede realizarse usando la capacidad de permisos extendidos con ACLs (listas de control de acceso) cuyo soporte para el sistema de archivos *ext3* ya viene incluido en el núcleo 2.6.

El sistema centralizado es un paso importante para lograr la movilidad de los usuarios, es decir, que un usuario pueda usar cualquier equipo de la red para trabajar con sus documentos.

No hay que ignorar los posibles problemas relacionados con puntos únicos de fallo (SPOF) y los cuellos de botella al estar los documentos ubicados en una máquina individual. Cualquier disfunción relativa a esa máquina afectará a todos los usuarios de la red. La infraestructura de red y los componentes redundantes deben ser analizados para minimizar los riesgos. Los sistemas distribuidos basados en clusters de alta disponibilidad también deberían ser es-

tudiados en profundidad.

6.3. Autenticación de usuarios centralizada. OpenLdap [4]

La autenticación de los usuarios en la red debe ser común a los usuarios de Microsoft Windows y GNU/Linux para evitar duplicidad de trabajos y posibles problemas de integridad con la gestión de los usuarios (altas / bajas / modificaciones). Esta centralización está ligada fuertemente con el acceso a los archivos de red para compartir también los permisos de accesos a esos elementos. OpenLdap [4] es una solución que permite la integración de la gestión de usuarios y la comunicación con Samba. De cara al uso en sistemas clientes Microsoft Windows se plantearon dos posibilidades.

- Instalar Samba como un controlador de dominio principal para resolver las peticiones de los usuarios a la hora de acceder al sistema desde los puestos clientes.
- Usar el módulo pGina [5] que se inserta dentro de la autenticación estándar de Microsoft Windows para permitir la consulta a un servidor OpenLdap centralizado.

La opción seleccionada y actualmente en fase de pruebas es la primera. El motivo de esta elección es que era menos intrusiva en los equipos clientes de Microsoft Windows y la ausencia de dificultades importantes durante las pruebas.

6.4. Sistema gestor de bases de datos. PostgreSQL [6]

Idénticos motivos que aconsejan la centralización de los documentos son los que conducen a una gestión centralizada de los datos de las aplicaciones de bases de datos. Toda esa información debe estar alojada en un sistema gestor de base de datos que se haga responsable de mantener la seguridad, integridad y coherencia interna. Se logra, por tanto, un mayor control y seguridad sobre la información, así como facilitar las tareas de copias de seguridad e integridad de datos. La elección de PostgreSQL como SGDB se debe a que es un proyecto con gran respaldo en la comunidad Open Source y que uno de sus principales objetivos es la fiabilidad de los datos.

6.5. Web, correo, cortafuegos, proxy y vpn

6.5.1. Web departamental

El servidor Apache [7] es utilizado para ofrecer una web en la intranet con las siguientes características:

- Desarrollo de aplicaciones web en Php 5 [8] y Java (Tomcat [9] + Springframework [10]).
- Noticias, wiki, foros y entornos colaborativos.

6.5.2. Correo corporativo

Qmail [11] fue la opción elegida a la hora de implementar el correo corporativo. Seguridad, fiabilidad y rendimiento fueron las razones principales para esa decisión.

Se han adoptado mecanismo para evitar el envío de spam. También se ha integrado con un antivirus (F-prot) mediante el Qmail-scanner [12]. No se ha llegado a poner en marcha ninguna solución para la recepción de spam por el problema de los falsos positivos.

6.5.3. Cortafuegos y proxy

El mecanismo provisto por el núcleo para el filtrado de paquetes Iptables [13] es una herramienta de gran potencia y fiabilidad a la hora de crear un cortafuegos. Por otro lado, Squid [14] es un proxy que permite aumentar el rendimiento almacenando en caché las páginas solicitadas y además activar el filtrado a nivel de url de las peticiones de los usuarios. Combinando los dos mecanismos se consiguen sistemas bastante robustos respecto a ataques remotos y de gran rendimiento.

6.5.4. Redes privadas virtuales

Como respuesta a la necesidad de conectar diferentes sedes con el edificio principal y disponiendo de conexiones de banda ancha a Internet se ha utilizado OpenVpn [15] para unir las de forma segura estableciendo redes privadas virtuales.

6.6. El lado cliente. Software de Internet

Tanto para navegar como para leer el correo se ha optado por los productos de Mozilla [16], Mozilla Firefox y Mozilla Thunderbird. Entre los motivos para esta elección cabe destacar la existencia de versiones tanto para Microsoft Windows como

GNU/Linux y que son productos sencillos y ágiles. La seguridad aportada por una alternativa en Microsoft Windows al Internet Explorer y Outlook Express es ya un factor que no se puede ignorar.

6.7. Paquete ofimático. OpenOffice 1.1.x [17]

Actualmente ya está disponible la versión 2.0 que mejora las características de muchos puntos, aunque la implantación inicial se realizó con la 1.1.x.

■ Ventajas

- Hay versiones multiplataforma que se pueden utilizar en Microsoft Windows y GNU/Linux.
- El formato OpenDocument [9] permite que los documentos no están vinculados a ninguna aplicación propietaria.
- Su funcionalidad es muy similar a productos propietarios existentes.
- Puede importar documentos de otros productos propietarios.

■ Inconvenientes

- Parecido pero no igual. Requiere formación para el usuario final. Además los documentos de texto importados necesitan retoques más o menos importantes. Esto puede convertirse en una ventaja añadida al aprovechar la migración para normalizar los diseños de los diferentes departamentos.

- El módulo de BD en la versión 1.1.x no implementa un *SGDB*. Esto se soluciona a partir de la versión 2.0, pero de todas formas se prefiere conectar el interfaz mediante *ODBC* al PostgreSQL centralizado. El problema fundamental es que a la hora de importar sólo se puede trabajar con los datos. El resto (formularios, informes) hay que recrearlo desde cero. También podría ser considerado como una ventaja para promover una depuración de las aplicaciones y estructuras de datos creados por los usuarios sin formación en Ingeniería del Software. De todas formas, los informes y los formularios avanzados requieren una mejora significativa en un futuro próximo.

6.8. GNU/Linux de escritorio

Surge el mismo problema de 'parecido pero no igual' lo que impone la necesidad de formación. El diseño del escritorio deber ser 'bonito' y fácil de usar, pero la usabilidad no debe entrar en conflicto con las prestaciones o la comodidad. Tras analizar las impresiones iniciales de los usuarios se constató que el entorno aportado por Kde es mucho más agradable para estos usuarios que el de Gnome.

6.8.1. Hardware a nivel de usuario

El usuario debe poder tratar con disquetes, CDs y memorias USB. Kde responde

bastante bien en este sentido.

6.8.2. Tareas del administrador

- Administración del hardware:
 - Impresoras locales y en red (cups)
 - Escaneres
 - Dispositivos inalámbricos
- Administración del software:
 - Integración con OpenLdap [4] y Samba [3]
 - Conexión via *ODBC* con el *SGDB* centralizado.

6.9. Fases de implantación

- Formación en Guadalínx 2004 [20] y OpenOffice 1.1 [17]
- Reemplazo de Internet Explorer, Outlook Express y Microsoft Office por Mozilla Firefox, Mozilla Thunderbird y OpenOffice.
- Instalación de GNU/Linux e integración con el servidor de archivos y OpenLdap. Se realiza puesto por puesto, primero con los equipos que no usan access ni otras aplicaciones específicas.

6.9.1. Escollos en la migración

- Navegador de internet

- Se encuentran algunos sitios donde el diseño de las páginas no es compatible ya que están basado en etiquetas y opciones no estandarizados.
- Bases de datos de clientes
 - Es necesario establecer un protocolo de migración. Normalizar la nomenclatura para los campos, y estructuras de datos. También hay que normalizar los tipos de datos.
 - Al importar los datos en bruto hay que tener en cuenta el juego de caracteres.
 - Las consultas, formularios e informes deben ser creadas desde cero. Falta documentación para formularios avanzados con tablas relacionadas y documentos vinculados y/o generados mediante fusión.
- En algunos puestos clientes existen aplicaciones propietarias específicas para Microsoft Windows
 - Banca electrónica
 - Diseño asistido por ordenador
 - Servicios Sociales

6.10. Distribución LinRoque [1]

- GNU/Linux es un sistema operativo que combina el núcleo Linux con aplicaciones del proyecto Gnu
- Debian [19] puede ser considerada como la principal distribución GNU/Linux

- Guadalinux 2004 [20]. Metadistro elaborada por la Junta de Andalucía y basada en Debian GNU/Linux. Tiene un mecanismo dual (Metadistros [21]) para ejecutarse en modo Live o instalarse en el equipo del usuario.
- LinRoque es una personalización de Guadalinux 2004 con Kde como escritorio por omisión, drivers wireless para algunos dispositivos USB, driver *ODBC* para PostgreSQL, soporte SATA, eliminación de los paquetes lúricos y de software *p2p* y algunos otros cambios menores.

6.11. Otras ideas

Entre otros puntos interesantes a estudiar a medio plazo citamos:

- Aulas y sitios públicos
 - Modo kiosko
 - Clusters y clientes ligeros
- Groupware
- Mensajería instantánea (Jabber)

6.12. Consideraciones legales

Todo administrador de sistemas en la administración pública española debe tener en mente el marco legislativo vigente y en concreto dos puntos fundamentales:

- Ley Orgánica de Protección de Datos de carácter personal (LOPD [22]). La información debe estar perfectamente localizada y el acceso a ella debe cumplir unos requisitos de seguridad.
- Ley de Propiedad Intelectual (LPI [23]). No se puede tener ningún software sin licencia.

las funcionalidades de bases de datos de Access.

El proceso de migración e implantación es una gran oportunidad para realizar una auditoría exhaustiva tanto de los datos sensibles a la LOPD como de los productos afectados por la LPI.

6.13. Conclusiones

El proyecto de implantar GNU/Linux en un entorno real de producción en la administración local no es una tarea imposible pero, sobre todo a nivel de usuario final, se encuentran obstáculos que requieren una importante inversión en recursos humanos para la migración completa de la funcionalidad. Durante bastante tiempo se mantendrán instalaciones parciales Microsoft Windows y GNU/Linux.

La Administración y Organismos Públicos deben apostar fuertemente en el desarrollo y promoción de aplicaciones con licencia GPL [3] que presenten alternativas a las aplicaciones propietarias y específicas. De igual forma toda administración pública debería asegurarse que el diseño web de sus portales sea independiente del navegador.

