



Proceedings of the FLOSS International Conference 2007

**J. Rafael Rodríguez Galván
Manuel Palomo Duarte
Coords.**

Published by:



.FIC | FLOSS International Conference

Organizadores:



Patrocinadores ORO:



Patrocinadores:



Entidades Colaboradoras:



Medios Asociados:



Colaboran: Vicerrectorado de Alumnos - Consejo Social Universidad de Cádiz - Facultad de Ciencias Sociales y de la Comunicación
Vicerrectorado de Investigación, Desarrollo Tecnológico e Innovación - Vicerrectorado de extensión universitaria de la Universidad de Cádiz

Este libro se distribuye bajo licencia Creative Commons Reconocimiento-CompartirIgual 2.5 España

Usted es libre de:

- Copiar, distribuir y comunicar públicamente la obra
- Hacer obras derivadas

Bajo las condiciones siguientes:

- Reconocimiento. Debe reconocer los créditos de la obra de la manera especificada por el autor o el licenciador (pero no de una manera que sugiera que tiene su apoyo o apoyan el uso que hace de su obra).
- Compartir bajo la misma licencia. Si altera o transforma esta obra, o genera una obra derivada, sólo puede distribuir la obra generada bajo una licencia idéntica a ésta.
- Al reutilizar o distribuir la obra, tiene que dejar bien claro los términos de la licencia de esta obra.
- Alguna de estas condiciones puede no aplicarse si se obtiene el permiso del titular de los derechos de autor
- Nada en esta licencia menoscaba o restringe los derechos morales del autor, ni tan siquiera el derecho de transformación cedido mediante ella.

Coordinan:

J. Rafael Rodríguez Galván
Manuel Palomo Duarte
Universidad de Cádiz

Edita: Servicio de Publicaciones de la Universidad de Cádiz
C/ Doctor Marañón, nº 3, 11002 Cádiz
<http://www.uca.es/publicaciones>
ISBN: 978-84-9828-124-8

Table of Contents

Apertium, una plataforma de código abierto para el desarrollo de sistemas de traducción automática	5
REST: Representational State Transfer, en Ruby on Rails	21
Babel: A collaborative language learning system	27
Sistema experto para la simulación de sistemas tácticos de baloncesto con software libre	38
3D Distributed Rendering and Optimization using Free Software	52
Servicios de información sobre la EPSJ basados en entornos de publicación	67
El uso de software libre en los sitios web universitarios españoles	81
Libre software for research	97
JavaVis: open source code for computer vision subjects	106
El reto de los servicios Web para el software libre	116
An Open Source Framework Oriented to Modular Web-Solution Development based in Software Product Lines	133
SIGA: Sistema Integrado de Gestión de Aulas	141
siLeDAP: Implementando XLDAP	152
Primer Concurso Universitario de Software Libre	159
Cursos accesibles y reusables sobre la plataforma ALPE	170
Sistema de Colas Condor	186
Why using dotLRN? UNED use cases	195

Euspell: corrección ortográfica del euskera en software libre	213
HIGEIA, la interoperabilidad sanitaria	221
Código fuente de Linux en la docencia de Sistemas Operativos I	231
Accessibility Requirements for educational packages in dotLRN	240
phpRADmin project	253
Proyecto RESTAD: Herramientas de Código Libre para la Traducción y Postedición de Documentos	262
El software libre en la docencia de multimedia	270
Experiencias en el Desarrollo de una Taxonomía de Estándares Abiertos	280

Apertium, una plataforma de código abierto para el desarrollo de sistemas de traducción automática

Carme Armentano-Oller, Antonio M. Corbí-Bellot, Mikel L. Forcada,
Mireia Ginestí-Rosell, Marco A. Montava Belda, Sergio Ortiz-Rojas,
Juan Antonio Pérez-Ortiz, Gema Ramírez-Sánchez y Felipe Sánchez-Martínez

Grup Transducens
Departament de Llenguatges i Sistemes Informàtics
Universitat d'Alacant, E-03071 Alacant

{carmentano, acorbi, mlf, mginesti}@dlsi.ua.es
{amontava, sortiz, japerez, gramirez, fsanchez}@dlsi.ua.es

<http://transducens.dlsi.ua.es>

Resumen Uno de los principales retos de la informática para las próximas décadas es el desarrollo de sistemas capaces de procesar eficazmente el lenguaje natural (o lenguaje humano). Dentro de este campo, los sistemas de traducción automática, encargados de traducir un texto escrito en un idioma a una versión equivalente en otro idioma, reciben especial atención dado, por ejemplo, el carácter multilingüe de sociedades como la europea. La automatización de dicho proceso es particularmente compleja porque los programas han de enfrentarse a características del lenguaje natural, como la ambigüedad, cuyo tratamiento algorítmico no es factible, de modo que una mera aproximación o automatización parcial del proceso ya se considera un éxito. Los programas de traducción automática han sido tradicionalmente sistemas cerrados, pero en los últimos tiempos la tendencia marcada por el software libre ha llegado también a este campo. En este artículo describimos Apertium, apertium.org, una plataforma avanzada de código abierto, con licencia GNU GPL, que, gracias al desacoplamiento que ofrece entre datos y programas permite desarrollar cómodamente nuevos traductores automáticos.

La plataforma Apertium ha sido desarrollada por el grupo de investigación Transducens de la Universitat d'Alacant en el marco de varios proyectos de colaboración con universidades y empresas de España en los que, además de los programas que conforman el motor de traducción, se han confeccionado datos lingüísticos abiertos para la traducción automática catalán-español, gallego-español, portugués-español, francés-catalán, inglés-catalán y occitano-catalán. Tanto la plataforma en la que se integra el motor de traducción como los datos para estos pares de lenguas están disponibles para su descarga en sf.net/projects/apertium/ y para su evaluación en línea en xixona.dlsi.ua.es/prototype/.

Palabras clave: Apertium, traducción automática de código abierto, datos lingüísticos de código abierto, procesamiento del lenguaje natural

1. Introducción

Los sistemas de *traducción automática*, encargados de traducir un texto escrito en un idioma a una versión equivalente en otro idioma, son cada vez más demandados dado el carácter multilingüe de las complejas redes sociales, organizativas o comerciales actuales. La automatización del proceso de traducción es particularmente compleja porque los programas han de enfrentarse a características del lenguaje natural, como la ambigüedad, cuyo tratamiento algorítmico no es factible, de modo que una mera aproximación o automatización parcial del proceso ya se considera un éxito.

Los programas de traducción automática han sido tradicionalmente sistemas cerrados, tanto a nivel de código como de datos (pese a que algunos de ellos pueden utilizarse, con ciertas restricciones, en internet), lo que impide adaptarlos a nuevos pares de lenguas o dominios, integrarlos con otras aplicaciones, o usar sus recursos en proyectos de investigación o desarrollo. La complejidad inherente a este tipo de sistemas, que requiere un esfuerzo considerable de implementación, junto a la propia tendencia del mercado, han sido durante muchos años las principales razones de la falta de alternativas abiertas. Sin embargo, en los últimos tiempos el cambio de enfoque provocado por el software libre está llegando tímidamente también a este campo.

En este artículo describimos Apertium, uno de los principales exponentes de sistemas de traducción automática de código abierto. Apertium¹ es una plataforma avanzada de código abierto, con licencia GNU GPL,² que permite prototipar y construir traductores automáticos eficientes sin necesidad de tener que comenzar su desarrollo desde cero. Apertium ha dejado atrás la fase de prototipado y ya es un sistema de alto interés tanto comercial como científico. Por un lado, Apertium está siendo usado por empresas en proyectos reales tales como la edición multilingüe de noticias en prensa; por otro lado, la completa disponibilidad de la plataforma y de los datos asegura la reproducibilidad científica de las investigaciones publicadas en el área de la traducción automática que se basen en Apertium.

El resto del artículo presenta los siguientes contenidos. En la sección 2 se introduce el concepto de traducción automática, sus principales ámbitos de aplicación y algunas ideas acerca de por qué es un clásico problema “difícil” en informática; además, se introducen las principales técnicas con las que se diseñan los sistemas de traducción automática y las aproximaciones actuales basadas en código abierto. En la sección 3 se describe Apertium, una plataforma de código abierto formada por programas y herramientas que permiten desarrollar e implementar sistemas de traducción automática de forma eficiente. Finalmente, las secciones 4 y 5 presentan las oportunidades que se abren a nivel económico

¹ <http://www.apertium.org>

² <http://www.gnu.org/copyleft/gpl.html>

y científico, respectivamente, ante la disponibilidad de sistemas abiertos como Apertium. El artículo termina con unas conclusiones a modo de recapitulación.

2. La traducción automática

La *traducción automática* consiste en la obtención de un texto *equivalente* (esto es, que preserve el contenido) en una *lengua destino* a partir de un texto en una *lengua origen*. Estamos hablando, en cualquier caso, de lenguas *naturales* (como, por ejemplo, el español, el inglés o incluso el esperanto), no de lenguajes informáticos (como, por ejemplo, Java o XML); la traducción entre lenguajes informáticos es una cuestión totalmente diferente (aunque puede compartir algunas técnicas básicas con la traducción automática) que se aborda en el campo del diseño de compiladores.

La traducción automática ha sido objeto de estudio desde los primeros tiempos de la informática en los años cincuenta y a día de hoy es un problema solo parcialmente resuelto. Es una de las áreas que más atención recibe dentro del *procesamiento del lenguaje natural*; este consiste en el estudio de aspectos como la formalización, la generación y la comprensión del lenguaje natural en todas sus formas, para, por ejemplo, permitir la realización de búsquedas en internet utilizando un lenguaje coloquial o la interacción con el ordenador a través de la voz.

2.1. ¿Por qué la traducción automática es difícil?

La ciencia actual no es capaz de expresar de manera formal, mediante reglas, todos los mecanismos que subyacen a los lenguajes naturales ni los procesos mentales involucrados en su uso. Esta imposibilidad para obtener una caracterización precisa, común a otros muchos campos de la inteligencia artificial, es uno de los principales argumentos que pueden darse para explicar la dificultad de escribir un programa de ordenador que traduzca textos. Sin embargo, podría aducirse que un entendimiento de los mecanismos profundos del funcionamiento del lenguaje natural puede no ser necesario para remedar el uso que las personas hacemos de él. Aún así, son muchos los elementos que hacen que la traducción automática siga siendo un problema difícil de resolver. Arnold [3] clasifica estos problemas en los siguientes grupos:

1. *La forma no determina completamente el contenido.* Esto se debe a la presencia de *ambigüedad* a distintos niveles: si decimos “el gato está debajo del coche”, ¿a qué nos estamos refiriendo, a un felino o a una herramienta?; si decimos, “he visto a tu hermano con el telescopio”, ¿quién de los dos es el que tiene un telescopio? Muchas veces es necesario *resolver* estas ambigüedades de cara a obtener una traducción correcta; por ejemplo, ¿cuál es el pronombre personal que ha de aparecer en la traducción al inglés de “tu amigo le dijo la verdad”, el pronombre *him* o el pronombre *her*? Aunque en ocasiones el contexto puede ayudar a la hora de resolver una ambigüedad, es

muy complicado hacer que un programa de ordenador sea capaz de analizar siempre el contexto adecuadamente para tomar una decisión. Muchas veces, lo que se necesita para resolver estos casos es lo que llamamos *conocimiento del mundo*, que los humanos poseemos y que es muy difícil sistematizar e introducir en un programa de ordenador.

2. *El contenido no determina completamente la forma.* Existen multitud de formas de expresar en una lengua un contenido dado y para que un ordenador no deba enfrentarse a la complejidad que esto acarrea, es necesario imponer estrategias que reduzcan las posibilidades, aunque sea a costa de perder expresividad.
3. *Distintas lenguas usan estructuras diferentes para expresar las mismas cosas.* Consideremos, la frase “me gustan las manzanas”; su traducción al inglés es “I like apples”, donde puede verse cómo el sujeto de la frase en español (*las manzanas*) se ha transformado en un complemento directo en el caso del inglés (*apples*). Aunque este es un ejemplo relativamente sencillo, en general, las estructuras usadas por lenguas distintas pueden ser tan divergentes que hagan que una simple traducción directa sea ininteligible.
4. Finalmente, como se ha comentado al inicio de este apartado, es difícil caracterizar con la precisión necesaria los principios involucrados en el uso del lenguaje. Existen, por otro lado, sistemas de traducción automática que mediante técnicas estadísticas de aprendizaje computacional son capaces de inferir nuevas traducciones a partir de ejemplos, pero la cantidad necesaria de estos ejemplos de traducciones es ingente y no siempre están disponibles para un par de lenguas dado.

Estos problemas se reducen ostensiblemente cuando las lenguas implicadas en la traducción están *emparentadas*. En ese caso las afinidades a nivel morfológico, sintáctico y semántico simplifican el diseño de estos sistemas y permiten llegar fácilmente a traducciones en las que solo un 5% del texto es incorrecto.

Por otro lado, existen tipos de textos que pueden ser más fácilmente traducidos (cartas comerciales, manuales de instrucciones, textos económicos) y otros cuya traducción automática es a día de hoy totalmente inviable (poesía, por ejemplo).

2.2. Usos de la traducción automática

Las distintas aplicaciones de la traducción automática se pueden dividir en tres grandes grupos: *asimilación*, *comunicación* y *diseminación*. En situaciones de *asimilación*, la traducción resultante se utiliza para hacerse una idea general del contenido del texto original, en este caso la calidad de la traducción no es relevante en tanto el lector pueda satisfacer este deseo. Un uso de la traducción automática cercano al anterior es el que se produce cuando dos o más personas que hablan idiomas distintos se *comunican* (vía chat o correo electrónico, por ejemplo) a través de un sistema de traducción automática. Finalmente, para la *diseminación* se precisa de una traducción de mayor calidad ya que el texto traducido se difunde públicamente. En este último caso, la calidad de la traducción resultante se puede conseguir de diversas formas:

- restringiendo el lenguaje origen, bien reduciendo el dominio lingüístico (por ejemplo, centrándose en el lenguaje de los partes meteorológicos o de las cartas comerciales), bien controlando el tipo de construcciones o palabras que pueden ser usados (por ejemplo, evitando el uso de oraciones subordinadas o de palabras polisémicas en el texto origen); en el primer caso se habla de traducción de *sublenguajes* y en el segundo caso de traducción de *lenguajes controlados*; lo más habitual es combinar ambas estrategias;
- corrigiendo la traducción automática resultante, lo que se conoce como *postedición*, y que constituye la forma más habitual de obtener traducciones adecuadas para la diseminación.

En cualquier caso, cuanto mejor sea el sistema de traducción automática menor será el esfuerzo humano necesario para obtener una traducción adecuada. A día de hoy, la *traducción de gran calidad completamente automatizada* es todavía una quimera, pero esto no invalida el uso de la *traducción automática* en tareas de diseminación, ya que en ciertos casos sale más rentable posteditar un texto traducido automáticamente que traducirlo manualmente desde cero. En el caso de la asimilación, por otro lado, muchos internautas utilizan sistemas de traducción automática para desenvolverse adecuadamente en servicios que no están disponibles en los idiomas que conocen; por ejemplo, para realizar compras en internet.

Evidentemente, en un entorno como el actual cada vez más globalizado y marcado por un vertiginoso ritmo de producción y consumo de información, la demanda de sistemas de traducción automática se ha disparado a la vez que surgen nuevas especializadas dentro de la profesión de traductor, como es la del *posteditor*. Por otro lado, el interés de algunas administraciones y el valor que en la nueva economía adquieren las minorías lingüísticas está provocando la aparición paulatina de sistemas de traducción que involucran lenguas minoritarias; considérese, por ejemplo, la reciente publicación de un sistema de traducción automática basado en Apertium entre el catalán y el aranés (lengua cooficial en Cataluña, usada en el pequeño territorio del Valle de Arán) [2].