También son necesarias herramientas que faciliten la migración e importación de

Bibliografía

- [1] *Iniciativas del Ayuntamiento de San Roque* Información sobre las diversas medidas emprendidas para fomentar el uso del Software Libre, <http://cpd.sanroque.es/SL/>
- [2] *GPL* Licencia para crear aplicaciones de Software Libre con los cuatro grados de libertad. <http://www.gnu.org/licenses/gpl.html>
- [3] *Samba* Sistema de archivos en red mediante protocolo CIFS. <http://www.samba.org/>
- [4] *OpenLdap* Autenticación centralizada <http://www.openldap.org/>
- [5] *pGina* Integración de la autenticación de Microsoft Windows con Ldap. <http://pgina.xpasystems.com/>
- [6] *PostgreSQL* Sistema Gestor de Base de Datos Relacional. <http://www.postgresql.org>
- [7] *Apache* Uno de los servidores web más potentes disponibles. <http://httpd.apache.org/>
- [8] *Php 5* Lenguaje de programación orientado a la programación Web. <http://www.php.net/>
- [9] *Tomcat* Componente independizado del proyecto jakarta de Apache para ejecutar aplicaciones java via Web. <http://tomcat.apache.org/>
- [10] *Spring Framework* Librerías java interrelacionadas para facilitar el desarrollo de aplicaciones siguiendo el paradigma MVC (Modelo-Vista-Controlador) y la inversión de control (IoC). Posee un módulo orientado al desarrollo web. <http://www.springframework.org/>
- [11] *Qmail* Servidor smtp y pop3 de gran seguridad y fiabilidad. <http://www.qmail.org/>
- [12] *Qmail-scanner* Interceptor para qmail que permite ejecutar un programa por cada mensaje de correo procesado. <http://qmail-scanner.sourceforge.net/>
- [13] *Netfilter / Iptables* Herramientas del núcleo para implementar cortafuegos robustos. <http://www.netfilter.org/>
- [14] *Squid* Proxy-caché para optimizar el rendimiento y filtrar direcciones. <http://www.squid-cache.org/>
- [15] *OpenVpn* Aplicación para establecer redes privadas virtuales tanto desde Microsoft Windows como desde GNU/Linux. <http://openvpn.net/>
- [16] *Mozilla* Sitio web de la organización Mozilla. <http://www.mozilla.org/>

- [17] *OpenOffice* Paquete ofimático altamente funcional. <http://www.openoffice.org/>
- [18] *OpenDocument* Estándar de OASIS para especificar un formato público basado en XML de documentos ofimáticos. <http://tinyurl.com/quotk>
- [19] *Debian* Distribución GNU/Linux considerada paradigma del OpenSource. <http://www.debian.org/>
- [20] *Guadalinux 2004* MetaDistribución basada en Debian y desarrollada por la Junta de Andalucía. <http://www.guadalinux.org/>
- [21] *MetaDistros* Proyecto español para la generación de distribuciones que pueden ejecutarse en modo Live y/o instalarse en el disco duro. <http://metadistros.hispalinux.org/>
- [22] *LOPD* Ley Orgánica 15/1999, de Protección de Datos de Carácter Personal. <http://tinyurl.com/dsrye> (pdf)
- [23] *Reforma del Código Penal* Ley Orgánica 15/2003, de 25 de noviembre, que reforma el código penal y endurece las penas relacionadas con delitos sobre la Propiedad Intelectual. <http://tinyurl.com/dwfgda>

TOMAS³: Towards an Open Management Architecture for Systems, Software and Services

TOMAS³: Hacia una arquitectura abierta para sistemas, *software* y servicios

Ismael Herrero Rodríguez

ismael.herrero@andago.com

María Isabel Martín

José Gato

Álvaro del Castillo

Rafael García Leiva

Ándago Ingeniería

Marta Patiño

Ricardo Jimenez-Peris

Álvaro Orús Cacho

José Antonio Sánchez

Universidad Politécnica de Madrid.

Resumen

En el presente documento se describe TOMAS³, un estándar de código abier-

to para el despliegue, instalación, confi-

guración, administración, monitorización y mantenimiento de medianas y grandes redes de ordenadores bajo el sistema operativo GNU/Linux. Las herramientas descritas permiten la total administración y operación de una red de ordenadores reduciendo sustancialmente los costes de operación.

7.1. Introducción

Ándago Ingeniería es una empresa del grupo Talde dedicada a servicios de consultoría que trabaja en proyectos de gran envergadura, utilizando para ello tecnologías de *Software Libre*.

Recientemente Ándago Ingeniería ha decidido dotarse de un departamento de Investigación, Desarrollo e Innovación, que junto con la colaboración de varias empresas privadas e instituciones universitarias investiga en nuevas tecnologías para avanzar en el desarrollo del *software* y ofrecer productos punteros y de valor añadido a sus actuales y futuros clientes.

Uno de esos proyectos es la creación de un estándar y una solución completa y abierta para la gestión efectiva de grandes redes de ordenadores corriendo sistemas operativos **nix*, principalmente Linux. Se trata del proyecto *TOMAS³: Towards An Open Management Architecture for Systems, Software and Services*.

El proyecto *TOMAS³* pretende fomentar el uso del *software* libre de dos formas: primero integrando en su arquitectura soluciones de código abierto, aunque no limitándose a éstas, y segundo basando la gestión del proyecto en herramientas de desarrollo colaborativo que permiten una libre y efectiva diseminación del conocimiento ad-

quirido durante el desarrollo del proyecto a toda la comunidad.

Asimismo se pretende avanzar en el desarrollo de estándares utilizándolos para el diseño de la arquitectura de *TOMAS³*, como la modelización por medio de CIM [7] del DMTF [6] y la aplicación de servicios web via WSDM [8] de OASIS [9] entre varios otros estándares.

Además de ello la intención de Ándago respecto a *TOMAS³* es la de liberar el resultado del proyecto en el dominio público a fin de implicar en su desarrollo a una comunidad de desarrolladores y usuarios interesados en el mismo para su permanente soporte, extensión y mejora, haciendo el resultado del proyecto útil y duradero en el tiempo. Actualmente tanto las fuentes como la documentación del proyecto son de acceso público, pero no está permitida su modificación. El equipo de *TOMAS³* pretende presentar a la comunidad un proyecto terminado y utilizable antes de comenzar su distribución.

TOMAS³ es un proyecto PROFIT [4] del Ministerio de industria, turismo y comercio [2] y del Ministerio de educación y ciencia [3].

Conjuntamente a Ándago están participando en el desarrollo de *TOMAS³* las siguientes entidades:

- el Centro de Referencia Linux [1] de la Universidad Autónoma de Madrid
- el Laboratorio de Sistemas Distribuidos [5] de la Universidad Politécnica de Madrid

que unen a los aportados por Ándago recursos de investigación. Algunas empresas

privadas también han mostrado su interés en el proyecto.

En este artículo se presentan los requisitos establecidos por el equipo de TOMAS³ para ser cumplidos por la aplicación.

Una red de ordenadores tipo *cluster* puede contener una gran variedad de equipos, con distintas características *hardware* y con funciones muy dispares. Ejemplos de los equipos que podemos encontrar en este tipo de redes son:

- Servidores de ficheros.
- Servidores de correo.
- Nodos de cálculo donde se ejecutan trabajos.
- Nodos maestro que coordinan el trabajo de dichos nodos.
- Servidores de almacenamiento de datos, como servidores de disco (NFS, AFS, GridFTP, etc.), o equipos de almacenamiento en cinta.
- Servidores de instalación y repositorios de *software*.
- Servidores de información, como por ejemplo gestores de bases de datos.
- Equipos de infraestructura de red, tales como servidores DNS o servidores DHCP.
- Otros muchos tipos de servidores, como equipos de sincronización horaria, gestores de claves y credenciales, etc.

Gestionar una red de ordenadores de esta complejidad es una tarea ardua y difícil. El mantenimiento de la red no sólo implica la gestión de los componentes individuales (actualizaciones, parches, etc.), sino también la gestión de las interdependencias entre los distintos componentes, lo que exige un orden preciso de las tareas a realizar. Por ejemplo, parar un servidor NFS para mantenimiento implica una modificación previa de la configuración de todos los clientes para que utilicen un servidor alternativo. Este caso tan trivial implica la necesidad de modelar las dependencias entre los distintos elementos de la red, en concreto, la dependencia entre los clientes y el servidor NFS.

También es importante que la ejecución de las tareas de administración tenga en cuenta la integridad de los trabajos. Por ejemplo, una operación de administración no debería abortar un trabajo o su entorno de ejecución a menos que haya buenas razones para ello (como por ejemplo una emergencia debido a un incidente de seguridad).

Distribuir la configuración entre las máquinas que componen una red también es un trabajo laborioso y delicado ya que puede ser diferente de una máquina a otra en un gran número de máquinas, y un mínimo error en una de ellas puede suponer una gran degradación o incluso la caída de todo el sistema. Por ello se hace necesario un sistema de gestión de la configuración que permita llevar un control preciso del estado de cada máquina.

Otra función compleja de administración es la monitorización de los equipos de la red, que debe ofrecer la posibilidad de escoger las métricas necesarias para poder obtener conocimiento del estado de la red y a

su vez agrupar la información de varios ordenadores para tener una visión global del conjunto.

Una vez establecido un sistema fiable de monitorización también es deseable un sistema de alarmas que permitan la localización rápida de problemas, y más aún un sistema de tolerancia a fallos que permita ejecutar acciones que solucionen los problemas localizados de forma automática.

Normalmente, estas complejas operaciones de administración se realizan de manera manual, aunque se trata de procedimientos difíciles, caros, y muy propensos a errores, sobre todo en redes de gran tamaño. TOMAS³ viene a subsanar esta carencia.

Herramientas de Administración de Sistemas

Existen numerosas herramientas de ayuda para la instalación, configuración y mantenimiento de una red de ordenadores bajo Linux. En [15] podemos encontrar una comparativa de las principales herramientas existentes, y en [16] un análisis más detallado de aquellas herramientas que han sido específicamente diseñadas para la instalación y gestión de *clusters* bajo Linux. Algunas de ellas son muy potentes pero no ofrecen soluciones completas sino parciales para las múltiples tareas de administración descritas. Otras son suficientemente generales, pero son productos propietarios y cerrados que no permiten auditorías, ampliación ni mejoras de su código. Por todo ello es deseable una solución de código abierto.

Entre las herramientas de *software* libre que proporcionan soluciones integradas de administración se encuentra Quattor [10]. Quattor proporciona un sistema de configuración de las máquinas centralizado, un

repositorio de *software* común y un sistema de despliegue automático, cumpliendo varios de los requisitos especificados para TOMAS³. Por ello el desarrollo de TOMAS³ partirá de los elementos proporcionados por Quattor adaptandolos a las nuevas necesidades y añadiéndole todo el resto de funcionalidades especificadas usando otras soluciones.

7.2. Descripción General del Sistema TOMAS³

TOMAS³ es tanto una aplicación *middleware* para la administración de medianos y grandes sistemas informáticos como la definición del conjunto de requisitos que ese sistema de administración centralizada debe cumplir. Además proporciona un *framework* para integrar las soluciones que ofrezcan las funcionalidades que cumplan dichos requisitos.

Por tanto TOMAS³ no es una aplicación monolítica sino una infraestructura donde se integran diversas soluciones. Además de dicho *framework* el proyecto TOMAS³ proporcionará un entorno completo que integrará un conjunto de soluciones seleccionadas y será perfectamente funcional.

La arquitectura global de TOMAS³ ofrece las siguientes funcionalidades:

- **configuración:** gestión y almacenamiento centralizado de la información de configuración de los equipos de la red, incluye información sobre el *hardware*, configuración del sistema y configuración de las aplicaciones,
- **instalación:** instalación inicial de los equipos, distribución e instalación de

- paquetes de *software*, y configuración y mantenimiento de los equipos de acuerdo a la información proporcionada por el subsistema de configuración,
- **monitorización:** recopilación, almacenamiento y consulta de la información sobre el estado actual de los equipos,
 - **tolerancia a fallos:** correlación de la información proporcionada por el subsistema de monitorización con una información patrón, y ejecución de acciones correctoras en caso de ser necesario,
 - **gestión de recursos:** gestión de la distribución de la carga entre los elementos de computación de la red, proporcionando una capa de abstracción sobre los gestores locales.
 - **administración de servicios:** Posibilidad de desplegar, arrancar, reiniciar y parar servicios de forma controlada y remota en una o múltiples máquinas simultáneamente.
 - **administración de software:** Control del *software* desplegado en una o varias máquinas y la posibilidad de instalar o desinstalar *software* de forma remota llevando un inventario preciso del *software* contenido en cada máquina.
 - **administración de hardware:** Inventario preciso y actualizado de los componentes *hardware* de una red o un *cluster*.
 - **administración de usuarios:** Gestión de usuarios del sistema tanto para los sistemas administrados como para los propios recursos de administración de TOMAS³

Consideraciones Generales de Diseño

Las características más importantes del diseño de las herramientas de instalación y configuración de TOMAS³ son:

- Se trata de un sistema no intrusivo, es decir, no se sustituye ninguno de los elementos de instalación y configuración de Linux, sino que éstos son ampliados.
- Está diseñado para ser escalable, pudiendo administrar desde redes de pocos nodos hasta redes de decenas de miles. La optimización y distribución en el uso de los recursos es un pilar fundamental en el diseño de TOMAS³.
- Es un sistema altamente modular, con unos interfaces entre módulos bien definidos, lo que facilita la sustitución o actualización de módulos individuales, y el desarrollo de nuevas versiones del sistema para distintas distribuciones Linux, e incluso para otras versiones de Unix.
- Se reutilizan tantos componentes externos como ha sido posible, lo que facilita el mantenimiento del *software*.

- Estándares de facto de la industria son integrados y reutilizados en la arquitectura, por ejemplo, protocolos estándares como HTTP, servicios web SOA y formatos como XML, son utilizados siempre que es posible.
- Se ha optado por un diseño distribuido donde la autonomía de los equipos de la red se mantiene tanto como sea posible, existiendo instancias locales en los equipos cliente de casi todos los subsistemas, realizando localmente todas las operaciones siempre que sea posible, asegurándose de esta manera la escalabilidad del sistema.
- Se proporciona un conjunto de librerías de base para acceder a cada subsistema vía funciones de control, y una librería de métodos comunes a todos los subsistemas.

Estado actual del proyecto

La primera versión utilizable de TOMAS³ está planificada para el 1 de junio de 2006. Actualmente las funcionalidades más desarrolladas son el sistema gestor de configuraciones y el de instalaciones, que comprende la instalación manual o automática de máquinas enteras y la instalación y actualización de paquetes de *software*. Dado que la incorporación de estas funcionalidades está más avanzada éstas se explicarán en más detalle, mientras que el resto, debido a que podrían sufrir ligeras modificaciones en sus requisitos, se explican de forma más breve.

7.3. Sistema gestor de configuraciones

El *Sistema Gestor de Configuraciones* proporciona un entorno para la gestión de la información de configuración de los equipos de una red. Por información de configuración entendemos cualquiera de los elementos de información necesarios para configurar de manera estática un equipo. No se incluye aquella información que es dinámica o que cambia, por ejemplo el contenido de una base de datos residente en el equipo, ni aquella información que es generada por la propia máquina, como por ejemplo la carga del sistema.

El Sistema Gestor de Configuraciones se compone de los siguientes elementos (véase la Figura 7.1):

- un servidor central donde se almacenan todos los elementos de configuración de los equipos de la red, junto con un lenguaje de alto nivel específicamente diseñado para describir estos elementos de configuración,
- un gestor local de configuraciones residente en los equipos cliente, junto a una librería que permite a las aplicaciones acceder a su información de configuración, y
- un conjunto de módulos de acceso, que nos facilitan la consulta de la información almacenada en el servidor central.

A continuación se describe cada uno de estos elementos.

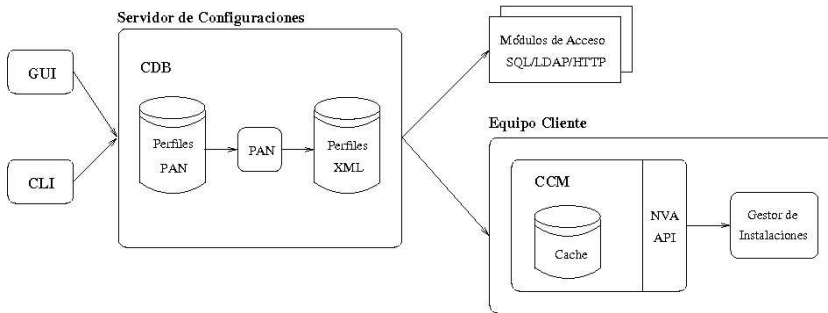


Figura 7.1: Sistema Gestor de Configuraciones

7.3.1. Servidor CDB

La información sobre la configuración de los equipos de la red se almacena de manera centralizada en una base de datos llamada CDB (del inglés *Configuration DataBase*). Los elementos de configuración de los clientes están organizados en esta base de datos en forma de *perfiles*. Por cada equipo cliente a instalar debe existir en el servidor CDB el correspondiente perfil que nos describa la configuración final deseada para el equipo. Los perfiles de configuración son escritos por el administrador de sistemas utilizando para ello un nuevo lenguaje llamado *Pan*, y a continuación son compilados a ficheros XML.

Pan (véanse [17] y [18]) es un lenguaje de alto nivel específicamente diseñado para escribir perfiles de configuración. Pan nos proporciona una vista de la información de configuración optimizada para realizar tareas de alto nivel, tales como la gestión de un gran número de equipos cliente. Por ejemplo, con Pan podemos agrupar los elementos de configuración comunes de los equipos de un mismo entorno de traba-

jo, describir la configuración de un determinado tipo de servicio, o caracterizar los elementos *hardware* de los clientes.

Los perfiles de configuración en formato XML están optimizados para realizar operaciones de instalación y configuración de los equipos cliente, por lo que contienen sólo aquella información que es relevante para estas tareas. Los perfiles XML, generados a partir de los correspondientes perfiles Pan, son descargados por el subsistema de configuración de los clientes (descrito en la sección 7.3.2) para su propio uso. La información contenida en los perfiles XML es transformada por los clientes en información de configuración que pueda ser entendida por el sistema operativo y las aplicaciones, como por ejemplo, en un fichero de configuración de sendmail `/etc/sendmail.cf`, o en el fichero `/etc/xinetd.conf`.

CDB dispone de dos interfaces públicos para gestionar la información de configuración:

- una interfaz de lectura/escritura para mantener la información de configuración expresada en lenguaje Pan,

que es accesible a través de una herramienta en línea de órdenes, o de una interfaz gráfica, y

- una interfaz de sólo lectura para que los clientes recuperen su información de configuración en forma de perfiles XML.

El servidor CDB no es sólo un lugar de almacenamiento y gestión de perfiles de configuración, sino que además, también nos proporciona mecanismos para validar la información introducida. CDB trabaja en modo transaccional: una vez que la validación y la compilación han sido realizadas con éxito, los cambios introducidos por el usuario son almacenados en la base de datos, y publicados para que sean accesibles a los clientes.

7.3.2. Clientes CCM

El *Gestor de Configuración de Clientes* o CCM (del inglés *Configuration Client Manager*) reside en cada uno de los clientes a configurar, y es el responsable del almacenamiento y la gestión de la información de configuración del cliente.

Los clientes acceden a su información de configuración descargándose desde el servidor CDB su perfil en formato XML. La comunicación entre el servidor y el cliente se realiza mediante un protocolo de distribución de perfiles basado en HTTP. Una vez descargado el perfil y almacenado localmente, CCM ejecuta el Sistema Gestor de Instalaciones (descrito en la sección 7.4), que es el encargado de hacer efectiva la nueva configuración. El servidor CDB envía una notificación de actualización (basada en UDP) a los equipos cliente cada vez

que el administrador de sistemas modifica su perfil de configuración. Cuando se recibe una notificación de actualización, CCM procede a la descarga del perfil actualizado.

El Sistema Gestor de Instalaciones puede acceder a los diferentes elementos de configuración del equipo cliente utilizando la librería NVA-API. Esta librería proporciona un acceso fácil y transparente a la información contenida en los perfiles.

7.4. Sistema Gestor de Instalaciones

El Sistema Gestor de Instalaciones o IMS (del inglés *Installation Management Subsystem*) proporciona las herramientas necesarias para instalar y configurar el sistema operativo y las aplicaciones en los equipos de la red. IMS nos permite gestionar la instalación y la actualización del sistema operativo y del *software* de aplicaciones, configurar correctamente los parámetros del sistema, y aplicar todas aquellas políticas y restricciones que hayamos definido para nuestra red.

En la figura 7.2 podemos ver un esquema simplificado de la arquitectura interna de IMS. Básicamente, el sistema se compone de los siguientes elementos:

- un gestor de la configuración de los equipos cliente, que se encarga de hacer efectivos los perfiles de configuración definidos en CDB,
- un repositorio de paquetes de *software*, que contiene todos los paquetes de *software* necesarios para instalar el sistema operativo y las aplicaciones,

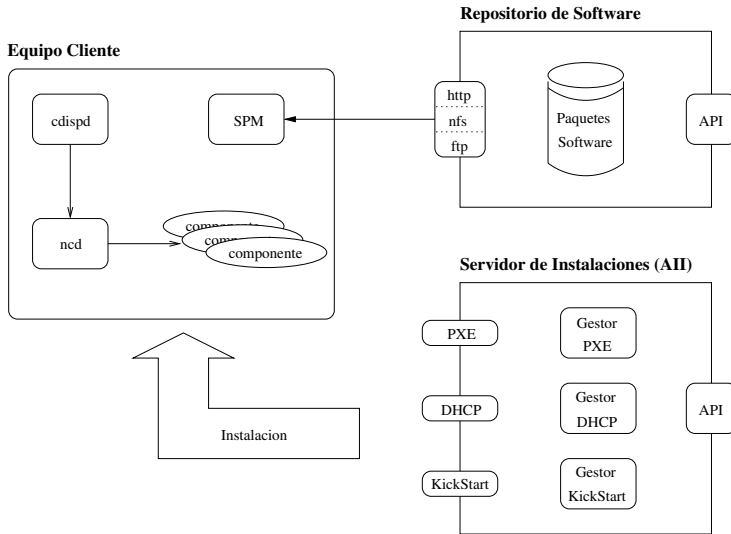


Figura 7.2: Sistema Gestor de Instalaciones

- un gestor y distribuidor de paquetes *de software*, encargado de descargar del repositorio *software* los paquetes a instalar, y de instalarlos en el equipo, y
- un conjunto de herramientas para la instalación inicial y desatendida de los equipos.

Nótese que la configuración de los elementos que forman IMS también es gestionada por el Sistema Gestor de Configuraciones. A continuación se describen los elementos que forman IMS.

7.4.1. Configuración de los Equipos

El Sistema Gestor de la Configuración de los Equipos o NCM (del inglés *Node Confi-*

guration Manager) es el responsable de hacer efectiva en los equipos la información de configuración proporcionada por CDB. NCM proporciona un entorno de trabajo para adaptar la configuración actual de un equipo a su configuración deseada, tal y como está descrita en el perfil del equipo residente en CDB.

NCM se basa en un conjunto de módulos *software* responsables de configurar localmente los servicios y las aplicaciones. Estos módulos, llamados *componentes*, son los encargados de mantener la correcta configuración de los equipos. Para ello acceden a la información contenida en los perfiles de configuración, utilizando la librería NVA-API, y crean, borran o actualizan los ficheros de configuración de los servicios y las aplicaciones locales. Cada componente contiene el conocimiento neces-

rio para trasladar la información contenida en los perfiles, a la sintaxis específica de cada fichero de configuración. En caso de que la configuración del equipo se ajuste a la descrita por su perfil, los componentes no modifican ningún fichero. Los componentes también son responsables de notificar a los servicios la existencia de cambios en su configuración (por ejemplo, ejecutando la orden `restart` de un guión de inicio de System V).

Nótese que los componentes no se encargan de realizar tareas de administración repetitivas, tales como limpiar el directorio `/tmp`, que han de ser realizadas mediante otras utilidades (por ejemplo con la herramienta `tmpwatch`).

Los componentes pueden ser ejecutados de múltiples maneras:

- por un demonio local, cada vez que cambie alguno de los elementos de configuración que les afectan,
- bajo demanda, por ejemplo manualmente por el administrador del sistema, o mediante algún mecanismo de ejecución remota, o
- a intervalos regulares, por ejemplo utilizando la herramienta `cron`.

Ejemplos de componentes son: un componente que se encargue de gestionar la configuración del programa `sendmail`, un componente para gestionar todos los parámetros de red (TCP/IP, DNS, etc.), o un componente para configurar el propio sistema NCM.