2.3. Técnicas de traducción automática

A la hora de construir sistemas de traducción automática se suelen seguir dos grandes enfoques: el de los sistemas *basados en reglas* y el de los sistemas basados en *corpus* (colecciones de documentos); los ejemplos más destacados de sistemas de traducción automática basados en corpus son los sistemas *estadísticos* y los sistemas *basados en ejemplos*. Por descontado, también son posibles enfoques híbridos.

En un sistema *basado en reglas* los datos lingüísticos (básicamente diccionarios monolingües, diccionarios bilingües y reglas de transferencia, que recogen las transformaciones estructurales necesarias para pasar de una lengua a otra) se recopilan manualmente y se codifican adecuadamente para que puedan ser usados por el motor de traducción. Los sistemas *basados en corpus* utilizan cadenas o patrones de traducción inferidos automáticamente a partir de las regularidades

observadas en *corpus paralelos*, esto es, corpus que contienen documentos en la lengua origen y su correspondiente traducción en lengua destino.

Los sistemas basados en reglas más habituales son los sistemas de *traducción automática por transferencia*, que funcionan mediante tres fases bien diferenciadas: análisis, transferencia y generación.

- La fase de *análisis* produce a partir de la frase en lengua origen una representación intermedia dependiente de la lengua origen.
- La fase de *transferencia* convierte la representación intermedia anterior en una nueva representación intermedia que, esta vez, es dependiente de la lengua destino, pero no de la de origen.
- La fase de *generación* produce una frase en lengua destino a partir de la representación intermedia obtenida en la fase anterior.

2.4. Traducción automática de código abierto

Un traductor automático de código abierto³ puede ser usado, copiado, estudiado, modificado y redistribuido con la única restricción de que el código fuente ha de estar siempre disponible. Para que el sistema sea de código abierto basta con acompañarlo con una licencia compatible con estos principios (tal como la licencia GNU GPL, la licencia Mozilla, las licencias de tipo BSD, etc.); si no se explicitan estos principios con una licencia, se aplican las restricciones habituales de los programas comerciales.

Las ideas del código abierto se pueden aplicar a elementos distintos del software siempre que esté claro qué es el *código fuente* del producto final. Por ejemplo, en muchos sistemas de traducción automática la forma original “textual” de los diccionarios se convierte (o se *compila*) en una forma “binaria” mucho más eficiente pero más complicada de editar; en este caso, las entradas del diccionario en forma textual se pueden considerar como el código fuente del diccionario.

Independientemente del enfoque, de entre los comentados en el apartado 2.3, que utilice un sistema de traducción automática, en todos los sistemas se pueden distinguir una serie de componentes comunes (aunque en función del sistema particular algunos componentes pueden no estar presentes o estar integrados en otros):

- el motor de traducción,
- los datos lingüísticos en forma “textual”,
- los *compiladores* (en un sentido general) que transforman los datos anteriores en una forma “binaria” más eficiente usada por el motor, y

³ Para los propósitos de este trabajo consideraremos *código abierto* (término acuñado por la Open Source Initiative, <http://www.opensource.org>) y *software libre* (término acuñado por la Free Software Foundation, <http://www.fsf.org>) como términos equivalentes; del mismo modo, usaremos como términos opuestos a estos los de *código cerrado* o *software comercial*. La terminología en el campo es bastante ambigua: por ejemplo, algunos programas comerciales tienen su código fuente publicado, pero no pueden considerarse como de código abierto, según la definición de la Open Source Initiative, debido a determinadas cláusulas restrictivas de la licencia.

- las herramientas de mantenimiento de los datos lingüísticos.

Para que un sistema de traducción automática se pueda considerar *abierto*, es necesario que el código fuente del motor de traducción y de los datos lingüísticos se distribuya con licencias adecuadas; debe tenerse en cuenta que es mucho más probable que los usuarios de un sistema de traducción automática de código abierto realicen modificaciones en los datos lingüísticos que que lo hagan sobre el código fuente del motor de traducción. Por otro lado, si se realizan determinados cambios en el código fuente del motor, puede ser necesario también modificar los compiladores, por lo que su código también debe ser abierto. Finalmente, si se alteran algunas características de los datos lingüísticos puede ser necesario cambiar las herramientas de mantenimiento. En definitiva, los cuatro componentes deben idealmente ser abiertos para que el sistema de traducción automática pueda considerarse estrictamente como tal.

Aunque los ejemplos de sistemas de traducción automática de código abierto no son abundantes, en los últimos años han aparecido varios de ellos. En este artículo describimos Apertium, un sistema de traducción automática totalmente abierto, según la definición anterior.⁴ En el terreno de los sistemas comerciales, la mayoría de los traductores automáticos han estado tradicionalmente basados en reglas⁵, pero cada vez son más los sistemas basados en corpus⁶.

3. Apertium, una plataforma de código abierto para la traducción automática

OpenTrad Apertium⁷ (Apertium para abreviar) es una plataforma de código abierto que incluye las herramientas y programas necesarios para construir y ejecutar sistemas de traducción automática basados en reglas, aunque alguno de sus módulos son de tipo estadístico o híbrido; los traductores construidos con Apertium pueden llegar a traducir decenas de miles de palabras por segundo.⁸ Originalmente, Apertium se diseñó como un sistema de transferencia superficial

⁴ Otros sistemas total o parcialmente abiertos son OpenLogos, la versión de código abierto del sistema comercial basado en reglas Logos, <http://logos-os.dfki.de>, o el sistema de traducción estadística Moses, <http://www.statmt.org/moses/>; en el pasado se realizaron algunos intentos de desarrollar sistemas de traducción automática de código abierto, que quedaron incompletos, como GPLTrans, <http://www.translator.cx>, Traduki, <http://traduki.sourceforge.net>, o Linguaphile, <http://linguaphile.sourceforge.net>.

⁵ Por ejemplo, Reverso, <http://www.reverso.net>, o Babylon, <http://www.babylon.com>.

⁶ Por ejemplo, AutomaticTrans, <http://www.automatictrans.es>, o Language Weaver, <http://www.languageweaver.com>.

⁷ La plataforma OpenTrad, <http://www.opentrad.org>, está formada por dos arquitecturas de código abierto: Apertium y Matxin, <http://matxin.sourceforge.net/>; esta última utiliza un sistema de transferencia sintáctica más profundo que el de Apertium.

⁸ En un ordenador personal de sobremesa.

orientado a lenguas emparentadas (como catalán-español, portugués-español, checo-eslovaco, sueco-danés, bahasa indonesia-bahasa malasia, etc.), pero desde su versión 2, publicada en diciembre de 2006, el motor de traducción usa un sistema de transferencia más avanzado que permite manejar rasgos lingüísticos presentes en lenguas no tan emparentadas (como español-inglés).

Apertium es el resultado de diversos proyectos de subvención pública en los que han participado hasta el momento diferentes universidades (Universitat d'Àlacant, Universidade de Vigo, Universitat Politècnica de Catalunya, Euskal Herriko Unibertsitatea y Universitat Pompeu Fabra) y empresas (Eleka Ingeniaritza Linguistikoa, Imaxin Software, Elhuyar Fundazioa y Prompsit Language Engineering). La plataforma Apertium se basa en la experiencia y conocimientos adquiridos por el grupo Transducens⁹ de la Universitat d'Àlacant en el desarrollo del sistema catalán-español interNOSTRUM¹⁰ y del traductor portugués-español Tradutor Universia¹¹, que no son, sin embargo, de código abierto.

Apertium está escrito principalmente en C++ y puede compilarse y ejecutarse sobre el sistema operativo Linux,¹² aunque su adaptación a otros sistemas operativos no debería plantear, teóricamente, muchos problemas; en cualquier caso, el programa puede instalarse fácilmente en un servidor de aplicaciones de internet para ser accedido de forma remota. Hasta la fecha se han desarrollado datos lingüísticos funcionales para los siguientes pares de lenguas (en ambos sentidos de traducción): catalán-español, gallego-español, portugués-español, aranés-catalán, catalán-francés y catalán-inglés;¹³ además, los autores de este artículo tienen conocimiento de desarrollos iniciales para rumano-español y sueco-danés. Para algunos de los pares de lenguas mencionados, las tasas de error¹⁴ se sitúan entre el 5% y el 10% con textos periodísticos; estos resultados aún pueden mejorarse fácilmente aumentando la cobertura de los diccionarios o de las reglas de transferencia.

El motor de traducción de Apertium, las herramientas auxiliares, la documentación correspondiente y la mayoría de los datos lingüísticos desarrollados hasta la fecha para Apertium pueden descargarse desde la web del proyecto en <http://apertium.sourceforge.net>. Existe, además, una web donde pueden usarse en línea los distintos traductores automáticos disponibles para Apertium: <http://xixona.dlsi.ua.es/prototype/>.

A continuación se describe brevemente algunas de las características de Apertium; el lector interesado en más detalles puede consultar la documentación de la plataforma o alguno de los artículos publicados sobre ella [1].

⁹ <http://transducens.dlsi.ua.es>

¹⁰ <http://www.internostrum.com>

¹¹ <http://tradutor.universia.net>

¹² Para algunas de las distribuciones de Linux basadas en Debian existen paquetes precompilados presentes en los repositorios oficiales.

¹³ Algunos de estos datos se distribuyen con licencia GNU GPL y otros con licencia Creative Commons, <http://www.creativecommons.org>.

¹⁴ Calculadas como el porcentaje de palabras que deben ser añadidas, eliminadas o modificadas para que la traducción resultante sea correcta.

3.1. El motor de traducción

Los traductores que pueden desarrollarse con la plataforma Apertium son sistemas de traducción automática por transferencia consistentes en una serie de módulos [7] dispuestos en cascada (véase la figura 1):

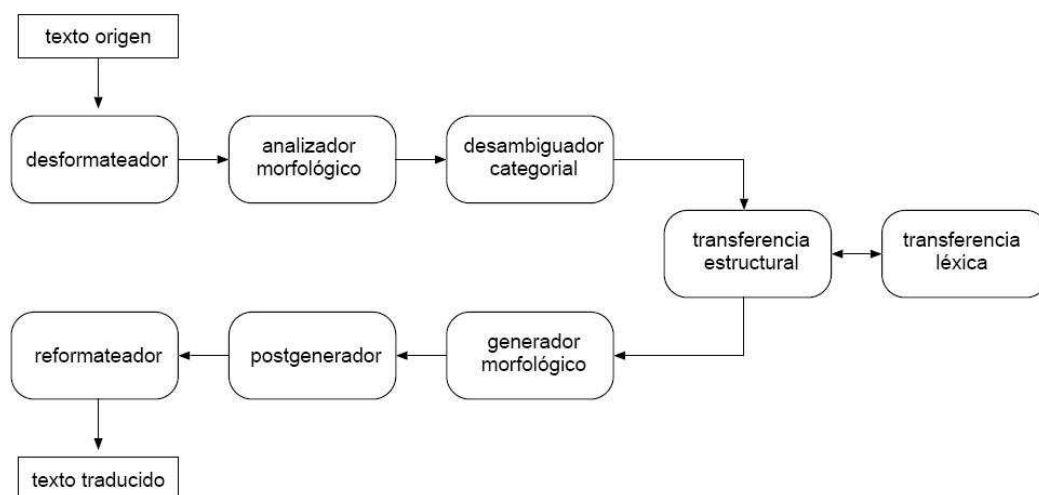


Figura 1. Los módulos de un sistema de traducción automática construido con la plataforma Apertium.

El **desformateador**, que separa el texto a traducir de la información de formato del documento (XML, HTML, etc.). La información de formato es *encapsulada* de manera que los módulos restantes la tratan como blancos entre palabras. Por ejemplo, ante el texto HTML en español:

```
es <em>una señal</em>
```

el desformateador encapsula las etiquetas HTML entre corchetes y proporciona como salida:

```
es [<em>]una señal[</em>]
```

Las secuencias de caracteres entre corchetes son tratadas por el resto de módulos como simples espacios en blanco entre palabras.

El **analizador morfológico**, que segmenta el texto en *formas superficiales* (las unidades léxicas tal como se presentan en los textos) y entrega para cada forma superficial una o más *formas léxicas* consistentes en un *lema* (la forma base comúnmente usada para las entradas de los diccionarios clásicos),

la *categoría léxica* (nombre, verbo, preposición, etc.) y la información de flexión morfológica (número, género, persona, tiempo, etc.). La división de un texto en formas superficiales presenta aspectos complejos debido a la existencia, por un lado, de contracciones (*del, teniéndolo, vámonos*) y, por otro, de unidades léxicas de más de una palabra (*a pesar de, echó de menos*). El analizador morfológico permite analizar estas formas superficiales complejas y tratarlas adecuadamente para que sean procesadas por los módulos posteriores. En el caso de las contracciones, el sistema lee una única forma superficial y da como salida una secuencia de dos o más formas léxicas (por ejemplo, la contracción *del* sería analizada como dos formas léxicas, una para la preposición *de* y otra para el artículo *el*). Las unidades léxicas de más de una palabra (*multipalabras*) son tratadas como formas léxicas individuales y, según su naturaleza, reciben un tratamiento específico. Al recibir como entrada el texto de ejemplo proveniente del módulo anterior, el analizador morfológico proporcionaría como salida:

```

^es/ser<vbser><pri><p3><sg>${ <em>}
^una/un<det><ind><f><sg>/unir<vblex><prs><1><sg>/unir
<vblex><prs><3><sg>${
^señal/señal<n><f><sg>${</em>}

```

donde cada forma superficial es analizada como una o más formas léxicas. Así, *es* es analizada como una forma superficial con lema *ser*, mientras que *una* recibe tres análisis: lema *un*, determinante indefinido femenino singular; lema *unir*, verbo en presente de subjuntivo, primera persona del singular, y lema *unir*, verbo en presente de subjuntivo, tercera persona del singular.

El desambiguador léxico categorial, elige, usando un modelo estadístico (*modelo oculto de Markov* [4]), uno de los análisis de una palabra ambigua de acuerdo con su contexto; en el ejemplo utilizado, la palabra ambigua sería la forma superficial *una*, que puede analizarse de tres maneras diferentes. Las formas superficiales ambiguas por tener más de un lema, más de una categoría o representar más de una flexión son muy comunes (en las lenguas románicas, en torno a una de cada tres palabras) y son una fuente muy importante de errores de traducción en caso de elegir el equivalente incorrecto. El modelo estadístico se entrena sobre corpus suficientemente representativos de textos en lengua origen, aunque también puede emplearse información de la lengua destino de la traducción durante el entrenamiento [2]. El resultado tras la desambiguación léxica categorial del texto de ejemplo proporcionado por el analizador morfológico sería:

```

^ser<vbser><pri><p3><sg>${ <em>}^un<det><ind><f><sg>${
^señal<n><f><sg>${</em>}

```

en el que se ha elegido la forma léxica correcta, determinante, para la palabra *una*.