Además de los mencionados componentes, el sistema NCM incluye los siguientes elementos:

- `ncd`: (del inglés *Node Configuration Deployer*) es el entorno de trabajo bajo el cual se ejecutan los componentes,
- `cdispd`: (del inglés *Configuration Dispatch Daemon*) está encargado de monitorizar los perfiles de configuración del equipo, y en caso de modificación del perfil, ejecutar (vía `ncd`) aquellos componentes que se vean afectados,
- librerías de soporte para componentes, proporcionan ayuda para las tareas de gestión rutinarias (información del sistema, interfaces a servicios del sistema, edición de ficheros), manejo de ficheros de registro, etc.

7.4.2. Distribución e Instalación de Paquetes de Software

El Sistema Gestor de Paquetes de Software o SPM (del inglés *Software Package Management*) es el responsable del almacenamiento y la gestión de los paquetes de software, así como de la distribución e instalación de los mismos en los equipos cliente.

El sistema SPM se compone de los siguientes elementos:

- un repositorio de software: donde se almacenan todos los paquetes de software a instalar en los clientes, tanto los paquetes correspondientes al sistema operativo, como aquellos que correspondan al software de aplicación,

- un gestor avanzado de paquetes (*rpm*): una aplicación construida sobre la utilidad *rpm*, y que le añade la capacidad de realizar múltiples operaciones en múltiples paquetes de *software* en una única transacción,
- herramienta SPMA: (del inglés *SPM Agent*) encargada de calcular la lista de operaciones de instalación, desinstalación o actualización de paquetes necesarias para actualizar el equipo,
- componente SPM: integrado dentro del sistema NCM, se encarga de configurar y ejecutar la herramienta SPMA con la lista de paquetes a instalar en el equipo en base al contenido de su perfil,
- procedimientos CDB: un conjunto de procedimientos del lenguaje Pan pensados para facilitar la definición y validación de listas de paquetes.

Todos los paquetes de *software* a instalar en los equipos cliente son almacenados de manera centralizada en un repositorio de *software*. Este repositorio cuenta con una interfaz de usuario para facilitar su gestión, permitiendo a los administradores del sistema añadir, borrar, y consultar los paquetes disponibles. Los clientes pueden acceder al contenido del repositorio utilizando los protocolos HTTP o FTP, o también mediante algún sistema de ficheros compartidos, como por ejemplo NFS. Los clientes pueden incluso almacenar localmente los paquetes a instalar, asegurándonos de esta manera la escalabilidad del sistema.

La lista de paquetes que han de ser instalados en cada equipo viene dada por su perfil de configuración. El componente SPM es el encargado de extraer esta lista de paquetes, con la ayuda de la librería NVA-API, y de crear un fichero de configuración local para la utilidad SPMA. A continuación, el componente SPM ejecuta la aplicación SPMA, que lee el fichero de configuración con los paquetes objetivo a instalar en el equipo, los compara con la lista de paquetes actualmente instalados, y calcula las operaciones de instalación, desinstalación o actualización necesarias para actualizar el equipo. Finalmente, SPMA ejecuta la herramienta avanzada de instalación de paquetes *rpmt*, que se encarga de llevar a cabo la transacción necesaria para hacer efectiva la nueva lista de paquetes.

La herramienta SPMA puede ser utilizada para instalar la totalidad de los paquetes de los equipos clientes, o sólo un subgrupo de ellos, dependiendo si se trata de equipos sobre los que tenemos un control total, o si lo que se quiere es garantizar que los equipos disponen de un grupo de aplicaciones determinado.

7.4.3. Instalación Automática de Equipos

El Subsistema de Instalación Automática de Equipos o AII (del inglés *Automated Installation Infrastructure*) proporciona un conjunto de herramientas para la instalación inicial del sistema operativo en los equipos cliente. Esta primera instalación se realiza de manera automática y desatendida, a través de la propia red, y mediante el uso de servicios y herramientas estándares como son PXE, DHCP o TFTP.

Para poder llevar a cabo la instalación

inicial de un equipo de manera remota, hay que realizar previamente los siguientes pasos:

- configurar los servicios de red necesarios, como añadir las correspondientes entradas en el servidor DNS y en el servidor DHCP para cada uno de los equipos a instalar,
- configurar el programa cargador, los equipos descargan desde la red un programa cargador o NBP (*Network Bootstrap Program*), que basado en un fichero de configuración, decidirá la manera de arrancar el equipo (desde disco local, o iniciar la instalación a través de la red), y
- configurar el programa de instalación del sistema operativo: si el programa NBP decide que hay que iniciar la instalación del sistema operativo, el programa instalador necesita de un fichero que contenga todos los datos básicos de configuración del sistema, tales como el esquema de particiones, el *software* inicial a instalar, los detalles del *hardware*, etc.

AII proporciona las herramientas necesarias para simplificar las tareas descritas. Estas herramientas están totalmente integradas con el sistema CDB, desde donde recuperan todos los datos de configuración que son necesarios, aunque también podrían ser utilizadas de manera aislada.

El sistema AII se compone de los siguientes módulos:

- `aii-dhcp`: que gestiona la parte de configuración del servidor DHCP para cada nodo (añadir, modificar o borrar una entrada correspondiente a un nodo),
- `aii-nbp`: encargado de gestionar la configuración de NBP para cada equipo (arrancar desde disco local, o iniciar un proceso de re-instalación remota desde red), y
- `aii-osinstall`: que gestiona los ficheros de configuración del instalador del sistema operativo.

Mientras que para redes del orden de varios centenares de equipos, los servicios involucrados (DHCP, TFTP, instalador del sistema operativo) pueden ejecutarse en una única máquina sin que ésta se sobrecargue, en redes de mayor tamaño es esencial que los equipos a instalar puedan ser particionados, debido a problemas de escalabilidad. Por esta razón, es posible instalar los tres módulos en máquinas diferentes. E incluso, para proporcionar redundancia al sistema, es también posible instalar el mismo módulo en más de una máquina (por ejemplo, tener dos servidores de DHCP o tres de instalación del SO). La elección de qué servidores son utilizados por los clientes se realiza mediante los perfiles de configuración almacenados en CDB.

7.4.4. Administración de servicios

El subsistema de administración de servicios proporciona un conjunto de herramientas para la administración de los servicios ofrecidos por los nodos de la red. Desde la consola de administración se podrá

monitorizar el estado de un servicio, arrancarlo, pararlo, etc. El subsistema estará dotado de una cierta inteligencia para resolver contingencias sencillas, como por ejemplo las dependencias entre servicios, de forma que será capaz de arrancarlos o pararlos en un orden concreto si fuera necesario.

7.4.5. Administración de *hardware*

El sistema de administración de *hardware* permite llevar un inventario actualizado del *hardware* que constituye el sistema, tanto a nivel de nodos como al de *hardware* incluido en esos nodos, así como de todos los elementos configurables susceptibles de ser incluidos en el ámbito de administración de TOMAS³, que será el de aquellos ordenadores donde corra el cliente de TOMAS³ y los dispositivos que ofrezcan una interfaz de administración WSDM.

El sistema de administración consiste en un repositorio de información sobre *hardware* y los servicios de almacenamiento y recuperación de información que actúan sobre ese repositorio. Además ofrecerá servicios de información sobre *hardware* de forma que será posible crear perfiles con unos determinados requisitos de *hardware* en función de su uso, y así consultar al sistema de administración de *hardware* si ciertas máquinas cumplen con esos perfiles.

7.4.6. Monitorización y alarmas

El sistema de monitorización permite saber en tiempo real el estado del sistema de forma visual. Permite agrupar la información de forma que se puede tener una visión global de un conjunto de nodos o una particular y específica de uno solo de ellos.

Los grupos de nodos a monitorizar pueden ser configurados por el usuario según la arquitectura que desee, así como las métricas que desee obtener. También se guardará un histórico de los datos para poder realizar estudios estadísticos (*datawarehouse*).

Es posible determinar límites para las métricas monitorizadas de máquinas o grupos concretos y activar alarmas en base a éstos. Estas alarmas se visualizarán en la consola de administración y se notificará al administrador de diversas formas configurables.

7.4.7. Tolerancia a fallos

Utilizando el subsistema de monitorización y alarmas se podrán detectar problemas y se permitirá lanzar automáticamente y de forma desatendida procesos que solucionen los problemas detectados. Estas acciones podrán ser tan sencillas como rearrancar un servicio o tan complejas como balancear la carga de un servicio instalando desde cero una máquina, el *software* y la configuración necesaria y añadiéndolo al servicio. Estos procesos pueden ser determinados por el usuario.

7.4.8. Administración de usuarios

TOMAS³ proporcionará una interfaz para administrar de forma centralizada usuarios tanto para los sistemas administrados por TOMAS³ como para la administración de sus propias funcionalidades. La principal novedad aportada por el sistema de administración de usuarios de TOMAS³ es que se trata un punto único para múltiples *clusters*.

7.5. Conclusiones

La arquitectura de TOMAS³ definida por Ándago y el *middleware* resultante de implementarla permite incrementar significativamente la automatización de las tareas de instalación y mantenimiento de una red de ordenadores Linux, simplificando las tareas de administración y por tanto, reduciendo los costes. Además, el sistema TOMAS³ permite mantener el control absoluto desde el punto de vista global del sistema y desde el particular de los nodos y la supervisión de las tareas resulta simple y rápida en las manos de un administrador de sistemas experto.

Bibliografía

- [1] Universidad autónoma de Madrid, <http://150.244.56.228>
- [2] Ministerio de turismo, industria y comercio, <http://www2.mityc.es>
- [3] Ministerios de educación y ciencia, <http://www.mec.es>
- [4] Programa de Fomento de la investigación Técnica (PROFIT) <http://www2.mityc.es/Profit/Profit/Guia/Index.htm>
- [5] Universidad politécnica de Madrid, <http://internetng.dit.upm.es>
- [6] Distributed Management Task Force, <http://www.dtmf.org>
- [7] Common Information Model, <http://www.dtmf.org/standards/cim>
- [8] Web Services Distributed Management, http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsdm
- [9] OASIS, <http://www.oasis-open.org>
- [10] Quattor, <http://www.quattor.org>
- [11] *European Union DataGrid Project*, <http://www.eu-datagrid.org>.
- [12] Foster, Ian, *The Grid: Blueprint for a New Computing Infrastructure*, Morgan-Kaufmann, Julio 1998.
- [13] *CERN: European Organization for Nuclear Research*, <http://www.cern.ch>.
- [14] *EU DataGrid Fabric Management workpackage*, <http://hep-proj-grid-fabric.web.cern.ch/hep-proj-grid-fabric>.
- [15] Barroso, Maite, “WP4 Report on Current Technology”, *DataGrid Technical Report DataGrid-04-TED-0101-3_0*, Apr. 2001
- [16] García Leiva, Rafael y Peso, José del, “Open Source Solutions for Installation and Management of PC Clusters under Linux for ATLAS”, *ATLAS Notes ATL-SOFT-2003-001*, Oct. 2002.
- [17] Cons, Lionel and Poznański, Piotr, “Pan: A High-Level Configuration Language”, *LISA Conference Proceedings*, 2002.
- [18] Cons, Lionel and Poznański, Piotr, “Pan Language Specification”, *DataGrid Technical Report DataGrid-04-TED-0153*, Nov. 2002 (<http://hep-proj-grid-fabric-config.web.cern.ch/hep-proj-grid-fabric-config/pan-spec.pdf>)

Parte II

Proyectos fin de carrera

Intérprete de diagramas de flujo.

Desarrollo en software libre

Ismael Moreno Caballero

ismael.morenocaballero@alum.uca.es

Escuela Superior de Ingeniería de Cádiz. Universidad de Cádiz.

Carlos Rioja del Río

carlos.rioja@uca.es

Depto. de Lenguajes y Sistemas Informáticos. Universidad de Cádiz.

Resumen

Presentamos aquí la realización de un Proyecto Fin de Carrera desarrollado bajo la idea de ayudar a futuros estudiantes de las titulaciones de informática en la Escuela Superior de Ingeniería.

Para este *Intérprete de diagramas de flujo* (en adelante FCI) se han utilizado diversas herramientas de software libre, que posibilitan la comunicación y extensión a la comunidad de la aplicación. Además, por su naturaleza didáctica, pretende ayudar a los alumnos noveles en el aprendizaje de desarrollo de diagramas.

8.1. Introducción

Se requiere cierta destreza para aprender a crear algoritmos bien diseñados y, sobre todo, que arrojen resultados correctos, por lo que toda buena metodología de aprendizaje contemplará un período donde el alumno practicará la creación de estos, aumentando la complejidad progresivamente.

Un obstáculo es el hecho de que los algoritmos necesitan ser traducidos a programas que lo implementen, para poder ser comprobados en funcionamiento. Aprender el primer lenguaje de programación y acostumbrarse a su sintaxis y normas gramaticales llevará un determinado tiempo al

alumno, durante el cual solo podrá probar la corrección de sus algoritmos o bien usando métodos formales (en ocasiones, complejos) que requieren cierta destreza matemática, o bien tener siempre disponible un tutor (o, simplemente, alguien con más experiencia), que sea quien realice las tareas de comprobación (con todos los problemas que el factor humano pueda implicar).

FCI pretende ser un entorno de comprobación empírica de algoritmos, apta para programadores noveles, donde el tiempo de aprendizaje o adaptación a este sea mínimo.

FCI permite crear algoritmos de una manera muy intuitiva y ejecutarlos, para poder comprobar que, sea cual sea la entrada que reciban, arrojan los resultados deseados. En el caso de que no funcionen como se esperaba, o bien se quiera analizar detenidamente los pasos que realizan, pueden ser ejecutados paso a paso y, por ende, depurados, permitiendo además visualizar el valor de las variables en cada instante de la ejecución.

8.2. Interfaz Gráfica de Usuario

El modelado de la aplicación basado en el patrón de diseño MVC ha permitido separar la programación de la interfaz gráfica de la generación de diagramas y su procesado. Esta metodología está ampliamente aceptada en la industria actual y ha servido de base de desarrollo para las aplicaciones más usadas cotidianamente por el usuario medio.

Con esto, la interfaz gráfica (GUI) aporta al usuario los mecanismos necesarios para la manipulación de los diagramas (tanto su

creación y edición como su compilación y depurado).

No difiere demasiado de las interfaces de la gran mayoría de programas de diseño de diagramas (tales como Dia, Visio, Umbrello, etc), presentando menús con iconos gráficos de edición y, también, menús textuales desplegables.

8.3. Características de los diagramas.

Un diagrama de FCI está formado por un conjunto de nodos, cuya interconexión define el flujo de ejecución de estos que debe seguir el algoritmo.

Los nodos se clasifican según el tipo de cometido al que han sido destinados, y su dibujo en la vista del diagrama está adaptada al estándar ISO 5807.

Cada tipo de nodo es interpretado como una instrucción atómica de alto nivel y que puede equipararse a una línea de código de cualquier lenguaje basado en el paradigma de la programación estructurada. Así, se dispone de nodos para definir variables, para realizar asignaciones a variables, para definir matrices, para evaluar el resultado booleano de una expresión, para llamar a procedimientos y para delimitar bloques de código.

8.4. Características del intérprete.

Se enumeran las características principales del intérprete que permite ejecutar los diagramas y, de esta manera, comprobar las diferentes soluciones que devuelvan estos:

1. Ejecución sobre una máquina virtual interna (los diagramas son traducidos a código de esta).

2. Dos tipos genéricos de datos: Numérico (números reales) y Strings (cadenas de texto).
3. Operadores aritméticos para expresiones (incluyendo uno de potencia y otro de módulo), de comparación y lógicos.
4. Posibilidad de llamada a bloques de nodos independientes al flujo principal de ejecución.
5. Funciones internas para la realización de tareas muy frecuentes y que son difíciles de implementar con los diagramas, con ejecución optimizada con respecto a las llamadas a bloques.
6. Evaluación de expresiones lógicas por cortocircuito.

No es necesaria la especificación explícita de variables locales (se definen automáticamente en una asignación).

Matrices multidimensionales con tamaños que pueden ser definidos a partir de expresiones aritméticas complejas.

Reconocimiento de identificadores case-sensitive (sensible a las mayúsculas).

8.5. Desarrollo del proyecto.

FCI es software libre (licenciado con GPLv2) desarrollado, íntegramente, usando herramientas libres.

Está programado en

y compilado con el GNU Compiler Collection (GCC). La interfaz gráfica se ha

creado con la biblioteca wxWidgets, licenciada con una pequeñísima variante de la LGPL.

En lo que respecta al apartado de traducción de diagramas a código de máquina virtual, ha sido necesario el uso de Flex y Bison para poder traducir expresiones complejas.

La creación del software y las tareas de edición de documentos asociados ha sido bajo un entorno KDE y herramientas para este, como el entorno de desarrollo KDevelop, el editor de textos Kate y el IDE para \LaTeX Kile.

En la actual versión, FCI se ejecuta bajo sistemas operativos GNU/Linux que puedan ejecutar un servidor de ventanas X-window y que tengan instalados la biblioteca GTK+. No obstante, al haberse usado bibliotecas y herramientas transportables, puede prepararse este software para ser ejecutado bajo sistemas Windows 9x/NT/2000/XP realizando, presumiblemente, una mínima cantidad de cambios en el código.

8.6. Muestras.

A continuación se presenta un par de capturas de pantalla del software en funcionamiento:

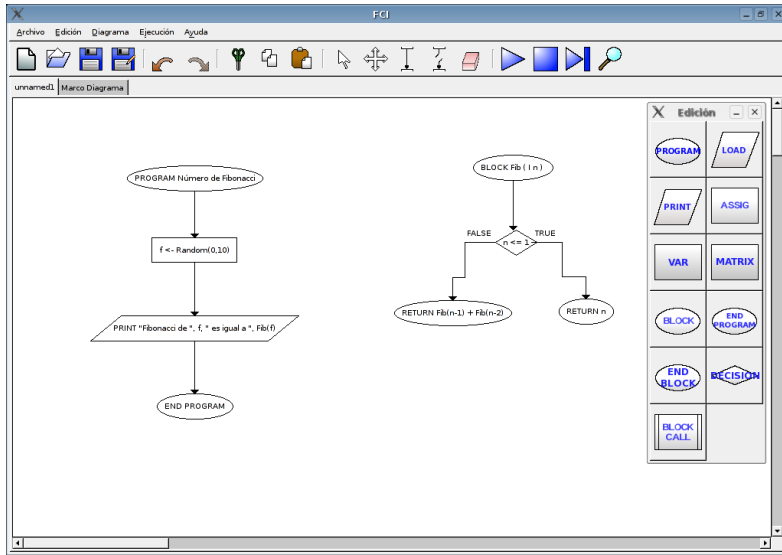


Figura 8.1: captura de pantalla del intérprete

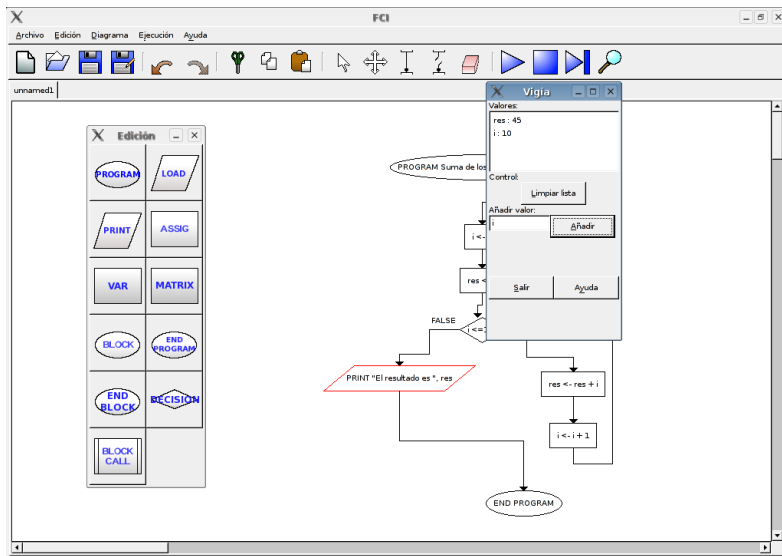


Figura 8.2: captura con watch(vigía) de variables

Bibliografía

- [1] wxWidgets
<http://www.wxwidgets.org>
- [2] Bison <http://www.bison.org>
- [3] Flex <http://www.gnu.org/software/flex/>

Plataforma libre de desarrollo de algoritmos para el proyecto Pelican

y su interfaz basada en Web Services¹

Adrián Santos

alu2527@etsii.uull.es

Escuela Técnica Superior de Ingeniería Informática. Universidad de La Laguna.

Francisco Almeida

falmeida@ull.es

Depto. de Estadística, Investigación Operativa y Computación. Universidad de La Laguna.

Vicente Blanco

vblanco@ull.es

Depto. de Estadística, Investigación Operativa y Computación. Universidad de La Laguna.

Resumen

Presentamos en este artículo el proyecto de fin de carrera que estamos realizando para implementar una plataforma libre de desarrollo y acceso a algoritmos del pro-

yecto Pelican. Describiremos brevemente el diseño de la plataforma existente y justificaremos los objetivos del proyecto.

¹Este trabajo ha sido parcialmente soportado por la EC (FEDER) y por el Ministerio de Educación y Ciencia, (TIN2005-09037-C02-01).

9.1. Introducción

El presente proyecto se basa en el trabajo previo realizado en Pelican (Pelican [1]). El objetivo del proyecto Pelican es desarrollar una versión paralela integrada de la GSL (*GNU Scientific Library*, GSL [2]) que pueda ser utilizada como un entorno para la resolución de problemas relacionados con la computación numérica científica. En particular, se pretende que dicha biblioteca sea transportable a varias arquitecturas paralelas, incluyendo procesadores con memoria compartida y distribuida, sistemas híbridos (consistentes en una combinación de ambos tipos de arquitecturas) y *clusters* de nodos heterogéneos.

La Biblioteca Científica GNU está formada por cientos de rutinas para la realización de cálculo numérico que abarcan aritmética de números complejos, matrices y vectores, álgebra lineal, estadística y optimización, etc.

Nuestro proyecto consiste en ampliar Pelican dotándolo de una herramienta que permita acceder, implementar y usar los algoritmos ya disponibles y aquellos aportados por la comunidad. Todo el código implementado se pondrá a disposición del público bajo una licencia libre.

9.2. Arquitectura software

El diseño de la biblioteca Pelican se basa en una arquitectura software multinivel (ver Figura 9.1): cada capa ofrece ciertos servicios a las capas superiores a la vez que oculta cómo son implementados.

La capa en la que nos centraremos en el proyecto es la denominada *Web Service*. Consiste en un servicio web que permite ac-

ceder a las bibliotecas disponibles en Pelican (ver Figura 9.2).

Los principales problemas a resolver durante el análisis y desarrollo del módulo son los siguientes:

Instalación: Dado la heterogeneidad de los sistemas en los que se planea instalar el servicio se debería dotar de un sistema de instalación independiente de la plataforma.

Autenticación: Dado que el servicio permite acceder al sistema de cómputo donde sea instalado se ha de dotar de medidas de seguridad que controlen el acceso al mismo.

Gestión de algoritmos: El problema más complejo viene dado por la necesidad de dotar al usuario de un método *automático* para añadir nuevos algoritmos. El servicio web necesita una descripción XML del algoritmo para conocer sus argumentos de entrada y salida. Se ha de desarrollar un sistema capaz de extraer estos datos del código fuente, generar la descripción y añadir las nuevas rutinas al servicio.