El módulo de transferencia léxica, que gestiona un diccionario bilingüe y es invocado por el módulo de transferencia estructural, lee cada forma léxica

en lengua origen y entrega la forma léxica correspondiente en lengua destino. El diccionario contiene actualmente un único equivalente para cada forma léxica en la lengua destino, pero próximamente la plataforma permitirá indicar más de una (polisemia) y un nuevo módulo se encargará de elegir el equivalente adecuado. Las multipalabras son traducidas como una unidad. En el ejemplo utilizado, cada una de las formas léxicas se traduciría al catalán de la siguiente manera:

```
ser<vbser> → ser<vbser>
un<det> → un<det>
señal<n><f> → senyal<n><m>
```

El módulo de transferencia estructural, que detecta y trata patrones de palabras (sintagmas) que exigen un tratamiento especial por causa de las divergencias gramaticales entre las lenguas (cambios de género y número, reordenamientos, cambios preposicionales, etc.). Este módulo lee de un archivo las reglas que describen la acción que debe realizarse para cada patrón. Ante la frase de ejemplo, el patrón formado por `^un<det><ind><f><sg>$` `^señal<n><f><sg>$` sería detectado por una regla determinante–nombre, que en este caso modificaría el género del determinante para que concuerde con el nombre; el resultado sería:

```
^ser<vbser><pri><p3><sg>$[ <em>]^un<det><ind><m><sg>$
^senyal<n><m><sg>$[</em>]
```

El generador morfológico, que genera a partir de la forma léxica en lengua destino una forma superficial flexionada adecuadamente. El resultado para la frase de ejemplo sería:

```
és[ <em>]un senyal[</em>]
```

El postgenerador, que realiza algunas operaciones ortográficas en lengua destino tales como contracciones y apostrofaciones. En la frase de ejemplo utilizada no hay que realizar ninguna contracción ni apostrofación.

El reformateador, que reintegra la información de formato original al texto traducido; el resultado para la frase de ejemplo sería la correcta conversión del texto a formato HTML:

```
és <em>un senyal</em>
```

Como se ha podido observar, los módulos del sistema se comunican entre sí mediante flujos de texto, lo que permite su evaluación independiente; de hecho, algunos módulos pueden ser utilizados de forma aislada en otras aplicaciones de procesamiento del lenguaje natural.

3.2. Los datos lingüísticos

Los datos lingüísticos de Apertium están completamente desacoplados del motor de traducción y se codifican mediante formatos basados en XML¹⁵; esto permite su interoperabilidad (es decir, la posibilidad de utilizar los datos XML en entornos diferentes) y facilita la transformación y el mantenimiento. En gran medida, el éxito de un motor de traducción automática de código abierto depende de que otros colectivos acepten los formatos que utiliza; este es, de hecho, el mecanismo por el cual aparecen los estándares *de facto*. La aceptación puede verse facilitada por el uso de un formato interoperable basado en XML, así como por la disponibilidad de herramientas para gestionar los datos lingüísticos. También es cierto que, a su vez, la aceptación de los formatos depende considerablemente del éxito del sistema de traducción en sí. Los formatos XML para los diferentes tipos de datos lingüísticos de Apertium se definen mediante definiciones de tipo de documento (DTD) en XML diseñadas específicamente.

3.3. Los compiladores

La plataforma Apertium contiene *compiladores* que convierten los datos lingüísticos a la forma “binaria” eficiente que es usada por cada módulo. Los cuatro módulos de procesamiento léxico (el analizador morfológico, el módulo de transferencia léxica, el generador morfológico y el postgenerador) utilizan un único compilador, que genera una representación basada en *transductores de estados finitos* [5]. El módulo de transferencia estructural utiliza una representación de las reglas de transferencia en forma de *máquina de estados finitos*.

Estos compiladores no solo necesitan unos pocos segundos para compilar diccionarios con miles de entradas (lo que simplifica la labor de desarrollo, ya que el efecto en el sistema de un cambio en una regla de transferencia o en una entrada del diccionario se puede evaluar casi instantáneamente), sino que, además, las estructuras compiladas resultantes permiten que el motor pueda traducir a velocidades del orden de decenas de miles de palabras por segundo.

4. Modelos de negocio basados en Apertium

La plataforma Apertium, junto con la documentación y las herramientas de gestión de datos lingüísticos ya disponibles, ofrece buenas oportunidades para investigadores, traductores profesionales y empresas que trabajen en ingeniería lingüística u ofrezcan servicios lingüísticos.

Los usuarios individuales, los investigadores y las comunidades lingüísticas (al igual que las empresas privadas) pueden unir sus esfuerzos y contribuir al desarrollo de las tecnologías y los datos de Apertium; cualquier persona que tenga las habilidades informáticas y lingüísticas necesarias puede adaptar o ampliar Apertium para producir nuevos o mejores sistemas de traducción automática, incluso para nuevos pares lingüísticos.

¹⁵ <http://www.w3.org/XML/>

El desarrollo en código abierto de software y datos garantiza el acceso libre e ilimitado a los recursos creados. Además, no es necesario crear estos recursos desde cero, puesto que el punto de partida de nuevos proyectos puede arrancar donde se quedaron proyectos anteriores.

Como el código y los datos de Apertium pueden reutilizarse, los investigadores y desarrolladores pueden centrarse en mejorarlos. Nuestra experiencia nos dice que reutilizar datos de pares de lenguas existentes acelera el desarrollo de datos para nuevos pares; por ejemplo, las reglas para la concordancia de género y número son básicamente las mismas para todos los pares de lenguas emparentadas que nuestro equipo ha desarrollado. Además, los diccionarios monolingües pueden utilizarse en más de un par de lenguas con pequeñas modificaciones.

4.1. Servicios ofertables en torno a Apertium

El software de código abierto permite a las empresas trabajar con nuevos modelos de negocio. En concreto, una empresa podría ofrecer un amplio rango de servicios en torno a Apertium [6], tales como:

- instalación y mantenimiento de servidores de traducción automática basados en Apertium;
- mantenimiento y ampliación de los datos lingüísticos;
- adaptación de los datos lingüísticos a dominios particulares o a variedades dialectales;
- adaptación de los datos lingüísticos para reducir el esfuerzo de postedición en los documentos de un cliente particular;
- construcción de datos lingüísticos para nuevos pares de lenguas;
- integración de Apertium en sistemas de gestión documental multilingües;
- desarrollo de extensiones, nuevos módulos y herramientas auxiliares para Apertium;
- traducción corregida por traductores humanos especializados en la postedición de Apertium; el conocimiento del sistema de traducción automática permite reducir el tiempo necesario para realizar postediciones de calidad y esto puede permitir, en función de la calidad del traductor automático, que estas traducciones se ofrezcan a precios inferiores a los que ofrece el mercado para traducciones humanas “desde cero”.

Además, algunos de los servicios anteriores también pueden ser ofrecidos por traductores profesionales autónomos que conozcan con suficiente detalle el funcionamiento del traductor, lo que abre nuevas oportunidades de trabajo para este colectivo.

Las empresas que han participado en algunos de los proyectos en los que se ha desarrollado Apertium¹⁶ han comenzado a explotar las posibilidades de negocio que surgen en torno al traductor, y pese a que la primera versión estable

¹⁶ Eleka Ingeniaritza Linguistikoa, <http://www.eleka.net>, Imaxin Software, <http://www.imaxin.com>, Elhuyar Fundazioa, <http://www.elhuyar.org>, y Prompsit Language Engineering, <http://www.prompsit.com>.

de Apertium fue publicada a finales de 2005, ya existen casos reales de comercialización. Así, las ediciones en internet de dos de los principales periódicos de Galicia,¹⁷ publicados originalmente en español, cuentan desde hace meses con una versión diaria en gallego basada en la tecnología de Apertium. Algunos organismos públicos, por otro lado, han comenzado a estudiar la posibilidad de utilizar Apertium como plataforma de traducción automática de cara a ofrecer servicios de traducción de alta velocidad por internet.

Un argumento esgrimido con cierta frecuencia es que este mercado sería más rentable si los sistemas fueran de código cerrado; al mismo tiempo, es habitual que los potenciales clientes sean reticentes a tecnologías abiertas como Apertium. Sin embargo, pueden enumerarse ciertas ventajas sobre los sistemas cerrados; una de las más significativas es que los clientes de sistemas de traducción automática de software abierto no ven a la empresa proveedora como una empresa de la que *dependen* tecnológicamente, sino, más bien, como *socios* tecnológicos, ya que la puerta siempre está abierta para, llegado el caso, contratar los servicios con otra empresa que los ofrezca. Si tanto los programas como los datos son de código abierto, esto no solo es posible, sino que genera una interesante competencia.

Además, para instituciones, entidades públicas y grandes empresas es incluso más interesante la contribución social que pueden ejercer al hacer modificaciones abiertas, mejorar los datos lingüísticos o añadir nuevas funcionalidades al software de código abierto desarrollado específicamente para ellas.

Esto les da una imagen muy positiva ante sus clientes y usuarios, puesto que no solo ofrecen un mejor servicio, sino que benefician a toda la comunidad.

5. Modelos de investigación basados en Apertium

El software de código abierto garantiza la reproducibilidad de los experimentos de investigación. Cuando se publican resultados de evaluación de un sistema de traducción automática al que se han incorporado nuevas técnicas, si los experimentos realizados no pueden ser reproducidos, los autores obligan a los lectores a confiar en sus afirmaciones. Si el software y los datos lingüísticos son de código abierto es posible reproducir los experimentos fácilmente, realizar nuevos experimentos y compararlos entre sí. La *reproducibilidad* es muy importante en la ciencia: es necesaria para poder comprobar o verificar los resultados experimentales y constituye una de las bases del progreso científico, ya que si un experimento concreto no puede repetirse, no se considera que proporcione una *prueba científica* sólida.

El software de código abierto fomenta la interacción colaborativa entre grupos de investigación que trabajen en la misma área. Tanto las administraciones públicas como las universidades deberían apoyar y financiar este tipo de trabajo colaborativo, que tiene un beneficio especial para grupos de investigación con pocos recursos. La perspectiva del código abierto estimula y facilita la presen-

¹⁷ La Voz de Galicia, <http://www.lavozdeg Galicia.es>, y El Correo Gallego, <http://www.elcorreogallego.es>.

cia de grupos de investigación en programas locales, nacionales y europeos de investigación, desarrollo e innovación tecnológica.

El código abierto también puede facilitar la transferencia de nuevos avances de la comunidad investigadora a empresas interesadas en su aplicación. Este papel que pueden jugar los productos de código abierto debería ser tenido en cuenta por los organismos públicos que promocionan la cooperación entre investigadores y empresas en su entorno socioeconómico.

El software de código abierto plantea retos comerciales a los investigadores que trabajen en nuevos métodos y técnicas. De hecho, el número de *spin-offs* de base tecnológica (nos referimos aquí a empresas creadas por investigadores como resultado de una actividad investigadora concreta) ha aumentado en los últimos años. Estas empresas no sólo tienen un producto a ofrecer y un catálogo de servicios relacionados, sino también el *know-how* desarrollado durante el trabajo investigador de sus miembros; además, al estar situados en un punto intermedio, pueden ofrecer mejores servicios en colaboración con universidades y empresas. Este es, por ejemplo, el caso de Prompsit Language Engineering,¹⁸ una empresa *spin-off* creada por investigadores del grupo Transducens de la Universitat d'Alacant, que ofrece servicios alrededor de la plataforma Apertium.

6. Conclusiones

El software libre plantea nuevos retos tanto para las empresas como para los centros de investigación. Aunque la filosofía del código abierto acaba de aterrizar en el campo de la traducción automática, un campo concentrado hasta ahora exclusivamente en productos comerciales, ya comienza a vislumbrarse el enorme potencial de las tecnologías abiertas en el terreno de la traducción. Es posible, por tanto, un nuevo modelo de negocio orientado a la oferta y demanda de servicios en torno a los motores de traducción y los datos lingüísticos, más que a los programas y datos en sí mismos. Además, con el desarrollo de lo que se ha dado en llamar la web 2.0, aparecerán nuevas aplicaciones en internet que establecerán redes sociales para el desarrollo colaborativo de mejores sistemas de traducción automática que funcionen sobre sistemas abiertos. En este artículo hemos presentado Apertium, una plataforma completamente funcional de traducción automática desarrollada en la Universitat d'Alacant, que es a día de hoy uno de los principales exponentes de código abierto dentro del área.

Agradecimientos

Este trabajo ha sido parcialmente subvencionado por el Ministerio de Industria, Comercio y Turismo a través de los proyectos FIT-340101-2004-3, FIT-340001-2005-2 y FIT-350401-2006-5, por el Ministerio de Educación y Ciencia a través de los proyectos TIC2003-08681-C02-01 y TIN2006-15071-C03-01, y por

¹⁸ <http://www.prompsit.com>

la Generalitat de Catalunya a través del proyecto DURSII-05I. Felipe Sánchez-Martínez disfruta de la ayuda para la formación de personal investigador BES-2004-4711, financiada por el Fondo Social Europeo y el Ministerio de Educación y Ciencia.

Referencias

1. Armentano-Oller, C., Carrasco, R. C., Corbí-Bellot, A. M., Forcada, M. L., Ginestí-Rosell, M., Ortiz-Rojas, S., Pérez-Ortiz, J. A., Ramírez-Sánchez, G., Sánchez-Martínez, F., Scalco, M. A.: Open-source Portuguese-Spanish machine translation. En: Computational Processing of the Portuguese Language: 7th Workshop on Computational Processing of Written and Spoken Portuguese, PROPOR 2006. Lecture Notes in Artificial Intelligence 3960. Springer-Verlag (2006) 50–59
2. Armentano-Oller, C., Forcada, M. L.: Open-source machine translation between small languages: Catalan and Aranese Occitan. En: Strategies for developing machine translation for minority languages: 5th SALT MIL workshop on Minority Languages (2006) 51–54
3. Arnold, D.: Why machine translation is difficult for computers. In Somers, H. L. (coordinador): Computers and Translation: a translator's guide. John Benjamins, Amsterdam (2003) 119–142.
4. Cutting, D., Kupiec, J., Pealersen, J., Sibun, P.: A practical part-of-speech tagger. En: Proceedings of the Third Conference on Applied Natural Language Processing (1992) 133–140.
5. Garrido, A., Iturraspe, A., Montserrat, S., Pastor, H., Forcada, M. L.: A compiler for morphological analysers and generators based on finite-state transducers. En: Procesamiento del Lenguaje Natural, 25 (1999) 93–98.
6. Ramírez-Sánchez, G., Sánchez-Martínez, F., Ortiz-Rojas, S., Pérez-Ortiz, J. A., Forcada, M. L.: Opentrad Apertium open-source machine translation system: an opportunity for business and research. En: Proceedings of the Twenty-Eighth International Conference on Translating and the Computer (2006)
7. Ginestí Rosell, M. (coordinadora): Documentación del sistema de código abierto Opentrad Apertium de traducción automática de transferencia sintáctica superficial. <http://apertium.svn.sourceforge.net/viewvc/apertium/> (febrero 2007).
8. Sánchez-Martínez, F., Pérez-Ortiz, J. A., Forcada, M. L.: Speeding up target language driven part-of-speech tagger training for machine translation. En: Proceedings of the 5th Mexican International Conference on Artificial Intelligence, MICAI 2006. Lecture Notes in Artificial Intelligence 4293. Springer-Verlag (2006) 844–854

REST: Representational State Transfer, en Ruby on Rails

Carlos Alberto Paramio Danta

Evolve Studio
carlos@evolve.st
<http://www.evolve.st/>

Resumen Roy Fielding, uno de los principales autores de la especificación del protocolo HTTP, publicó en el año 2000 un trabajo de doctorado [1] donde proponía un estilo a seguir en la arquitectura de sistemas software de hipermedios. Recientemente, las últimas revisiones del entorno de programación Ruby on Rails [2] ha incorporado un conjunto de macros para definir rutas que siguen este mismo patrón de diseño, lo que sumado a otras características del entorno proporciona una vía realmente sencilla para implementar interfaces que operan sobre los recursos de nuestra aplicación web; siendo este método mucho más sencillo que la utilización de RPC y tecnologías similares para la interconexión de aplicaciones.