Ejecución: Al igual que en la instalación, se ha de proveer interfaces para el acceso a varios sistemas de colas. Por ello es necesario desarrollar una librería de ejecución con unos métodos comunes (parte independiente de la plataforma) y varios *backends* dependientes de la máquina donde se vaya a ejecutar.

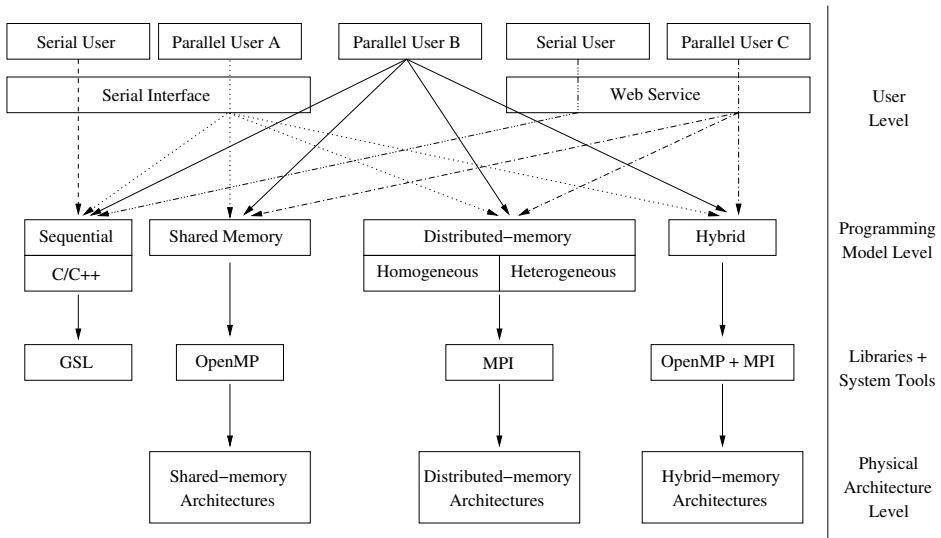


Figura 9.1: Arquitectura software de la biblioteca paralela integrada para el cálculo numérico científico.

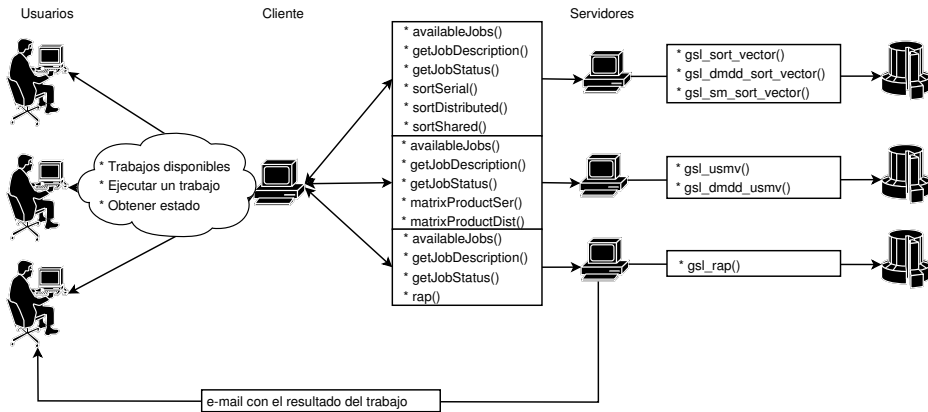


Figura 9.2: Esquema genérico del funcionamiento del servicio web

9.3. Proyecto

Actualmente existe una versión preliminar del proyecto desarrollada como prueba

de concepto. Se han propuesto soluciones para varios de los problemas planteados pero no se ha obtenido una solución factible al problema completo. En nuestro proyec-

to nos centraremos en aquellos puntos que han quedado pendientes en el trabajo actual. En definitiva, los objetivos marcados son:

- Implementación de una forja en la que alojar el proyecto.
- Generalizar las descripciones XML de los algoritmos para que permitan estructuras de datos complejas en sus argumentos.
- Ampliar el número de algoritmos disponibles.
- Automatizar la instalación del servicio en las máquinas de cómputo.
- Mejorar la interfaz web de acceso a las rutinas para permitir la administración de usuarios y la adición de servidores de cómputo.

Bibliografía

- [1] J. Aliaga, F. Almeida, J.M. Badia, S. Barrachina, V. Blanco, M. Castillo, U. Dorta, R. Mayo, G. Quintana, C. Rodríguez, y F. Sande, “Parallelization of GSL: Architecture, interfaces, and programming models,” in *Proc. of the 11th European PVM/MPI Users’ Group Meeting in conjunction with DAPSYS’04 (EuroPVM/MPI 2004)*, Budapest, Hungary, September 19–22 2004, vol. 3241 of *Lecture Notes in Computer Science*, pp. 199–206.
- [2] M. Galassi, J. Davies et al., *GNU scientific library reference manual*, July 2002, for GSL Version 1.2.

Parte III

Talleres

Edición de audio digital con Audacity

Ignacio Palomo Duarte

bohemegm@gmail.com

Escuela Superior de Ingeniería de Cádiz. Universidad de Cádiz.

Resumen

Este artículo pretende mostrar las diversas posibilidades que ofrece el programa de edición de audio digital Audacity. Este programa facilita al usuario tareas comunes como pueden ser la restauración de vinilos (eliminación de ruido, ecualización), utilización de Audacity como multipistas digital o aplicación de efectos de audio. Audacity es un programa libre y de código abierto para grabar y editar sonidos, disponible para Mac OS X, Microsoft Windows, GNU/LINUX y otros sistemas operativos.

10.1. Introducción

Hoy en día el audio digital forma parte importante de nuestras vidas; ya sea en forma de música comprimida en MP3, *pod-*

casts, inserción de audio en vídeo digital o aplicaciones multimedia, el usuario de informática necesita llevar a cabo ese conjunto de tareas, pero sin el desembolso que puede acarrear la compra de un programa de edición de audio profesional.

A los usuarios informáticos se les exige un conocimiento general de todas las posibilidades que ofrece la informática moderna. Muchos usuarios poseen conocimientos de edición de texto, diseño gráfico, hojas de cálculo, presentaciones multimedia, pero muy pocos saben cómo editar audio digital, muy presente en nuestra sociedad gracias a la música en formato MP3, creación de contenidos multimedia o edición de vídeo digital, por ejemplo.

Este artículo pretende informar sobre el uso y situaciones en las que debemos usar un editor de audio, tanto en aplicaciones

ofimáticas como a la hora de la edición y *masterización* profesional de un tema musical.

10.2. Edición de audio digital

Para ilustrar el funcionamiento y la utilidad de un editor de audio expondremos un par de ejemplos, uno de ellos aplicado al campo del audio profesional y otro caso solventando el problema de la restauración de audio procedente de grabaciones de baja calidad, correspondiente a las necesidades de un usuario de informática común.

10.2.1. Primer caso: Composición y mezcla de un tema comercial

Pensemos en un caso real en el que se requiera el uso de un editor de audio. Imaginemos, por ejemplo un estudio musical profesional encargado de la elaboración de un tema comercial. El primer paso sería utilizar un secuenciador para componer la pieza musical, utilizando diferentes elementos hardware o sintetizadores virtuales. Luego tendría lugar la grabación de las diferentes voces en un estudio de grabación. En este punto, usando un editor de audio se corregirían los pequeños defectos en las voces, tales como sibilancia, compresión dinámica, volumen, normalización o ajustes en frecuencias.

Una vez tenemos tanto las voces como la música unimos y mezclamos ambas partes en un multipistas digital (función que también cumpliría Audacity, como se muestra en la figura 10.1), ajustamos los volúmenes de cada canal y aplicamos pequeños retoques a los instrumentos por separado.

Por último, una vez hemos obtenido la pista de audio final, es el momento del *mastering*. Usando un editor de audio analizamos el espectro de frecuencias del tema y aplicamos diferentes efectos, ecualizaciones y técnicas para obtener un sonido final más compacto y mucho más equilibrado en volumen y frecuencias.

10.2.2. Segundo caso: necesidades de un usuario no profesional

Sea el caso de estudio un usuario de informática común. Supongamos que posee un conjunto de vinilos, música en formato cassette o grabaciones de baja calidad, y le gustaría digitalizar esa música para evitar un mayor deterioro de la misma con el paso del tiempo y también para poder disponer de esa música en su reproductor de MP3.

Utilizando las capacidades de grabación de un editor de audio digitalizaría la señal de su equipo de música, grabando, por ejemplo, toda una cara del vinilo. Luego aplicaría una reducción de ruido para limpiar la señal y normalizaría el volumen. Para finalizar dividiría la toma de audio en diferentes archivos, cada uno correspondiendo a una canción del vinilo.

Por ejemplo, en la figura 10.2 vemos como la forma de onda en la imagen de la derecha goza de un mayor rango dinámico y un volumen más constante que la onda de la izquierda, obteniéndose un sonido más rico, cálido y agradable de escuchar.

10.3. Programa principal

Audacity es un editor de sonidos libre para Windows, Mac OS X, GNU/Linux, y

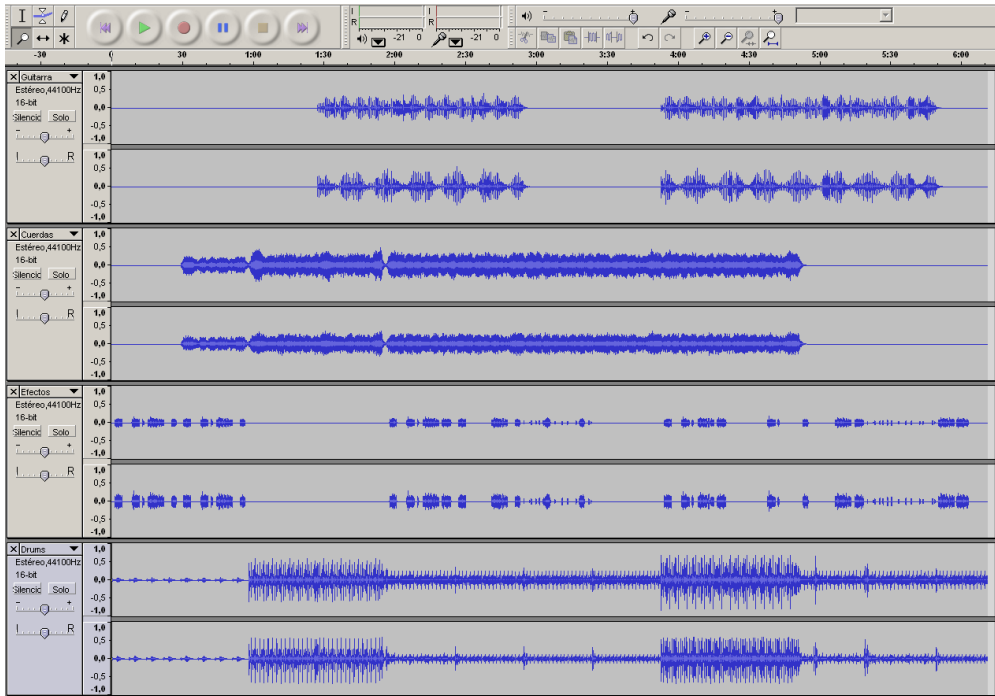


Figura 10.1: Audacity trabajando en modo multipistas digital.

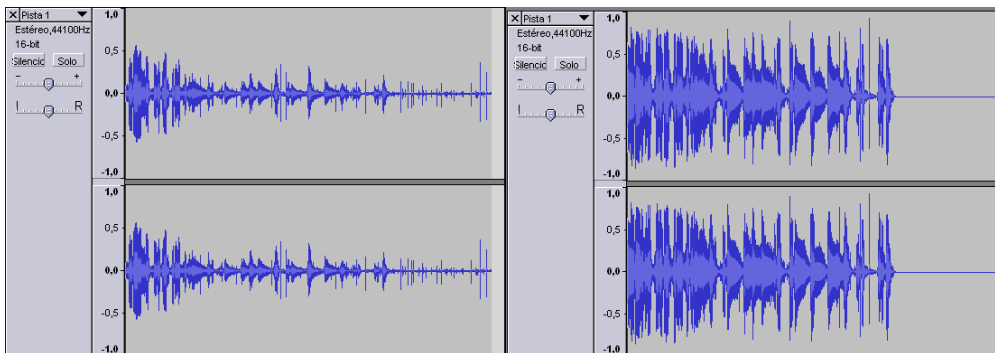


Figura 10.2: Audio antes y después del tratamiento con Audacity.

otros sistemas operativos. Funciona en modo gráfico y es muy fácil de usar. Este programa presenta, como funciones principales:

- Grabar sonidos en vivo.
- Convertir cintas y grabaciones a sonido digital o CD.
- Editar Ogg Vorbis, MP3, WAV, archivos AIFF, AU, LOF. Importar MIDI y RAW.
- Corta, copia, pega y mezcla sonidos. Puede trabajar con múltiples pistas.
- Se puede eliminar ruido, normalizar, ecualizar, amplificar, cambiar velocidad, modificar tono...
- Es posible programar cualquier efecto musical y/o sonoro, mediante el panel Nyquist, usando Xlisp

Una lista completa de las características de Audacity pueden ser leídas en [1].

10.4. Otros programas

Además de los editores de audio existen numerosas herramientas basadas en software libre relacionadas con el audio digital. Algunas de estas aplicaciones son:

Ardour edición de audio y grabación multipistas. Se puede consulta su sitio web en [2].

Pure Data entorno gráfico de programación en tiempo real, diseñado para el procesamiento de sonido, vídeo y gráficos. Su web oficial es [3].

CheeseTracker secuenciador basado en trackers. Su página oficial es [4].

RoseGarden secuenciador profesional basado en partituras. Más información en [5].

ZynAddSubFX sintetizador virtual. Si se desea información adicional se puede consultar [6].

Si se desea obtener información sobre otros programas complementarios se puede consultar [7].

10.5. Conclusiones

Como vemos, la edición de audio digital hoy en día resulta sumamente sencilla. Hace unos años, poseer un equipo de edición de audio, mezcla, *masterización* y aplicación de efectos hubiera supuesto un desembolso disparatado, accesible sólo a unos pocos bolsillos. Hoy, en cambio, cualquier usuario de un ordenador puede realizar todas estas tareas con software libre, siendo el único desembolso el equipo hardware usado. Esto ha posibilitado el auge de los *home studios* y capacitando a muchos compositores a editar y *masterizar* ellos mismos sus propios temas.

Bibliografía

- [1] Varios *Página principal de Audacity.* <http://audacity.sourceforge.net/about/features>
- [2] Ardour. <http://www.ardour.org/>
- [3] Pure Data. <http://www.puredata.info/>
- [4] CheeseTracker. <http://freshmeat.net/projects/cheesetracker/>
- [5] RoseGarden. <http://www.rosegardenmusic.com/>
- [6] Zynaddsubfx. <http://zynaddsubfx.sourceforge.net/>
- [7] Wikipedia *software de audio libre.* http://es.wikipedia.org/wiki/Software_de_audio_libre

Introducción a la creación y tratamiento de imágenes digitales mediante *Gimp*

Ignacio Montoya García

ignacio.montoya@uca.es

Oficina del Software libre de la Universidad de Cádiz

Juan Carlos Gonzalez Cerezo

juanca.gonzalez@uca.es

Oficina del Software libre de la Universidad de Cádiz

Resumen

Este artículo presenta *Gimp*, la herramienta de creación y tratamiento de imágenes digitales más popular en la comunidad de software libre, realizando un recorrido rápido por algunas de sus opciones y mostrando algunas de sus capacidades más interesantes.

11.1. Introducción

Gimp (GNU Imagen Manipulation Programm), tal y como podemos leer en su página web [1], es el programa de manipula-

ción de imágenes GNU [2]. Es un software libremente distribuido mediante licencia GPL [3] que sirve para llevar a cabo tareas como retoque fotográfico, composición de imágenes y creación de imágenes de autor. Los usos habituales de *Gimp* abarcan desde los propios de un programa simple de dibujo o diseño gráfico digital a los de uno de retoque fotográfico profesional. También se puede usar como traductor de formatos gráficos, para crear imágenes animadas o incluso como un sistema automatizado de proceso y renderización de imágenes.

Una de las principales bondades de *Gimp*

es que está portado libremente a multitud de plataformas y sistemas operativos. La mayoría de las distribuciones GNU/Linux de hecho incluyen *Gimp* como el editor gráfico estándar.

11.2. La interfaz gráfica

Al contrario que en la mayoría de programas de tratamiento de imágenes, la interfaz de *Gimp* no consiste en una única ventana dentro de la cual se encuentran la imagen y el acceso a los menús. *Gimp* usa ventanas independientes para el gráfico que se va a editar, el menú de herramientas y otros menús y paneles de control (ver figura 11.1). Se podría hacer una división de las ventanas de *Gimp* en tres categorías principales; La ventana o caja de herramientas, la ventana en la cual se encuentra la imagen con la que se está trabajando y las denominadas ventanas de diálogo.

El control básico se lleva a cabo normalmente a través de la caja de herramientas, la cual permite el acceso rápido mediante iconos a tareas específicas. Por otro lado, el acceso al resto de herramientas se realiza, bien a través de la barra de herramientas ubicada en la parte superior de la ventana de imagen, bien a través del menú contextual de la misma. Las ventanas de diálogo permiten el acceso a opciones de tratamiento de la imagen específicas, como por ejemplo canales, capas, caminos, colores, brochas, gradientes, modelos y paletas.

Una de las características más interesantes de la interfaz de *Gimp* es su capacidad de ser adaptada por el usuario a sus necesidades de forma muy sencilla. Por ejemplo, el usuario puede configurar la caja de herramientas principal añadiendo o quitando

las opciones que desee de la misma. Además, aunque provee por defecto combinaciones de teclas de acceso rápido para la mayoría de opciones y funciones, *Gimp* permite que el usuario pueda configurar a su gusto dichos accesos rápidos para cualquier opción o función del programa.

11.3. Formatos gráficos

El formato gráfico nativo de *Gimp* es XCF (.xcf); En dicho formato *Gimp* guarda toda la información relativa a la imagen tanto sobre las capas como sobre el resto de información general de la imagen y específica de *Gimp*. Por otro lado, *Gimp* es capaz de trabajar sin problema en modo lectura o escritura con imágenes que estén en la mayoría de los formatos gráficos existentes más comunes, tanto bitmap como vectoriales: BMP, GIF, JPG, PCX, PNG, PS, TIF, TGA, XPM, DCM, ICO, PSD, PPM, SGI, RGB, etc.

11.4. Textos

El soporte de textos en *Gimp* se basa en cómo maneja texto el sistema X Window [4]. *Gimp* tiene herramientas para insertar textos en las imágenes usando cualquier fuente instalada en el sistema. Una vez introducido el texto, éste sigue siendo editable a través de las herramientas de manejo de textos. Al introducir un texto en *Gimp* éste se trata como una nueva capa de la imagen sobre la cual estamos trabajando. De esta forma podremos aplicar sobre dicha capa de texto cualquiera de los tratamientos o efectos que deseemos.

De hecho, una de las principales aplicaciones prácticas de *Gimp* es la creación de



Figura 11.1: Interfaz gráfica de *Gimp*

logotipos. Mediante la introducción de texto y aplicando muy pocos efectos se pueden conseguir llamativos logotipos de forma muy sencilla y rápida.

11.5. Color

Aunque el modo de color más recomendado para trabajar en una imagen digital es el modo RGB (ya que nos asegura imágenes con profundidad de color de 24 bits) al trabajar con una imagen en *Gimp* se puede hacer en base a tres modos de color: RGB, escala de grises y color indexado. En cualquier momento de hecho tenemos la opción de cambiar la imagen entre cualesquiera de dichos modos.

Las herramientas de trabajo con color que ofrece *Gimp* son muchas y variadas. Algunas de las más representativas son las siguientes:

Recolector de color: Nos permite adquirir el color de un píxel concreto para tra-

bajar con él.

Paletas de colores: Nos ofrecen distintas maneras de seleccionar un color. Introduciendo el valor hexadecimal RGB del mismo, mediante valores de tono, saturación y brillo o bien a través de alguna de las herramientas de selección visual.

Histograma: Analiza los píxeles de una imagen y los interpreta de forma gráfica, ofreciéndonos una amplia información sobre la calidad de la imagen.

Niveles: Mediante esta herramienta podemos reducir o aumentar las sombras, medios tonos y luces de la imagen o de algunas zonas determinadas de la misma simplemente mediante una simple pulsación de ratón.

Curvas: Esta es quizás la herramienta más potente para trabajar sobre la colorización de una imagen ya que permite

afinar al máximo las modificaciones realizadas sobre píxeles concretos de la misma.

Corrección de color: Aquí se agrupan una serie de herramientas como son el balance de color, ajustes de brillo y contraste, ajustes de tono y saturación, colorización, umbral de color y posterización.

11.6. Herramientas de selección

Se podría realizar una primera división de las herramientas de selección que incluye *Gimp* en dos grupos básicos:

Selección mediante dibujo de contornos:

El usuario crea un contorno y selecciona después el interior o el exterior del mismo.

Selección especificando píxeles representativos.

El usuario decide una característica de selección del píxel y *Gimp* se encarga de seleccionar todos aquellos píxeles que contengan dicha característica.

Aparte de las opciones habituales (como son cortar, copiar, mover y pegar), sobre las selecciones se pueden realizar prácticamente todas las operaciones que se pueden llevar a cabo sobre la imagen total. Aparte de eso, *Gimp* permite la adición, substracción e intersección de una o varias selecciones entre sí.

11.7. Capas

Aunque *Gimp* es en principio una herramienta de manipulación de imágenes basa-

da en el píxel, en un nivel superior *Gimp* está orientado al trabajo con capas. Las ventajas más importantes de manejar una imagen en diferentes capas son que podemos desplazar ciertos elementos o figuras en cualquier momento y aplicarles tratamientos gráficos de cualquier tipo sin afectar a la integridad del resto de la escena. *Gimp* soporta todo tipo de operaciones sobre capas: creación o borrado, cambio de orden y fusión de las mismas.

11.8. Extensiones

Quizás uno de los puntos fuertes de *Gimp* sea su modularidad, la cual le proporciona capacidad de ampliación mediante extensiones. Existe una amplia base de datos [5] donde podemos consultar y acceder libremente a todas las extensiones existentes, así como enviar aquellas que nosotros creamos. Los dos tipos de extensiones más comunes y usados en *Gimp* se pueden agrupar en filtros y Script-Fu.

11.8.1. Filtros

El origen de los filtros proviene de la fotografía convencional en la cual se añadían lentes distintas a la cámara para conseguir efectos sobre la imagen. *Gimp* dispone de muchos tipos distintos de filtros digitales para conseguir efectos sobre la imagen digital: desenfoque, ruido, distorsión, efectos artísticos y de renderizado o mapas de bits son algunos de ellos.

11.8.2. Script-Fu

Básicamente, los Script-Fu son automatizaciones de tareas para aplicar distintos tratamientos y/o efectos sobre una imagen. Un Script-Fu puede consistir por ejemplo en aplicar varios filtros sobre una imagen en un orden determinado. La instalación estándar de *Gimp* incorpora varios Script-Fu, aunque en cualquier momento se le pueden añadir nuevos Script-Fu. Cualquiera puede escribir nuevos Script-Fu en su propio lenguaje *Schema* [6] o bien en *Perl* [7] a través de los llamados scripts de *Perl-Fu* [8].