1. REST: Representational State Transfer

Los principios que definen una arquitectura REST son los siguientes:

- El estado de la aplicación y la funcionalidad están divididos en lo que llamaremos *recursos*.
- Todos los recursos son accesibles de forma única mediante una sintaxis común en forma de enlaces a hipermedios.
- Todos los recursos comparten una interfaz común para la transferencia de estado entre el cliente y el recurso, consistente en un conjunto estricto de operaciones bien definidas.
- Un protocolo cliente/servidor que no guarde el estado, cuyos resultados puedan almacenarse en una memoria caché, y que esté claramente separado en capas.

Las ventajas prácticas de seguir estos principios son, al menos dos:

- Los encargados de poner la aplicación en explotación saben con certeza cuándo y dónde pueden guardar en una memoria caché las respuestas a ciertas peticiones, lo que incrementa el rendimiento disminuyendo notablemente la latencia.
- Las interfaces para acceder a los recursos que maneja la aplicación están bien definidas, lo que facilita la escritura y el mantenimiento de aplicaciones.

La nomenclatura típicamente usada en una aplicación REST es muy clara. En general, se utilizan sustantivos para nombrar a cada uno de los recursos, y un conjunto de verbos para definir las operaciones con dichos recursos. En el caso de las aplicaciones web, por ejemplo, los recursos son URLs (*Uniform Resources Locators*, localizadores uniformes de recursos), y los verbos usados son los distintos métodos que se pueden utilizar en las cabeceras de una petición mediante el protocolo HTTP, esto es: GET, POST, PUT y DELETE.

A modo de ejemplo, supongamos que estamos desarrollando una aplicación web para la gestión de un foro de conversación. Dicha aplicación debe almacenar cada uno de los mensajes publicados en el foro. Cada uno de los mensajes es un recurso, así como la colección de mensajes del foro que constituye otro recurso. Si queremos obtener un listado de todos los mensajes del foro, usaremos el método HTTP GET para solicitar el recurso situado en, digamos, la ruta */messages*. Si lo que deseamos es obtener el mensaje con el valor 1 en el identificador que hace las veces de clave primaria, podríamos realizar de nuevo una petición GET, esta vez con la ruta */message/1*. Para agregar un nuevo mensaje a nuestra colección, usaremos el método POST con la URL */messages*, que como hemos visto es la que representa a nuestro recurso "colección de mensajes". Para actualizar un mensaje, usaremos el método PUT con la URL del recurso correspondiente a ese mensaje, por ejemplo */message/1*. Y para eliminarlo, el método DELETE con la misma URL. En general, como hemos visto, tenemos un conjunto definido para los nombres de recursos (*/messages*, */message/1...*), y un conjunto cerrado de acciones (GET, POST, PUT, DELETE) para operar con ellos, el cual como veremos puede ser extendido para usar acciones algo más complejas.

Como se puede observar, el aspecto de los recursos definidos mediante un interfaz REST es el de una estructura jerárquica de directorios y archivos, donde el identificador del recurso colocado en la última parte de la URL hace el papel de archivo que contiene la información deseada, y el resto de la dirección parece un grupo de directorios anidados perfectamente lógicos en cuyo interior está almacenado el recurso.

El uso de este diseño para una aplicación fuerza a pensar en un patrón CRUD (*Create, Recover, Update and Delete*), es decir, en un conjunto de cuatro acciones básicas para cada recurso. Estas acciones se corresponden, respectivamente, con los métodos HTTP que se han nombrado anteriormente: POST, GET, PUT y DELETE. También existe este mismo patrón de diseño en el motor de base de datos operado mediante SQL, con los comandos INSERT, SELECT, UPDATE y DELETE. Y por supuesto, en la aplicación se corresponderán con cuatro métodos que ejecutarán las acciones pertinentes para alterar el estado de la misma. Así que se puede observar una relación directa entre el uso de un API REST y la utilización del patrón de diseño CRUD para cada uno de nuestros recursos. De hecho, CRUD cobra una gran importancia cuando tratamos de implementar una aplicación de este tipo, y salvo excepciones, la inmensa mayoría de los dominios de la aplicación pueden describirse mediante la manipulación de recursos usando estas cuatro operaciones básicas.

2. REST en Ruby on Rails

A partir de Ruby on Rails 1.2, el desarrollador dispone de un conjunto de macros para definir rutas de acceso a los diferentes recursos. Así, los recursos se corresponden con instancias de algunos de los modelos definidos en nuestra aplicación, o a una colección de ellos, y las mencionadas macros ponen a nuestra disposición métodos que devuelven la URL de recurso correspondiente. En la figura 1 se muestra un ejemplo del conjunto de métodos que Rails creará para acceder a los recursos correspondientes al modelo *Message*. Para que esto suceda, es necesario añadir una llamada a la macro *resources* en el archivo de configuración de rutas *routes.rb*:

```
map.resources :messages
```

Helpers	Métodos HTTP	Rutas	Acciones
messages_path	GET	/messages	index
message_path(id)	GET	/messages/1	show
new_message_path	GET	/messages/new	new
messages_path	POST	/messages	create
edit_message_path	GET	/messages/1;edit	edit
message_path(id)	PUT	/messages/1	update
message_path(id)	DELETE	/messages/1	destroy

Figura 1. Métodos definidos por la macro *resources* para *messages*

Cuando usemos las URL a los recursos, Rails comprobará el método HTTP que ha sido usado en la petición, y en función de éste determinará qué acción del controlador del mismo nombre que el recurso (en nuestro ejemplo, *messages*) deberá ejecutar. En la tabla mencionada anteriormente puede observarse la correspondencia entre las URL y las acciones, según el método usado. Construir el resto es bastante sencillo: Simplemente se deben implementar las distintas acciones del controlador, sabiendo cual es el comportamiento que éstas deben tener, a saber:

- *index* devuelve una colección con los recursos.
- *create* genera un nuevo recurso a partir de los datos obtenidos en una petición POST, y lo añade a la colección.
- *new* crea un nuevo recurso, sin guardarlo en el estado de la aplicación, y lo devuelve a la interfaz del cliente para que puedan rellenarse los datos correspondientes y enviarlo de nuevo a la aplicación.
- *show* devuelve el recurso especificado mediante el identificador guardado en `params[:id]`.

- *update* actualiza el recurso especificado mediante el identificador guardado en `params[:id]` con los datos que provienen de la petición.
- *edit* devuelve el recurso especificado mediante el identificador guardado en `params[:id]` a la interfaz del cliente para que pueda editarse su contenido.
- *destroy* elimina el recurso especificado mediante el identificador guardado en `params[:id]`.

El lenguaje HTML, sin embargo, no está preparado para enviar peticiones usando los métodos PUT o DELETE. Es por eso que los navegadores web no lo soportan. La solución adoptada en Rail consiste en incorporar un nuevo parámetro al formulario, como un campo oculto, llamado *__method*, donde se indica el método HTTP que queremos usar en la petición. Así, el programa cliente envía una petición POST, pero Rails la interpreta adecuadamente.

Rails 1.2 proporciona además un generador de código adicional que nos permite crear el esqueleto descrito en las líneas anteriores, denominado *scaffold_resource*. Pasándole un nombre de modelo como parámetro, éste creará un controlador con el nombre en plural de nuestro modelo, conteniendo las acciones descritas anteriormente junto a una implementación estándar, así como un conjunto de vistas básicas para moverse entre las mismas. Podemos crear un modelo básico del recurso *message*, y usar dicho generador para obtener una interfaz REST para el mismo, todo en una sola línea y en cuestión de segundos:

```
script/generate scaffold_resource Message \
  title:string body:text author:string created_at:datetime
```

Existe un plugin muy útil para visualizar en un mismo listado las direcciones a recursos definidas por nuestra aplicación Rails. Este plugin se denomina *routing_navigator*, y puede instalarse en la aplicación mediante el comando:

```
script/plugin install \
  http://svn.techno-weenie.net/projects/plugins/routing_navigator
```

Una vez instalado, sólo deberemos acceder a la URL */routing_navigator* para observar dicho listado.

2.1. Anidación de recursos

Adicionalmente, los recursos y sus URL correspondientes se pueden anidar. Así, siguiendo con el ejemplo anterior, podríamos querer añadir soporte para múltiples foros a nuestra aplicación, de manera que los mensajes deberán pertenecer a uno de dichos foros. Para definir los recursos relacionados con el nombre *forums*, crearemos el modelo y el controlador correspondiente, y agregaremos la macro *resources* al fichero de configuración de rutas, tal y como hicimos con *messages* anteriormente. Pero dado que ahora un recurso *message* pertenece a un recurso *forum*, agregaremos un bloque que contenga la declaración de este último recurso. Así, Rails construirá una URL al recurso *message* anidándola siempre al primero, de manera que para acceder, por ejemplo, a los mensajes del foro 1, utilizaremos una URL del tipo: */forum/1/messages*:

```
map.resources :forums do |forum|
  forum.resources :messages
end
```

2.2. Acciones propias adicionales

Las acciones CRUD pueden ser insuficientes a la hora de implementar las acciones necesarias con los recursos. Así que Rails nos proporciona una manera sencilla de definir acciones propias que operen sobre un único elemento o sobre una colección de ellos. Por ejemplo, si queremos definir una acción *latests* que devuelva solamente los últimos mensajes enviados al foro, podríamos incluirla en la lista de métodos de acceso al recurso en el fichero de configuración de rutas de la siguiente manera:

```
map.resources :forums do |forum|
  forum.resources :messages, :collection => { :latests => :get }
end
```

Se observa que debemos especificar que se trata de la dirección de un recurso para obtener una colección de elementos, de nombre *latests*, y que responderá al método HTTP GET. Rails definirá un helper denominado *latests_messages_url* que tomará como parámetro el identificador del recurso padre (por ejemplo, foro número 1) y nos devolverá la dirección del recurso (según el ejemplo, */forum/1/messages/latests*). Como puede verse, la nueva acción será añadida a la dirección del recurso con la colección de todos los elementos, y separada de ésta mediante un símbolo de punto y coma.

De igual manera, podemos definir una ruta para una acción que opere sobre un único miembro. Por ejemplo, una acción *sticky* que complemente el valor del atributo "pegajoso" de un mensaje, el cual determina si éste debe aparecer al principio de la lista de mensajes. Dado que se trata de una acción que variará el estado de la aplicación, deberá ser llamada mediante el método HTTP PUT indicando la dirección del recurso que se desea modificar:

```
map.resources :forums do |forum|
  forum.resources :messages, :member => { :sticky => :put }
end
```

3. Formato de las respuestas

Una funcionalidad adicional de Rails cobra si cabe mayor importancia cuando la utilizamos en una aplicación con interfaz REST. Se trata de la posibilidad de aceptar y devolver valores en formatos distintos al habitual HTML usado comúnmente en las vistas de la aplicación. Para ello, disponemos de un método denominado *respond_to* que permite establecer cuáles son los formatos aceptados por la misma, y qué devolver en cada caso. Cuando realicemos una petición a la aplicación, se observarán las cabeceras HTTP para comprobar qué tipos

acepta la aplicación cliente como respuesta, y se devolverá una acorde a las posibilidades de la misma. Por ejemplo, podemos hacer que la acción *index* del controlador *messages* pueda devolver la lista de mensajes en un archivo XML, en lugar de usar las vistas HTML para visualizar con un navegador. El siguiente bloque de código situado al final de la implementación de la acción sería una buena aproximación:

```
respond_to do |format|
  format.html # render index.rhtml
  format.xml { render :xml => @messages.to_xml }
```

Lo mismo podría hacerse para otros tipos MIME, definiendo respuestas para .text, .js, .ics, .rss, .atom, .yaml, etc. Existen algunos ya existentes, pero podemos definir los nuestros propios, y que la aplicación compruebe el agente del usuario usado para decidir en qué formato debe darse la respuesta.

La autodetección del formato no es siempre efectiva, así que otra posibilidad es asignar una extensión relacionada directamente con el formato de respuesta deseado. De esta forma, además, se pueden combinar formatos: Se puede solicitar una respuesta desde un formulario en HTML, y que la misma nos sea devuelta en XML.

4. Conclusiones

Como se ha podido comprobar, la implementación de una interfaz REST en Rails es verdaderamente sencilla, y nos permite homogeneizar el acceso a nuestra aplicación, proporcionando como añadido una vía fácil para la intercomunicación con otras aplicaciones. Esto se está utilizando como base para implementaciones similares a SOAP y otros protocolos para web services, pero mucho más sencillas de utilizar, como es el caso de ActiveResource, donde el desarrollador Rails describe clases externas que se comportarán de manera similar a los modelos definidos como ActiveRecord (sólo que los recursos están situados en una aplicación remota).

Además, REST obliga al desarrollador a pensar en sus aplicaciones según un patrón de diseño homogéneo y sencillo, si se siguen las convenciones impuestas por el framework, obteniendo como beneficio un código más legible, un API preparado para servicios web, y el evitar la repetición de código entre controladores que manejan recursos asociados.

5. Referencias

Referencias

1. Representational State Transfer. <http://www.ics.uci.edu/~fielding/pubs/dissertation/>.
2. Ruby on Rails Framework. <http://www.rubyonrails.org>.

Babel: A collaborative language learning system

Javier Albusac, Carlos Gonzalez-Morcillo Luis Jimenez-Linares

Oreto Research Group, University of Castilla-La Mancha (Spain)

{JavierAlonso.Albusac, Carlos.Gonzalez,
Luis.Jimenez}@uclm.es

<http://oreto.inf-cr.uclm.es/>

Resumen Nowadays, plenty of people from very different countries are involved in the Free Software community. Therefore, communication among the individuals involves a big deal that we should take care of. Currently, Most of the existing language learning applications on the Internet are e-Learning tools which do not allow users to help and to share their knowledge with each other. These are a sort of applications where the teacher is the only one who produces information and users just can consume it. Babel is a free web-based content management system which provides users with synchronous and asynchronous communication tools and with the benefits of learning in decentralized communities of users. Our proposal is a common framework to learn and to practice different languages which may be used and adapted in any educational institution as the system is distributed under the GNU/GPL license.

Palabras clave: Free software; Language learning; Content management; Decentralized communities

1. Introduction

Over the last ten years, the use of new technologies and Internet have broken the communication barriers among people from different countries. This fact has facilitated the appearance of virtual communities with common objectives (e.g Free Software Community). If the goal is to learn, we talk about learning communities.

The fast-growing online learning communities is owed to the increasing use of cooperative distributed systems. In the case of learning communities, users collaborate in the knowledge construction process together with the adaptations and the corrections made by other members of the community. In this approach [4], the responsibility over the contents and the learning of the community is distributed among its members, so the distribution of knowledge and the growth of the database are reinforced. On the other hand, the interaction among the community members is carried out by means of synchronous mechanisms (videoconferences, voice IP, chats) and asynchronous mechanisms (forums, mail, news).

Babel has been developed to offer a common framework to learn and practice different languages [5], which may be used and adapted in any educational institution as the system is distributed under the GNU/GPL license. All the related software is available at <http://raro.inf-cr.uclm.es/apps/babel/>

This paper proceeds as follows. Section 1 presents an introduction and some traditional teaching languages. Section 3 presents the objectives in the Babel design. In section 4, we present the architecture of Babel, where we emphasize the importance related to the behaviour of the State Machine, the communication tools, how the system searches the information, how the system use the dictionaries and a market research which justifies the chosen technologies. Section 5 presents how traditional teaching methods can be applied into the system. Finally, the paper concludes with the conclusions and directions for future research in this area.