11.9. Conclusiones

Gimp es una de las aplicaciones de tratamiento de imágenes digitales más potentes que existen actualmente. Ofrece prácticamente las mismas funcionalidades que cualquier otra aplicación propietaria de tratamiento de imágenes digitales que podamos encontrar en el mercado con todas las ventajas añadidas innatas a un programa con licencia GPL [3].

Gimp cuenta con una importante comunidad de usuarios y desarrolladores [9] que ofrece extensiones y soporte al programa en forma de tutoriales, artículos, listas de correo, foros o libros como por ejemplo *Grokking the Gimp* [10].

Bibliografía

- [1] *Gimp*: <http://www.Gimp.org>
- [2] GNU: <http://www.gnu.org>
- [3] GPL: <http://www.gnu.org/copyleft/gpl.html>
- [4] Fundación X.Org: <http://www.x.org/>
- [5] Extensiones para Gimp <http://registry.Gimp.org/index.jsp>
- [6] Schema in Gimp http://www.Gimp.org/tutorials/Basic_Scheme/
- [7] Perl <http://www.perl.org>
- [8] Perl-Fu http://www.Gimp.org/tutorials/Basic_Perl/
- [9] Grupo de usuarios de Gimp <http://gug.sunsite.dk/>
- [10] Bunks, Carey: *Grokking the Gimp* New Riders Publishing 2000, Open Publication License http://Gimp-savvy.com/BOOK/index.html?Grokking_the_Gimp.html

Creación de presentaciones en DVD con dvd-slideshow

Manuel Palomo Duarte

manuel.palomo@uca.es

Depto. de Lenguajes y Sistemas Informáticos. Universidad de Cádiz.

Resumen

Este artículo presenta las posibilidades que ofrece el sistema operativo GNU/LINUX para crear DVDs con contenidos multimedia haciendo uso de la herramienta dvd-slideshow.

12.1. Introducción

Hoy en día el DVD se ha hecho un hueco en nuestra vida diaria. Existen reproductores de DVDs (u ordenadores con capacidad de reproducirlos) en prácticamente todos los hogares y centros de enseñanza. Incluso existen lectores portátiles que incorporan una pantalla para poder visualizarlos en cualquier lugar.

Aparte de servir para visualizar películas, el DVD puede facilitar la labor de los

profesionales de la enseñanza, ya sea apoyando la docencia presencial o como medio clave en la docencia a distancia.

Así pues, el objetivo de este artículo es presentar las bondades del sistema dvd-slideshow para crear DVDs fácilmente a partir de galerías de fotos así como presentar otras herramientas que puedan complementarlo.

12.2. dvd-slideshow

El paquete dvd-slideshow incluye varios programas (todos ellos distribuidos bajo licencia *GNU/GPL*) que facilitan la creación de DVDs que muestren una sucesión de fotografías. Permite añadirle música o sonido, así como efectos del tipo *fadein*, recorte de fotografías, etc.

Su página principal está en [1], pero se

recomienda su descarga en paquete *.deb* o *.rpm* desde el servidor de *freshmeat* [2], pues el sistema tiene numerosas dependencias.

12.2.1. Programa principal

El programa principal se llama como el paquete, *dvd-slideshow*. Es un *script* programado en *bash* que se invoca desde línea de órdenes.

Su entrada es un fichero de texto que indica los datos del DVD a generar: las fotos que se mostrarán, el orden de aparición, el tiempo de exposición en pantalla de cada una, la música que sonará en cada momento, si aparecerán subtítulos (esta característica es interesante para añadir información a las fotos o para facilitar la accesibilidad a sordos), etc. Las fotos se aceptan en formato JPG o PNG, y el sonido en MP3.

Por ejemplo, para mostrar una serie de fotos se puede usar el código mostrado en la figura 12.1 (basado en el ejemplo *very-simple.txt* de [1]): La salida del programa son un par de carpetas con los ficheros de vídeo y audio en formato MPEG2, preparadas para ser grabadas en un DVD con algún programa como *dvdauthor* [3] (que funciona desde línea de órdenes) o *QDVDAuthor* [4] (su equivalente en modo gráfico). Además genera un fichero XML para crear posteriormente un menú para el DVD con el programa *dvd-menu*.

Para personalizar el comportamiento del programa, cada usuario puede tener un fichero llamado *.dvd-slideshowrc* en su directorio de entrada que proporcione valores por defecto para el programa.

12.2.2. Otros programas relacionados

El mismo paquete incluye algunos programas que extienden las posibilidades de *dvd-slideshow*, como *dvd-menu* que crea un menú para el DVD con de las indicaciones que genera *dvd-slideshow*. O *dir2slideshow*, que genera un fichero de entrada para *dvd-slideshow* a partir de las fotos de un directorio determinado.

En el caso de que las fotos se tengan organizadas en galerías digitales, los programas *gallery1-to-slideshow* y *jigl2slideshow* generan un fichero de entrada para *dvd-slideshow* leyendo los datos de una galería de fotos creada con el programa *Gallery* versión 1 o *jigl*.

Otro programa muy interesante es *Slideshow Creator*. Es una interfaz gráfica para crear o modificar ficheros de entrada de *dvd-slideshow*, que permite visualizar en entorno X-Windows las fotos en concreto y sus parámetros. En la figura 12.2 se muestra una captura proporcionada por la web del programa [5].

Por último pueden resultar de interés, como herramientas auxiliares a *dvd-slideshow*, el programa de retoque fotográfico *gimp*, el programa de dibujo lineal *xfig*, la grabadora de micrófono *krecord* o *lame*, que codifica ficheros WAV (como los generados por *krecord*) en formato MP3.

12.3. Alternativas

Existen otras alternativas para crear presentaciones en otros formatos, que pueden complementar en algunos casos a *dvd-slideshow*:

LaTeX Beamer Class Permite crear diapositivas para mostrar en videoproyec-

```

# Las líneas que comienzan
# por almohadilla son comentarios

# Con "title" se escribe en pantalla
# title:duración:descripción
# La duración es en segundo.
title:5:Ejemplo de presentación

# Cada foto necesita una línea. Formato:
# fichero.jpg:duración:subtítulo
# La duración es en segundos.
# El subtítulo es opcional.
foto1.jpg:4:Introducción
foto2.jpg:8:Primera
fase grafico008.jpg:7

```

Figura 12.1: Ejemplo de código para mostrar una serie de fotos

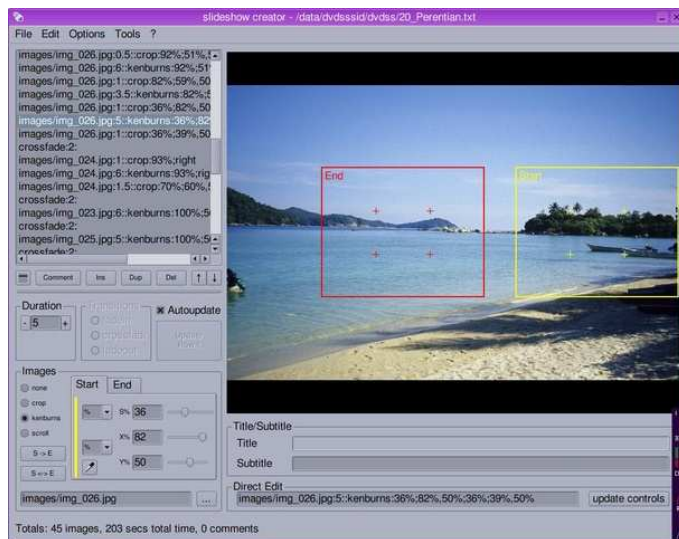


Figura 12.2: Programa Slideshow Creator en funcionamiento.

tor desde \LaTeX o LyX usando la clase *Beamer*. Automatiza la creación de índices, barras de navegación, bibliografía y otras características de interés. Si se desea más información se puede consultar [6].

S5 Permite crear presentaciones con XHTML, CSS y Javascript. A partir de una misma entrada produce una versión para presentaciones vía web y otra para imprimir. Los datos de entrada se escriben usando un subconjunto de XHTML, y el aspecto gráfico se define con CSS. Su página principal ofrece información adicional y ejemplos, [7].

RaccoonShow Es un programa que crea una presentación en Macromedia Flash leyendo los datos de un fichero PDF y (opcionalmente) con música de un fichero en formato WAV. Su página principal en Internet es [8].

Tpp Este programa lee la información de entrada de un fichero de texto y crea una presentación para mostrarla en cualquier consola soportada por la biblioteca *ncurses*. Para obtener información adicional se puede consultar [9].

12.4. Conclusiones

Dvd-slideshow facilita la creación de DVDs. Sus aplicaciones son muy diversas, desde la creación de DVDs personales con colecciones de fotos a creación de contenidos didácticos.

Al estar programado en un lenguaje de shell como *bash* es fácil interactuar con la

shell, abriendo un mundo de posibilidades. Además, su entrada es un fichero de texto plano, por lo que se puede automatizar la generación de diversas secuencias.

Es un programa de muy reciente creación (su primera versión apareció en 2003), pero con un crecimiento rápido. Constantemente aparecen versiones que amplían sus capacidades así como extensiones para poder conectarlo a otros programas.

Bibliografía

- [1] Dylewski, Scott. *Página principal de dvd-slideshow*. <http://dvd-slideshow.sourceforge.net/>
- [2] Dylewski, Scott. *Descarga de dvd-slideshow*. <http://freshmeat.net/projects/dvdslideshow/>
- [3] Smith, Scott. *Página principal de dvdauthor*. <http://dvdauthor.sourceforge.net/>
- [4] Okan, Varol. *Página principal de qdvdauthor*. <http://qdvdauthor.sourceforge.net/>
- [5] Colnaghi, Marco. *Página principal de Slideshow Creator*. <http://slcreator.sourceforge.net/>
- [6] Tantau, Till. *Página principal en Internet de la clase Beamer para L^AT_EX*. <http://latex-beamer.sourceforge.net/>
- [7] Meyer, Eric A. *Página principal en Internet de S5*. <http://www.meyerweb.com/eric/tools/s5/>
- [8] Bacon, Jono. *Página principal en Internet de RaccoonShow*. <http://www.jonobacon.org/projects/raccoonshow/>
- [9] Krennmair, Andreas & Golde, Nico. *Página principal en Internet de tpp*. <http://synflood.at/tpp.html>

Índice de Autores

- Almeida, Francisco, 103
- Blanco, Vicente, 103
- Briebe Sánchez, José, 15
- Carmona Murillo, Javier Domingo, 43
- Castuera Toro, Montaña, 15
- del Castillo, Álvaro, 80
- Fernando Sánchez, Juan Fernando, 69
- García Leiva, Rafael, 80
- Gato, José, 80
- Gazo Cervero, Alfonso, 43
- González Sánchez, José Luis, 43
- Gonzalez Cerezo, Juan Carlos, 117
- Gonzalez Sánchez, José Luis, 15
- Hernández Fernández, José Julio, 5
- Herrero Rodríguez, Ismael, 80
- Jimenez-Peris, Ricardo, 80
- Martín, María Isabel, 80
- Martínez Bravo, Lorenzo, 43
- Meléndez Aganzo, Luis, 29
- Montoya García, Ignacio, 117
- Moreno Caballero, Ismael, 99
- Orús Cacho, Álvaro, 80
- Palomo Duarte, Ignacio, 113
- Palomo Duarte, Manuel, 125
- Paramio Danta, Carlos Alberto, 57
- Patiño, Marta, 80
- Rioja del Río, Carlos, 99
- Ruiz Arahall, Manuel, 5
- Sánchez Monedero, Javier, 29
- Sánchez, José Antonio, 80
- Santos, Adrián, 103
- Ventura Soto, Sebastián, 29