1.1. Teaching languages

Flexibility on web systems(based on hypertext), in their non-sequential organization, allows us to improve the traditional methods of teaching, by adapting them in a more comfortable way to the natural mechanism of relationship among concepts. According to Sanchez Perez [2] and Richards [1], there are different models whose election will depend on the interest, the objectives or the learning environment of each person. Examples of these models are commonly used: *Grammar-Translation Method*, in which the aim is to reach the student ends up writing the language which he/she wants to learn. For this reason, the student should study the grammar and the vocabulary in detail. These activities are completed with many practical exercises. On the *Direct Method*, students learn a language through the oral-practical and reading exercises, reaching a great fluent linguistic. Unlike the method mentioned before, grammar is not studied at first place. The direct method is used by people who learn the native language at an early age. The *Audio-Oral and Audio-Visual Method* are based on the use of multimedia resources to ensure the knowledge by means of the continuing repetition of sounds and the association of images. Finally, the *Notional-Functional Method* structures the support material for the learning of a language according to the necessities of each people. For example, a computer specialist who wants to learn English to chat with other computer specialists from other countries, will study the vocabulary, technical terms, and expressions made in that language and related to the computer field.

1.2. Related works

To design Babel we analyzed the services offered by several web systems for the learning of languages. The following table resumes this analysis.

	MI	AF	H	AD	EAI	WS	LET	SS	LS	UE	ET	CA	Babel
Several Languages		✓	✓			✓							✓
Practical exercises	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Theoretical documents	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Automatic correction			✓	✓				✓	✓	✓	✓	✓	✓
User collaboration													✓
Internal tools			✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Games							✓						✓
Pay per use			✓	✓	✓			✓	✓	✓	✓	✓	

[MI] mansioningles.com

[AF] aulafacil.com

[H] home.es

[AD] auladiez.com

[EAI] e-aprender-ingles.com

[WP] worldspeaking.com

[LET] learn-english-today.com

[SS] speakspeak.com

[LS] learnspanish.com

[UE] usingenglish.com

[ET] englishtown.com

[CA] curso-de-aleman.de

After this study, some conclusions are observed. First, most of these systems are focused in the learning of only one language. Another important fact is all systems do not allow the users to contribute with their knowledge (there is no collaboration possibility). This is the case of the paradigm of the class imparted by the teacher in which the students are only passive receivers of information [3]).

2. Objectives

All analyzed systems in section 1.2 do not allow the users to collaborate in the growth of the base of contents. The users can only work with the contents exposed by the web site administrators. On the other hand, all systems always behave in the same way. The adaptation of these systems on an educational center can be complicated due to the difference of objectives and norms among different centers. Bearing in mind the results of section 1.2, we outline a series of initial objectives for the design and development of Babel.

1. *Distributed content management.* The system has a common content-base in which the users may create, update, and reuse several kinds of contents. The information stored in the system will flow through different stages like acquisition, supervision, and finally, publication. The workflows allow more control over the information of the system.
2. *Dynamic behaviour of the system.* The system behaves according to the configuration of a state machine. This behaviour allows the system to be installed in several scenarios with different objectives and necessities.
3. *Concurrency control.* The concurrency control is necessary in systems in which a large community of users update the contents at the same time. In Babel, several access control mechanisms are used to avoid inconsistencies on the stored information.

4. *Search contents.* On systems in which the base of contents reaches a great size, achieving a fast access to the data is complicated, although the data are properly classified according to some criteria. The use of a search engine facilitates the information access to users by employing a set of keywords.
5. *Dictionaries and voice synthesis.* The system works with a dictionary for each language registered in the system. To facilitate the work (creation and update of the dictionary contents), standard formats are used.
6. *Student motivation.* The system contents not only are ordered according to the language below, but also it is ordered according difficulty level. On the other hand, the user has several points associated which determined the user's level on the system. The student can promote to superior levels if he/she completes practical exercises or exams proposed in the system. Playing the games or helping to other users is other way.
7. *Multilanguage Interface.* Because the system will be used by people from different countries to learn languages, the interface of the application is logical to be made in different languages. The system is able to add new languages in a simple way.
8. *Communication among users.* The System has different chat rooms with suitable communication tools for many users from different countries. The user may practice talking and improving his/her handwriting. All chat rooms are achieved according to native language and target language.
9. *Usability.* The application of different usability techniques [7] so that the system is the most intuitive possible and simple of using. The user should forget the tool and should center his/her efforts in learning. Due to the use of templates and global sheets of style, the interface consistency is also guaranteed; the tools are accessible from any place of the web and always in the same way.
10. *Multiplatform.* The device (PC,PDA ...) or Operating System (GNU/linux, Microsof Windows, Solaris ...) used is not a problem to access to the system because only a browser is necessary. The resolution of screen is not a problem either, given that CSS style is used to determine the width and height proportional to the page.
11. *Free Software* Babel is a system in wich all registered users can create new contents or to modify the existing contents. The created contents can be used freely by any user of the system. The system may also be used and adapted in any educational institution as the system is distributed under the GNU/GPL license.

3. Architecture

In large information web systems, the architecture is usual to be divided into several layers. This multi-layer programming guarantees maximum clarity, easy debugging of errors, and code reuse. Specifically, the design of Babel has been divided in three layers: Presentation, Logic and Persistence. *The Presentation layer* involves how the data are presented. A great advantage is that the same

data can be shown in different modes (e.g many devices with different size of the screen), without modifying the behaviour of the system. *Logic layer* involves the behaviour of the system, in other words, the functionality of the system is implemented here. This layer processes and generates data and itself communicates with the Presentation layer to show the data. Finally, the *Persistence layer* is responsible for interacting with the database manager and storing the data on files. This layer responses to the logic layer requests and stores the data produced by the latter. The three layer are independent; a modification in a layer does not imply a modification in another layer. For example, if we decide to change the database manager, then the *Persistent layer* has to be modified, but the rest of layers remain intact. The architecture of Babel is illustrated in figure 1:

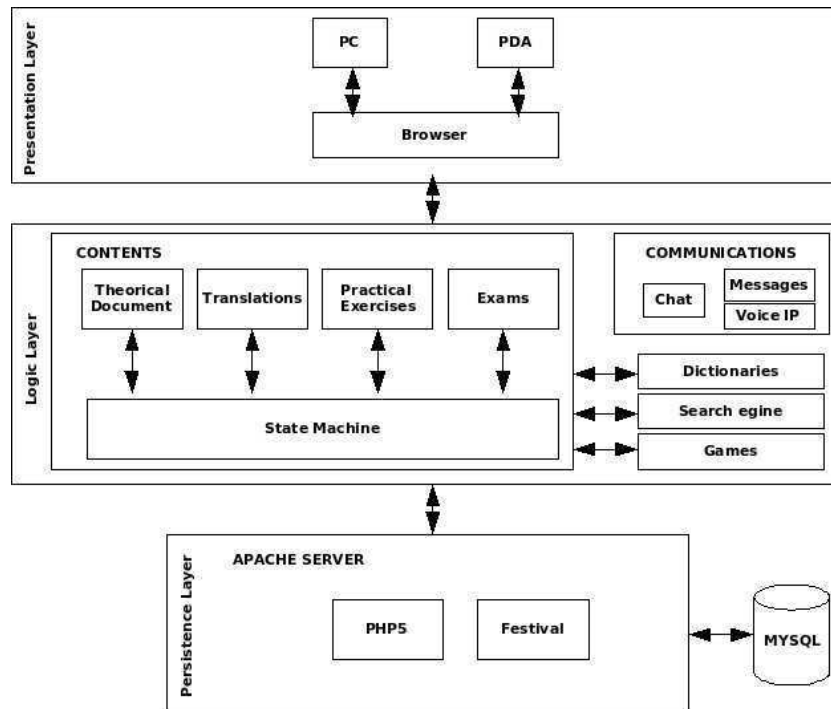


Figure 1. Architecture of Babel

Although the general architecture of Babel is generic and independent of the content types, the current installation (as platform for the learning languages) makes the system has the several types of contents: theoretical documents, exercises of translation, practical exercises, and exams. The system is designed to add new contents or new categories easily.

3.1. State Machine

The contents of the system are managed according to the configuration of the state machine, obtaining a great configuration flexibility. The elements which form the state machine can be modified totally. Next, it is shown the elements belong to the state machine.

- *User Roles.* Systems in which to have the information under the biggest control possible, user's roles are necessary. The system of roles allows us to define different types of user groups. The actions which an user can carry out in the system are determined by the roles and the permissions.
- *States.* A state define the situation that a content is it. All the documents of the system are always in a certain state. When an user makes an action on the document, the latter can change state.
- *Actions.* The actions that an user can carry out on any content type are defined in the configuration of the state machine. It is possible to create new actions, modify them or delete those that already exist.
- *State Transitions (state-action-state).* A transition indicates a state change. In Babel, an user can undertake an action on a content (whenever the role system allows him to do it). This content is in a certain state, the action carried out on the document can provoke the content has a new state different to the previous one. When a content changes its state, the actions that can be carried out on it, can vary.
- *Permissions.* There are two types of permission in Babel: Role permissions and user permissions. Role permissions indicate the actions an user can carry out with his/her role, and user permissions indicate which actions an user can make. An user that has permission to make an action can make it although its role does not allow him/her to do it.

The figure 2 samples the default configuration when the system is introduced for the first time. The control on the contents is critical in distributed and collaborative environments. The system administrator can configure the state machine, determining the workflow among the users of the system according to theirs roles. The number of states determines in great measure the control on the information and the number of workflows. A high number of states implies several things: a bigger control on the information, a slow process until the publication and a bigger number of workflows among users. On the contrary, a low number of states implies another questions: a quick publication, few workflows among users, and scarce control on the information.

Note that the adopted solution configures an intermediate option. This solution allows us to have control on the information and it avoids the time of publication of the contents to be too high. Next, we show the values that the elements of the state machine take initially. As we have mentioned previously, these values can be modified by the system administrator according to the necessities of the place where it is introduced.

- *Roles:* anonymous, editor, reviewer, administrator, manager.

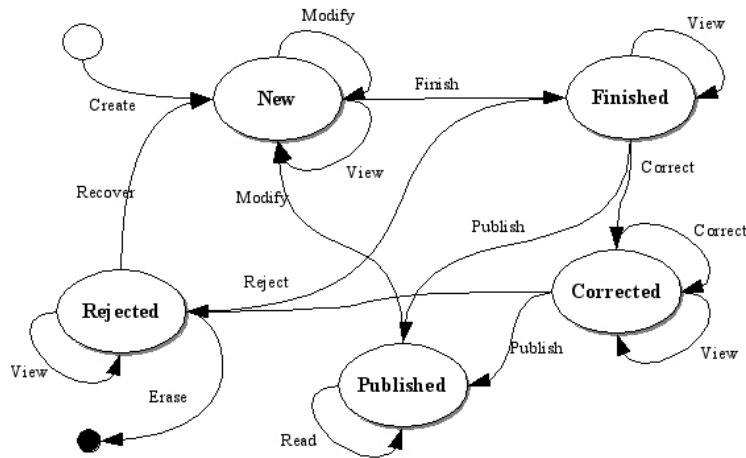


Figura 2. Initial state machine

- *Actions:* create, read, modify, view, finish, correct, publish, recover, reject, erase.
- *states:* new, finished, corrected, published, rejected.
- *state transitions:* the state transitions are exposed in the figure 2

When we create new actions in the system, it is necessary that we develop new source code, only in this case.

3.2. Communication tools

The asynchronous private messaging is one of the communication mechanisms to be employed by the user. Each user registered in the system has his own mailbox, which works in a similar way to the electronic mail. The users can also communicate among themselves in a synchronous way or real time by making use of the chat. There is a chat room for each pair of languages; the room “Spanish-English” will be formed by people who speak native Spanish or native English and who want to learn the other language.

The chat implemented in Babel has a series of features that make it specially interesting. It is possible to use dictionaries and to carry out corrections on the text written by another users. This way, the discussion in real time and the practice of the language with native users are facilitated.

Each user registered can specify its Skype account (any voice ip application can be used as long as this application has a web protocol with which an user can be called by other user from a web site). When an user enters in a chat room, an icon appears and any user can call it. Thus, the users can practice oral conversation.



Figura 3. Conversation written with corrections

3.3. Search of information

When coexisting different languages in a system, the management of a high number of dictionaries is necessary. Another added complexity is the combination of these languages (Spanish-English, English-German, English French ...) because each person needs the appropriate dictionary.

Babel is prepared for the direct importing of dictionaries in flat text format. These dictionaries are used in many tools for the search of terms. From the administration interface, new dictionaries can be added. Then, the base of terms is expanded. A more complete approach, based on model entity-relationship for the vocabulary learning can be seen in [6].

The search of contents allows the user to access to the information stored in the system starting from a group of keywords. The babel search engine allows the user to search on any content type. A previous processing of the stored contents is made with the purpose of extracting indexes and storing them in a data structure. The words which appear in a document are stored in the data structure together with the codes of the documents in which this word appears (each content in the system has a code that identifies). Therefore, the search is very fast because when we search a word, immediately the system recovers all the codes of the documents in which the word appears. The indexing of the contents, the same as in multitude of search engines, is made in schedules of low system activity.

3.4. Technologies used

We have used several technologies to develop babel: PHP5 like programming language, Apache like server of applications web, MySQL like database manager system and Festival like the voice synthesizer. Previously to the development of the system, we made a market research which has influenced in a great measure in the decision of the technologies used for the development. For this reason, we analyzed the hosting services offered by 23 companies, with platforms GNU/Linux and platforms Microsoft Windows. Each one of these companies offers different

proposals, in which the proposals of more cost offer better services. For this reason, we have only take into account the proposal with minimum services of each analyzed hosting servers. The figure 4 summarizes a part of the obtained results:

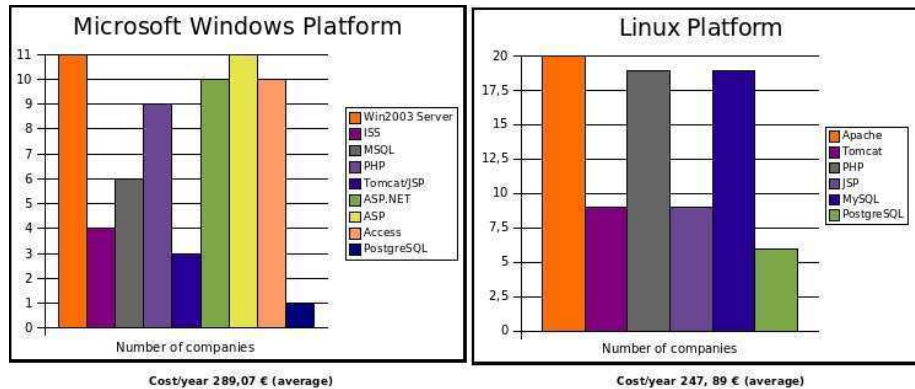


Figure 4. Part of the market research result

We have obtained important conclusions starting from our market research:

- Most of the servers provide support for PHP and MySQL, in GNU / Linux servers and Microsoft Windows servers.
- The number of servers which allow us to install applications implemented in Java and JSP is not very numerous.
- Of the 23 companies, only 11 of them work with GNU / Linux servers.
- Only 3 companies work exclusively with Windows servers.
- The GNU / linux servers suppose an annual smaller cost than the Microsoft Windows server.

4. Application with the traditional teaching methods

To apply different traditional teaching methods was one of the objectives that we sought to get with the creation of the system. If we remind the methods of the section 1.1, the *Grammar-Translation method* can be applied with the usage of many theoretical and practical documents. The user could improve his ability to write the language which he/she wants to learn correctly. The *Direct method* can be applied through the communication in real time with the rest of users of the system. The communication can be established by means of the writing process (chat) or by means of the speech (voice IP). The objective of this method is to reach a great linguistic fluency. On the other hand, it is possible to associate files (images, sounds ...) to the different types of contents. Therefore,

the user will be able to practice the *Audio-Oral* and *Audio-Visual* methods. Finally, the *Notional-Functional method* can also be used because all contents belong to thematic categories.

5. Conclusions

Most of countries consider important to learn one or two languages besides the native language, because this learning has important advantages at various levels (personal level, professional level, ...). Until some years ago, it was only possible the learning of languages in educational specialized centers by means of the study of books or travelling to foreign countries. The appearance of new technologies and Internet like a communication mechanism enlarges the number of possibilities in the process of learning language.

In the environment of the learning of languages, most of existing systems on the network have a series of static contents which are upgraded in a way centralized by the administrators of the web site. These systems delete the wealth of the collaborative learning among users. With the creation of Babel, we propose a free tool for the support in the learning of languages which the users can use to contribute freely with their knowledge to the growth of the community base contents.

The state machine regulates the behaviour in the process of acquisition and data handling, allowing the system to adapt to the necessities and preferences of the environment in which is implanted. Also, the system is supported by communication tools, dictionaries, voice synthesis, search engine of contents and games that make it more complete. The modular design of the system will allow the user to add new types of contents and new modules in the future easily.

6. Future works

In our current work, we concentrate on three research lines. First, is the enhancement of the dictionaries, that is to say, dictionaries which allow the semantic search and the definition of ontologies. Another line of work is the incorporation of an access mechanism more versatile than the based on semaphores. The use of a low-level layer to control the versions by means of CVS or SVN would allow a better parallelism in the work with the system documents. Finally, we are creating an interface in which the system administrator could see the current configuration. This way the administrator could modify it in a visual way. Currently, the setup of the state machine is shown by means of tables. This fact can hinder the administrator to have a general perspective of the setup.

7. Acknowledgments

This work has been funded by the *Consejería de Ciencia y Tecnología* and the *Junta de Comunidades de Castilla-La Mancha* under Research Projects PAC-06-0141 and PBC06-0064.

Referencias

1. Richards, C., Rodgers, T. S.: Enfoques y métodos en la enseñanza de idiomas. Cambridge Press, (1998).
2. Sanchez Perez, A.: La Enseñanza de Idiomas. Principios, problemas y metodos. HORA, S.A.(1982).
3. Parker A.: Interaction in Distance Education: The Critical Conversation. Educational Tecnology Review, 12. p. 13-17. (1999).
4. Rogers J.: Communities of Practice: A framework for fostering coherence in virtual learning communities. Educational Technology & Society. p. 384-392. (2000).
5. Romiszowski, A.J.: Web-Based Distance Learning and Teaching: Revolutionary Invention or Reaction to Necessity?. Web-Based Instruction – Englewood Cliffs. (1997).
6. Vaquero A., Saenz F., Barco A.: Improving the Language Mastery through Responsive Environments. Computers and Education – Towards an Interconnected Sopciety. Ed. Kluwer Academic Publishers. (2001).
7. Steve Krug.: Don not make me think! A common sense approach to web usability. Pearson Educacion. Madrid. 2001

Sistema experto para la simulación de sistemas tácticos de baloncesto con software libre

Manuel Palomo¹ y Francisco Jesús Martín–Mateos²

¹ Grupo Mejora del Proceso Software y Métodos Formales
Departamento de Lenguajes y Sistemas Informáticos, Universidad de Cádiz
Escuela Superior de Ingeniería, C/ Chile, s/n. 11003 Cádiz, Spain

`manuel.palomo@uca.es`

² Grupo de Lógica Computacional
Departamento de Ciencias de la Computación e Inteligencia Artificial, Universidad de Sevilla

E.T.S.I. Informática, Avda. Reina Mercedes, s/n. 41012 Sevilla, Spain

`fmartin@cs.us.es`

Resumen Este artículo presenta la labor de desarrollo de un sistema experto para la simulación táctica de baloncesto. El diseño de sistemas expertos es una rama de la inteligencia artificial cuya finalidad es realizar sistemas informáticos que simulen la capacidad de decisión de un experto humano en una materia determinada. Este sistema permite simular una jugada seleccionando la táctica del equipo atacante y la del defensor. Para la implementación se utilizó un sistema experto basado en reglas realizado en el entorno libre CLIPS.

Palabras clave: Sistema experto, reglas, simulación, táctica baloncesto

1. Introducción

El objetivo de este proyecto era desarrollar un sistema para la simulación de distintas tácticas de baloncesto. El programa presenta un manejo sencillo y permite que se puedan introducir nuevas tácticas de ataque y de defensa al conjunto de tácticas predefinidas que incorpora el sistema. Como principal objetivo se marcó que el comportamiento del sistema fuera lo más fiel posible al desarrollo de un sistema táctico en un partido de baloncesto (salvando las limitaciones de la representación en ordenador del mundo real), incluyendo todas las variantes de un sistema táctico en su justa medida. Cabe destacar que los jugadores no son autónomos, sino que están coordinados para comportarse de acuerdo a la táctica que desarrollan.

Este proyecto fue presentado como Proyecto Fin de Carrera de Ingeniería en Informática en la Escuela Técnica Superior de Ingeniería Informática de la Universidad de Sevilla, en septiembre de 2001. Fue realizado por Manuel Palomo Duarte y tutorizado por Francisco Jesús Martín–Mateos y obtuvo la calificación de *Matricula de Honor*.

2. Conceptos básicos sobre sistemas expertos

El diseño de sistemas expertos es una rama de la inteligencia artificial cuya finalidad es realizar sistemas informáticos que simulen la capacidad de decisión de un experto humano en una materia determinada. En la actualidad los sistemas expertos se usan con gran éxito en muchos campos de la ciencia como la medicina o la ingeniería.

Los principales elementos de los sistemas expertos basados en reglas son la base de conocimiento, su motor de inferencia y la interfaz de usuario.

La interfaz de usuario es el protocolo que permite ejecutar y controlar el sistema experto (ya sea por un humano o por otro sistema).

La base de conocimiento de un sistema experto es el conjunto formado por todo el conocimiento que tiene el sistema en un momento dado. Existen sistemas en los que el conocimiento se mantiene constante a través del tiempo, aunque suelen ser más comunes aquellos en que el sistema adquiere conocimiento a durante su ejecución. A partir de este conocimiento, el sistema infiere información, que será, directa o indirectamente, la salida del sistema.

En los sistemas basados en reglas el conocimiento se almacena en forma de hechos (afirmaciones sobre el sistema) y reglas que permiten inferir información.

El entorno usado en este proyecto es CLIPS [CLIPS]. CLIPS es un entorno para el desarrollo y ejecución de sistemas expertos. Inicialmente fue desarrollado por la NASA, pero actualmente se mantiene de manera independiente es software de dominio público. Su motor de inferencia aplica una serie de reglas del tipo *si {condición} entonces {acción}* a un conjunto de datos para generar datos nuevos. En la figura 1 se puede observar esta relación.

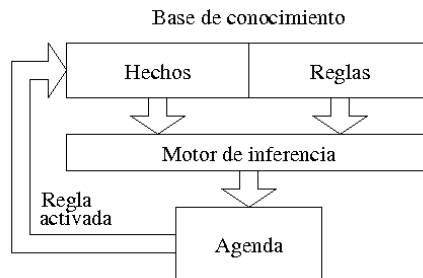


Figura 1. Diagrama esquemático de funcionamiento del algoritmo CLIPS.

La *agenda* es el conjunto de reglas que se pueden ejecutar en un momento de tiempo dado (porque todas sus condiciones se cumplen). Sin embargo eso no implica que todas esas regla se tengan que ejecutar necesariamente. De entre todas ellas se dispara una. Esa regla es probable que modifique la base de conocimiento y, por lo tanto, tenga que recalcularse la la agenda. La decisión de

qué regla en concreto se ejecuta depende de la implementación concreta, puede ser por prioridades, al azar, etc.

Un ejemplo de regla en CLIPS puede ser :

```
(defrule biblioteca-regla-1
  (libro (titulo ?X) (estado retraso) (prestado-a ?Y))
  (persona (nombre ?Y) (dirección ?Z))
  =>
  (mandar-nota-retraso ?X ?Y ?Z))
```

Esta regla en lenguaje natural sería algo como:

Regla 1:

Si

hay un libro, de título X, con retraso y prestado a Y

Y

la dirección de esa persona es Z

Entonces

mandar una nota de retraso a Y en Z del libro X.

Los sistemas expertos clásicos suelen tener un conjunto de reglas fijo y un conjunto de hechos en cambio constante. Sin embargo, se comprueba que en la mayoría de sistemas expertos, la mayor parte de este conjunto también permanece constante al aplicar las reglas. Aunque siempre se añaden hechos nuevos y algunos viejos se borran, la proporción de hechos que cambian por unidad de tiempo es más bien pequeña.

Por esta razón, la implementación directa del entorno para sistemas expertos es muy ineficiente. Esta implementación tendría una lista de reglas y las recorrería circularmente de modo indefinido, comprobando la parte izquierda de cada regla (el antecedente, denominado *LHS*, del inglés *Left-Hand-Side*) con la base de conocimiento y ejecutando la parte derecha (el consecuente, o *RHS*, *Right-Hand-Side* en inglés) de las reglas a aplicar. Esto es bastante ineficiente porque la mayoría de las comprobaciones de las condiciones en cada iteración tendrán igual resultado que en la iteración anterior. Esta idea se llama regla de aproximación a la búsqueda de hechos. Y tiene una complejidad computacional del orden de $O(R*F^P)$, siendo R el número de reglas, P la media de comprobaciones por el *LHS* de cada regla y F la cantidad de reglas de la base de conocimiento. Evidentemente, este orden no es nada deseable, pues se dispara al incrementarse las comprobaciones por regla.

CLIPS usa, en vez de el algoritmo anterior, otro más eficiente denominado algoritmo *Rete*¹. Este algoritmo [Rete82] ha sido la base de toda una generación de entornos rápidos y eficientes para sistemas expertos: *OPS5*, y sus derivados *ART* y CLIPS. El problema de eficiencia descrito anteriormente es atenuado en el algoritmo *Rete* recordando resultados anteriores de la evaluación de la regla en las iteraciones siguientes. Sólo se comprueban los hechos nuevos en las *LHS*

¹ *Rete* significa red en latín.

de las reglas. Además se crea un red de nodos que se comparten entre reglas y se reorganizan los hechos de los *LHS* de las reglas que tienen más probabilidad de ser relevantes. El resultado es, en complejidad computacional de operaciones por iteración, del orden $O(R * F * P)$ y lineal respecto al tamaño de la base de conocimiento.

Veamos un ejemplo. Sean las siguientes reglas:

```
(defrule example-2      (defrule example-3
(x)                    (x)
(y)                    (y)
(z)                    =>
=>                      )
)
```

Estas reglas se optimizarían en memoria como se observa en la figura 2.

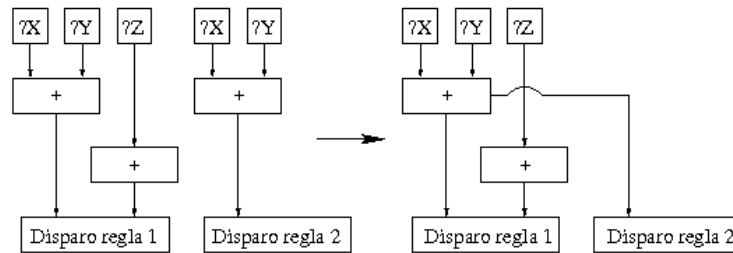


Figura 2. Optimización del algoritmo *Rete*.

Este apartado ha presentado las ideas fundamentales del algoritmo *Rete*. Para una visión más amplia, el lector puede remitirse a [Expe93].

3. Análisis

Se realizó un estudio sobre las funcionalidades que era necesarias para que el sistema realizara una simulación táctica de baloncesto de acuerdo a la realidad. Como principal referencia bibliográfica se usó [NuCo98].

Las principales conclusiones que se extrajeron fueron:

- En el baloncesto, por lo general, cualquier jugador que tire a canasta sin oposición (lo que se conoce como *realizar un tiro cómodo*) tiene muchas posibilidades de encestar el balón. Por el contrario, el porcentaje de aciertos en tiros no cómodos (cuando existe algún defensor amenazando al tirador) es mucho menor.

- Para obtener un tiro cómodo el método más común es realizar buenos bloqueos. Un bloqueo es un choque entre un atacante y un defensor que permite a un segundo atacante moverse más libremente (normalmente para tirar a canasta o recibir o dar un pase).
- En el baloncesto las tácticas tienen una importancia vital para el desarrollo de un partido. El reducido número de jugadores por equipo, cinco, hace que sea importante que todos los jugadores intervengan en mayor o menor medida en la jugada.

A continuación se describen las claves del movimiento de bloqueo y las principales tácticas de defensa y ataque del baloncesto moderno.

3.1. El bloqueo

Un bloqueo es un choque entre dos jugadores de distinto equipo que se provoca para que un atacante tenga más libertad. Es importante que el jugador atacante que choque esté quieto mientras se produce el contacto, pues en otro caso se le señalará falta personal.

Los pasos para realizar un bloqueo son los siguientes:

Preparación: un primer atacante se queda quieto mientras otro compañero se acerca a él.

Realización: el segundo atacante hace un movimiento rápido y pasa rozando al compañero (de modo que este moleste lo máximo posible a su defensor).

Acción del atacante: una vez ha pasado el atacante tiene un amplio abanico de posibilidades. Si tiene el balón puede tirar a canasta, acercarse a ella, pasar a un compañero, etc. En caso de que no lo tenga suele prepararse para recibir un posible pase.

Reacción del defensor: la reacción del defensor es muy importante. Entre otras acciones puede intentar seguir a su pareja (aunque a cierta distancia), puede ordenar al jugador que bloquea, etc.

No existe ninguna decisión netamente mejor que otra para cada uno de los participantes en un bloqueo. Según sus posiciones concretas, el entorno y las características de cada jugador, cada uno debe tomar la decisión que considere oportuna.

3.2. Principales tácticas de defensa

Las tácticas de defensa se clasifican fundamentalmente en dos tipos: individuales y en zona. Conviene destacar que las defensas no individuales no están permitidas en la N.B.A.²

En las individuales el entrenador indica a cada jugador qué contrincante debe defender. La posición ideal del defensa con respecto a su par depende de muchos

² La N.B.A. es la liga profesional de baloncesto de Estado Unidos, la competición más potente del mundo.

aspectos: si tiene o no el balón, las capacidades físicas de cada uno, el marcador y tiempo restante, las posiciones de los demás jugadores, etc. Por lo general un defensor suele colocarse entre su pareja y la canasta, a mayor o menor distancia de él según los factores comentados anteriormente.

Esta táctica tiene la ventaja de que, si se hacen bien, suelen cansar mucho a los contrarios física y psicológicamente. Por el contrario, su punto débil es que un fallo puntual de defensa (por un despiste o un bloqueo) suele permitir una canasta fácil.

Por contra, las tácticas en zona dividen el terreno a defender en varias regiones y asignan a cada jugador una para que la defienda. Pero, por supuesto, esta división no es estricta y los defensores se ayudan unos a otros según se muevan los atacantes. Las tácticas más famosas son la zona 2-1-2 (también conocida como zona 2-3) y la 1-3-1 (figura 3).

Las ventajas de este tipo de defensas es que el equipo defensor se cansa poco (lo que permite que ataque con más intensidad). Además (según la configuración que se realice) se suelen tener bastantes jugadores en las cercanías del aro, lo que ayuda a contrarrestar el juego interior del atacante y coge rebotes de tiros fallados.

Como punto débil hay que destacar que por lo general las defensas en zonas suelen dejar más libres a los atacantes que juegan alejados de la canasta. De modo que si el equipo contrario tiene buenos tiradores pueden conseguir canastas de tres puntos con cierta facilidad.

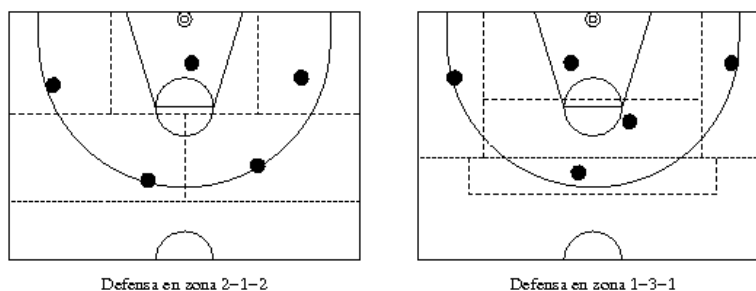


Figura 3. Defensas en zona.

Por último existen las denominadas tácticas mixtas, que mantienen a algunos jugadores defendiendo en zona cerca de la canasta mientras que otros realizan defensa individual sobre los atacantes más peligrosos.

Se suelen usar cuando en el equipo atacante sólo hay uno o dos jugadores que encestan con facilidad. Estos reciben defensa individual, para cansarles más,

mientras que los otros defensores hacen una defensa relajada sobre el resto del equipo y están atentos a un fallo en el marcaje individual para echar una mano.

3.3. Principales tácticas de ataque

Hay tantas tácticas de ataque como pueda dar de sí la imaginación de un entrenador. Cualquier secuencia de movimientos, bloqueos, pases, quiebros, etc que faciliten un tiro cómodo es válida. Sin embargo, por lo general, casi todas las tácticas suelen basarse en dos situaciones iniciales, que son jugar al poste alto y jugar al poste bajo (figura 4).

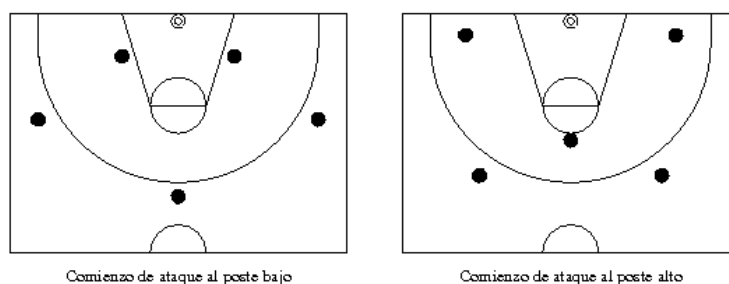


Figura 4. Formaciones iniciales de ataque.

Cuando se juega al poste bajo los dos jugadores más altos del equipo ocupan posiciones a sendos lados de la canasta. Los otros tres se distribuyen a lo largo de la línea de tres puntos, uno en la perpendicular al aro y los otros dos a los lados.

El punto fuerte de este tipo de ataque es que permite, si se tienen jugadores altos fuertes, que estos reciban el balón cerca de la canasta. Además, en el caso de que venga otro defensor para ayudar a cubrirlo, pueden realizar un pase hacia el jugador que quede libre, que dispondrá en ese caso de un tiro cómodo o libertad para moverse. La principal desventaja es que los jugadores altos del equipo defensor sean mejores que los del equipo atacante.

Por otro lado, al jugar al poste alto el jugador más alto del equipo ocupa posición en la cabecera de la zona, y los demás jugadores se distribuyen alrededor de él haciendo una especie de trapecio.

La ventaja de este ataque es que nuestro jugador más alto obliga al jugador más alto del equipo defensor a alejarse de la canasta, por lo que los atacantes que ocupan los laterales tendrán mucha más libertad de acción para acercarse a la canasta o driblar para hacer un tiro cómodo. También, si nuestro jugador alto es más hábil que su par, puede fintarle para entrar a canasta o bien hacer un pase a cualquiera de sus cuatro compañeros, pues desde el punto donde está los tiene a la vista.

El problema de esta táctica es que los jugadores de perímetro del equipo defensor sean más habilidosos que los nuestros, y estos no sepan que hacer para zafarse de ellos. Otro problema es que, prácticamente, se renuncia a capturar rebotes de ataque.

4. Modelado del sistema

En lo referente al modelado del sistema las principales decisiones fueron:

4.1. Campo de juego

Sólo se simula medio campo, como en toda la bibliografía táctica de baloncesto. La razón es que en baloncesto las jugadas de ataque se desarrollan íntegramente en el campo del rival y no está permitido volver al campo propio. Además, si se quiere ver la capacidad del equipo que ataca en ese momento para defender en el simulador, basta con volver a ejecutar el programa cambiando el equipo atacante por el defensor, y viceversa.

Una decisión de especial importancia es la representación del campo de juego. Se optó por una representación de la cancha como un tablero de 200 por 200 casillas. De este modo nos acercamos a la libertad total que ofrecería un modelo continuo (debido al alto número de casillas), pero manteniendo la manejabilidad de un modelo discreto.

4.2. Representación de los jugadores

La situación de los jugadores está determinada por una casilla, pero tienen a su alrededor un círculo de casillas en las que no puede haber otro jugador (que simula la masa o superficie que ocupan en el campo de juego). De este modo el porcentaje de espacio que ocupa un jugador en la cancha sería parecido al que ocupa en la realidad un jugador. Dada la importancia de los bloqueos y como por lo general el jugador que bloquea ayuda en lo posible a su compañero pero molesta al rival se permite que dos jugadores del mismo equipo solapen sus masas pero no con las de contrarios.

Todo los jugadores tienen una serie de características comunes: puesto en el equipo (un número del 1 al 5), posición en el campo (coordenadas x e y), masa que ocupa en el campo (de 3 a 5 casillas). y velocidad (número de micromovimientos por unidad de simulación, de 1 a 10).

Además los defensores tienen las siguientes características adicionales: par (jugador al que defiende individualmente), capacidad para interceptar un pase y capacidad para taponar un tiro (aunque los jugadores no taponan este número influye negativamente en el tiro de un contrario cercano).

Por su parte, los atacantes también tienen dos características propias: capacidad de tiro de 2 puntos y de 3 (pues los jugadores suelen especializarse en uno u otro tipo de tiro).

4.3. Representación del balón

El balón es una entidad asociada siempre a un jugador. Puede tenerlo asociado un jugador atacante que bota el balón, un defensor que lo haya interceptado en un pase, o un atacante que haya encestado o fallido un tiro.

4.4. Movimientos de los jugadores según los equipos

Se optó por realizar movimientos por turnos. En concreto, turnos en los que las posibilidades de cada jugador son pequeñas, de este modo prácticamente parezca una simulación en paralelo.

Como el baloncesto es un juego en que el equipo atacante tiene un tiempo limitado para conseguir canasta (en caso de no conseguirlo y consumirse ese tiempo, el balón pasa al equipo rival), se decidió que fuera el equipo atacante el que primero moviese todos sus jugadores para intentar conseguir una situación de ventaja y que después el equipo defensor respondiera a ese movimiento.

4.5. Acciones que pueden realizar los jugadores

Los jugadores atacantes pueden, en cada turno, realizar una sola de las siguientes acciones: desplazarse, pasar el balón o tirar a canasta. Así se evita que un jugador atacante se desplace a una posición cómoda y realice un tiro a canasta sin que su defensor pueda hacer nada por evitarlo.

La función que calcula si un tiro entra o no tiene en cuenta la capacidad de tiro concreto (2 o 3 puntos) del jugador, su distancia al aro, la habilidad para taponar de su defensor más cercano y un componente aleatorio.

Por su parte, la única acción que pueden realizar los defensores es moverse para evitar una canasta del rival. En caso de que se realice un pase cuya trayectoria pase lo suficientemente cerca de un defensor, este interceptará el balón automáticamente. El simulador no contempla la posibilidad de realizar taponar, pero sí que tiene en cuenta la capacidad de taponar de un jugador cercano al tirador para bajar las probabilidades de enceste.

4.6. Movimientos individuales de los jugadores

Se decidió que los jugadores efectuaran movimientos de varias casillas en cada turno, porque un movimiento de una doscientaava parte de campo sería imperceptible. Los movimientos posibles son ocho (arriba, abajo, izquierda, derecha y sus diagonales). Pero sus movimientos se dividen en una serie de micro-movimientos, para que así tengan más libertad y eviten problemas que se derivan de situaciones en que el movimiento más adecuado no sea ni horizontal, ni diagonal, sino un grado medio.

Un ejemplo se puede observar en el primer esquema de la figura 5. En él se observa que un jugador desea alcanzar una posición marcada con una \times .

En el caso que el jugador sólo pudiera hacer un movimiento largo en una única dirección, no podría hacer el movimiento diagonal, puesto que el otro jugador

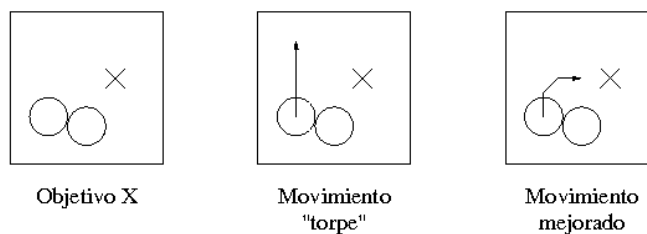


Figura 5. Movimientos del jugador.

se lo impide. Por lo tanto, el movimiento que debe de hacer es hacia arriba, consiguiendo el resultado que se ve en el segundo diagrama.

Sin embargo, se divide ese desplazamiento en varios micro-desplazamientos, existe una serie de micro-movimientos que nos llevan prácticamente al lugar deseado, como se ve en el último esquema.

4.7. Tácticas de defensa

Cada vez que un defensor tiene la posibilidad de moverse éste evalúa si es posible moverse a alguna de las ocho casillas adyacentes. Acto seguido se evalúan una serie de propiedades para cada posible estado (incluido el caso de quedarse en la casilla que ocupa) y decide el mejor movimiento. La figura 6 muestra un esquema.

Las propiedades concretas dependen de la táctica, y es lo que las diferencia. Por ejemplo, en una defensa al hombre los defensores intentarán colocarse entre su par y la canasta, y lo seguirán por todo el campo. Sin embargo, en una defensa en zona hay que evaluar si al hacer un movimiento un defensor sale de su zona de defensa, porque esa es su área de trabajo.

4.8. Tácticas de ataque

Las tácticas de ataque son algo más complejas que las de defensa. Por un lado cada atacante una mayor cantidad de posibles acciones a realizar: desplazarse, pasar el balón a cualquiera de los cuatro compañeros y tirar a canasta. Además, mientras que un defensor sólo tiene que reaccionar a los movimientos de los atacantes (lo que es relativamente sencillo), el atacante por su parte tiene que provocar movimientos que desestabilicen la defensa.

Respecto a las acciones que puede realizar se implementó un sistema táctico que permitiera indicar a los jugadores que hagan pases o tiros, de modo que los jugadores sólo tiran o pasan cuando el sistema se lo permitiera. Esto no quiere decir que los ataques sean siempre iguales, porque una táctica puede adaptarse a la situación en el campo y cambiar se desarrollo. Por su parte, el modelo para

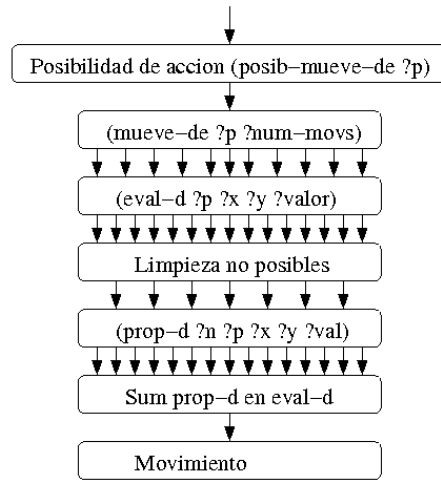


Figura 6. Movimientos.

hacer movimientos se realizan de igual modo que en la defensa, guiados por objetivo.

La forma más usada para implementar el ataque es por pasos. En cada paso de simulación cada jugador tiene que ir a una posición del campo (que puede ser fija o relativa a otros jugadores o el balón). Los jugadores atacantes se desplazan en dirección a una casilla objetivo que deben de alcanzar (o, al menos, acercarse lo máximo posible).

La transición de un paso a otro puede producirse porque los jugadores estén ya en sus objetivos (o suficientemente cerca) o porque pase un tiempo y el jugador no llegue.

Uno de los problemas que se detectó es que, a veces, el camino en línea recta no le permite llegar a esa casilla (e incluso puede que no acercarse mucho). Un ejemplo se puede ver en el primer diagrama de la figura 7.

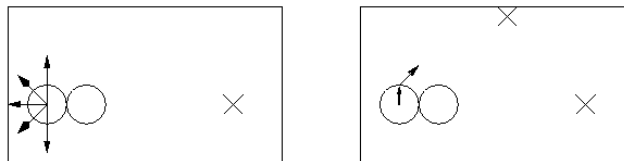


Figura 7. Paso directo al objetivo cerrado.

En esta figura, el atacante desea llegar al objetivo que está a su derecha (marcado con una \times), pero no puede porque hay un defensor justo en el centro de su camino. En esta situación el jugador sólo puede realizar los movimientos indicados por flechas. Está claro que los movimientos hacia atrás o hacia atrás en diagonal son movimientos malos y descartables. Sin embargo, los movimientos laterales, aunque midiendo directamente la distancia pudieran parecer malos (pues la distancia al objetivo aumenta), son interesantes, pues nos permiten un camino más directo hacia el objetivo.

Para conseguir este efecto se decidió que el sistema, en caso de que detecte una situación de este tipo, asigne un segundo objetivo al atacante de manera que este intente minimizar su distancia a ambos objetivos. Este segundo objetivo sólo permanecerá durante los micro-movimientos que haga el jugador el turno en curso, pero no para los siguientes. Así se evitan cálculos estériles y en cada movimiento, si es necesario, se vuelve a calcular otro subobjetivo mejor para esa situación concreta (como se ve en el segundo diagrama de la figura 7).

5. Implementación del sistema

Debido a la escasa potencia de las máquinas en las fechas en que se realizó el proyecto (año 2001) se decidió separar el programa en dos partes: un motor que realizara la simulación completa y una interfaz gráfica que mostrara el resultado en pantalla. Para la comunicación entre ambos se usa un fichero de texto.

Se estudió era usar JESS [Jess], un proyecto que comenzó siendo un clon de CLIPS en JAVA pero que actualmente incorpora muchas características adicionales. Sin embargo una serie de limitaciones que presentaba en el momento de realizar el proyecto relacionadas con la gestión de módulos en el programa y problemas con los números aleatorios hizo que se descartara dicha opción en favor de CLIPS con Tcl/Tk para la interfaz gráfica.

La interacción con el sistema es muy sencilla. El usuario selecciona el equipo atacante, el defensor y la táctica de cada uno. Entonces la interfaz carga en CLIPS los ficheros básicos del sistema y los correspondientes a la elección del usuario. Después manda realizar la simulación y cuando esta termina muestra al usuario la pantalla de control de la simulación que permite visualizar la simulación ya realizada de manera continua y avanzar o retroceder según se desee. En la figura 8 se observa la pantalla de simulación.

6. Conclusiones finales

Como se ha visto, el campo de la simulación táctica del baloncesto es un ejemplo interesante para aplicar técnicas de sistemas expertos basados en reglas. La potencia de estos sistemas facilita la implementación de los diferentes aspectos que tiene que considerar cada uno de los miembros de un equipo de baloncesto de acuerdo a su táctica.

El programa en sí, tiene una base muy interesante, pero a nivel técnico ha quedado desfasado. El aumento de la potencia de los ordenadores de hoy en día



Figura 8. Programa mostrando la simulación.

permite incrementar considerablemente el número de reglas del sistema (pudiendo considerar más aspectos en cada decisión). Además tampoco se hace necesario crear un fichero intermedio, pues se puede controlar la ejecución de cada paso de simulación en CLIPS desde Tcl/Tk o usar JESS.

Por último, la opción más interesante de cara a poder darle vida y difusión al proyecto probablemente sea reconvertirlo en un entorno donde se pudiera competir, al estilo de *Robocup SoccerServer* [Ress]. Este sistema permite realizar simulaciones de partidos de fútbol robótica en los que cada jugador es un agente autónomo controlado por un ordenador. De este modo se consigue unir la enseñanza con la diversión y la competición, consiguiendo resultados muy interesantes (puede consultarse [Palo05] si se desea más información sobre las posibilidades del sistema).

Referencias

- [CLIPS] *Página principal de CLIPS*. <http://www.ghg.net/clips/CLIPS.html>
- [NuCo98] “Lee” Walker, A.L. ; Donohue J. *Nuevos conceptos de ataque para un balon-cesto moderno*. Editorial Paidotribo, 1998.
- [Rete82] Charles L. Forgy. *Rete: a fast algorithm for the many pattern/ many object pattern match problem*. Artificial Intelligence n.19, 1982.

- [Expe93] Giarratano and Riley. *Expert Systems: Principles and Programming, Second Edition*. PWS Publishing , 1993.
- [Jess] *Página oficial de Jess*. <http://www.jessrules.com>
- [Rcss] *Página oficial de Soccerserver*. <http://sserver.sourceforge.net>
- [Palo05] Palomo, Manuel. *Soccerserver: fútbol robótico GPL*. Actas de las V Jornadas Andaluzas de Software Libre, 2005.

3D Distributed Rendering and Optimization using Free Software

Carlos González-Morcillo^{1,2}, Gerhard Weiss², David Vallejo¹, Luis Jiménez¹ y
Jose A. Fdez-Sorribes¹

¹ Escuela Superior de Informática, University of Castilla-La Mancha
Paseo de la Universidad, 4. Ciudad Real (Spain)
{Carlos.Gonzalez, David.Vallejo, Luis.Jimenez}@uclm.es

² Software Competence Center, Hagenberg (Austria)
{Carlos.Morcilla, Gerhard.Weiss}@scch.at

Resumen The media industry is demanding high fidelity images for their 3D synthesis projects. Rendering is the process by means of which a 2D image can be obtained from the abstract definition of a 3D scene. Despite the development of new techniques and algorithms, this process is computationally intensive and requires a lot of time to be done, specially when the source scene is complex or when photo-realistic images are required. This paper describes Yafrid (standing for *Yeah! A Free Render grID*) and MagArRO (*Multi Agent AppRoach to Rendering Optimization*) architectures developed in the University of Castilla-La Mancha for distributed rendering optimization.

Palabras clave: Rendering, Optimization, Artificial Intelligence, Agent.

1. Introduction

Physically based Rendering is the process of generating a 2D image from the abstract description of a 3D scene. The process of constructing a 2D image requires several phases such as modelling, setting materials and textures, placing the virtual light sources and rendering. Rendering algorithms take a definition of geometry, materials, textures, light sources and virtual camera as input and produce an image (or a sequence of images in the case of animations) as output. High-quality photorealistic rendering of complex scenes is one of the key goals of computer graphics. Unfortunately, this process is computationally intensive and requires a lot of time to be done when the rendering technique simulates global illumination. Depending on the rendering method and the scene characteristics, the generation of a single high quality image may take several hours (or even days!).

Because of the huge amount of time required to be done, the rendering phase is often considered to be a bottleneck in photorealistic projects in which one image may need some hours of rendering in a modern workstation.

To solve this problem, several approaches based on parallel and distributed processing have been developed. One of the most popular is the render farm: a computer cluster owned by an organization in which each frame of an animation is independently calculated by a single processor. There are new approaches called Computational Grids which uses the Internet to share CPU cycles. Yafrid is a computational Grid that distributes the rendering of a scene among a large number of heterogeneous computers connected to the Internet.

This paper describes the work flow and the free software tools used in the University of Castilla-La Mancha in several 3D rendering projects (based on OSCAR Open Source Cluster Application Resources, and Blender & Yafray render engines), as well as our new research software distributed under the GPL license. Going into detail, the global architecture of Yafrid and the optimization system based on principles from the area of multi-agent system called MagArRO are exposed. This last system uses expert knowledge to make local optimizations in distributed rendering. Finally, some experimental results which show the benefits of using these distributed approaches are presented. The paper is structured as follows. The following section overviews the state of the art and the current main research lines in rendering optimization. Thereby, the focus is on the issues related to parallel and distributed rendering. The next sections describe the general architecture of an Oscar-based cluster, the Grid-based rendering system called Yafrid and the Distributed Intelligent Optimization Architecture called MagArRO. Then, in the next section empirical results that have been obtained by using these systems are shown. The final section is dedicated to a careful discussion and concluding remarks.

1.1. Related Work

There are a lot of rendering methods and algorithms with different characteristics and properties [11,6,10]. However, as pointed out by Kajiya [6], all rendering algorithms aim to model the light behaviour over various types of surfaces, and try to solve the so-called rendering equation which forms the mathematical basis of all rendering algorithms. Common to these algorithms, the different levels of realism are related to the complexity and the computational time required to be done. Chalmers et al. [3] expose various research lines in rendering optimization issues.

Optimizations via Hardware. One alternative to decrease time is making special optimizations using hardware. In this research line there are different approaches; some methods use the programmable GPUs (Graphics Processing Units) as massively parallel, powerful streaming processors which run specialised portions of code of a raytracer. The use of programmable GPUs outperforms the standard workstation CPUs by over a factor of seven [2]. The use of the CPU in conjunction with the GPU requires new paradigms and alternatives to the traditional architectures. For example, the architectural configurations proposed by Rajagopalan et al. [8] demonstrate the use of a GPU to work on real-time

rendering of complex datasets of data which demand complex computations. There are some render engines designed to be used with GPU acceleration, such as Parthenon Renderer [5], which uses the floating-point of the GPU, or the Gelato render engine which works with Nvidia graphic cards.

Optimizations using distributed computing. If we divide the problem into a number of smaller problems (each of them is solved on a separate processor), the time required to solve the full problem would be reduced. In spite of being true in general, there are many distributed rendering problems that would be solved. To obtain a good solution of a problem on a distributed system, all processing elements must be fully utilized. Therefore, a good task scheduling strategy must be chosen. In a domain decomposition strategy [3], each processing unit has the same algorithm, and the problem domain is divided to be solved by the processors. The domain decomposition can be done using a data driven or demand driven approach. In data driven model, the tasks are assigned to the processing units before computation starts. In the other alternative, demand driven model, the tasks are allocated dynamically when the processing units become idle. This is done by implementing a pool of available tasks. This way, the processing units make a request for pending work.

In both models (data and demand driven) is necessary a cost estimation function of each task. This cost prediction is very difficult to be exactly done before the image has been completed due to the nature of global illumination algorithms (unpredictable ray interactions and random path of light samples).

The biggest group of distributed and parallel rendering system is formed by dedicated clusters and rendering farms. Some 3D animation companies use their rendering farms in their own productions and some of them offer rendering services via Internet. According to the way the division of tasks is done, we talk about **fine-grained** systems, in which each image is divided in small parts that are sent to a processor to be done independently, or **coarse-grained** (in case of animations) in which each frame of an animation is entirely done by one processing unit. Dr. Queue [17] is an open source tool designed to distribute frames through a farm of networked computers. This multiplatform software works in coarse-grained division level. In section 2, our solution based on Oscar open cluster [18] is exposed.

New approaches of distributed rendering use a grid design to allocate the tasks among a large number of heterogeneous computers connected to the Internet, using the idle time of the processor [1]. This emerge technology is called *Volunteer Computing* or *Peer-to-peer* computing, and is currently used in some projects based on the BOINC technology (such as BURP [16] (*Big and Ugly Rendering Project*)). In section 3, the main architecture of Yafrid and the key advantages are exposed.

Cost prediction. The knowledge about the cost distribution across the scene (i.e. across the different parts of a partitioned scene) can significantly aid the allocation of resources when using a distributed approach. In fact, this estimation

is absolutely necessary in commercial rendering production, to assure deadlines and provide accurate quotations for the work to be done. There are many approaches based on knowledge about cost distribution; a good example is [9]. In section 4.1, the cost prediction mechanism used in MAgArRO is exposed.

Distributed Multi-Agent Optimization. The distribution of multi-agent systems and their properties of intelligent interaction allow us to get an alternative view of rendering optimization. The work presented by Rangel-Kuoppa [7] uses a JADE-based implementation of a multi-agent platform to distribute interactive rendering tasks on a network. Although this work employs the multi-agent metaphor, essentially it does not make use of multi-agent technology itself. In fact, the use of the JADE framework is only for the purpose of realizing communication between nodes, but without knowledge neither learning and negotiation. The MAgArRO architecture proposed in section 4 is an example of a free and Distributed Multi-Agent architecture which employs expert knowledge to optimize the rendering parameters.

2. Oscar-based cluster approach

Nowadays, universities have good practical classrooms provided with plenty of computers. This equipment is maintained and updated frequently. Nevertheless, these computers are inactive over vacation and at night. This existing hardware infrastructure can be coordinated during idle time by using free software and by creating clusters and low-cost supercomputers [14]. Oscar [18] is a software platform which allows the programming of computer clusters based on GNU/Linux. In the next section, the general architecture of the system based on Oscar will be explained. This tool have been used in the University of Castilla-La Mancha to render some 3D projects [20,22].

2.1. Architectural Overview

In our execution environment, the system is composed by 130 workstations placed in different classrooms. These computers form a heterogeneous set of PCs. Every classroom has a specific type of hardware (all based on x86 architecture). The minimal requirements to belong to the system are 500MB of RAM, a swap partition of 1GB, and to have a connection of at least 100Mbits/s (all computers are connected to one network using 100 Mbits/s switches). The figure 1 illustrates these requeriments.

The classrooms, where Oscar cluster is used, are dedicated to educational activity. For this reason, it was most appropriate not to install permanently any software in them. The subproject Thin-Oscar [19] allows us to use machines without local HD or a partition to install the operating system as members of the Oscar cluster. A Swap partition is used for the management of temporal files (used in the rendering of each project).

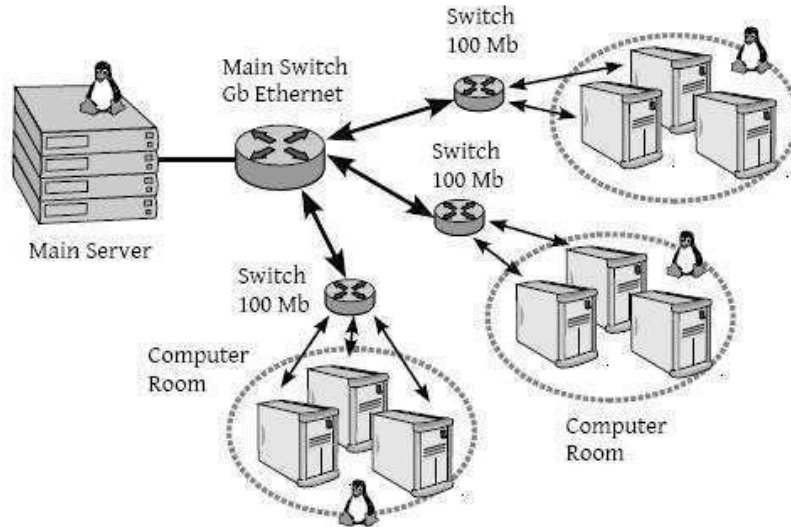


Figure 1. Oscar-based rendering farm at ESI-UCLM.

Each rendering node is configured by obtaining the configuration parameters from the network. This is done by using the PXE extension of the BIOS (Pre eXecution Environment). In our case, these data are the Operating System image in which will be executed. This way, without installing anything in the hard disk, the minimum image of the operating system is loaded (only the basic services and the render engine).

The server has two key processes to handle the PXE requests:

- **DHCPD:** It is the Dinamic Host Configuration Protocol daemon. This protocol is used to tell the client its IP and the operating system image to load.
- **TFTPD:** It is the Trivial Transfer Protocol daemon. When the server receives a file request, sends it to the client by using some configuration tables.

To begin and finish the execution of the computers in a controlled schedule, the WOL (Wake On Lan) functionality of modern computers is used. These BIOS extensions are used with the help of the motherboard and the software package Ether-Wake (developed by Donal Becker). Even if the computer is off, the network card continues listening. When the package generated by Ether-Wake arrives, the computer boots and by means of a PXE request, it loads the Operating System image. All the machines are in the same broadcast domain of the network. To avoid possible intrusions, all the machines which are being used in Oscar belong to a specific VLAN separated from the rest of the organization.

Finally, to halt the computers when the allowed time has finished, the ACPI interface must be correctly configured. To use this functionality, a simple script in the server side was made. The server establish a ssh connection to each node and send it a *shutdown* command.

3. Yafrid: Grid-based Rendering

Yafrid is basically a system which takes advantage of the characteristics of the computational grids by distributing the rendering of a scene through the Internet. The system also has other important tasks related to the management of the workunits and the controlled use of the grid.

3.1. Architectural Overview

The top-level components of Yafrid are basically the following ones:

- **Server.** The hub of Yafrid. Its basic component is the Distributor which takes works from a queue and sends them to the providers.
- **Service Provider.** This entity process the requests of the clients.
- **Client.** A client is an external entity which doesn't belongs to the system in a strict sense. Its role in the operation of the grids consists in submitting works to be done by the providers. Those works are stored in a queue from where distributor will take the next one to be scheduled.

In terms of access to the system, three user roles have been defined which determine the access privileges that the user has. Those user roles are:

- **Client.** Having this role allows an user to submit works to the grid. A client is also able to create and manage render groups (clients and providers can subscribe to these groups). When a project is created, it can belong to a group. In this case, only providers belonging to the same group can take part in the rendering of the project.
- **Administrator.** This user is necessary to operate the whole system and has complete privileges to access to the information about all the users of the system.
- **Provider.** The provider is a user that has installed the necessary software to allow the grid to sent works. Providers can access their own information and some statistics.

Yafrid server The server is the fundamental node around of which the whole Yafrid render system is established. Each one of the providers connects to this node in order to let the grid use its CPU cycles for the rendering of the scenes submitted by Yafrid clients.

Yafrid server is developed over an architecture in which, in general, four layers can be distinguished (Figure 2). This design is slightly based on the architecture

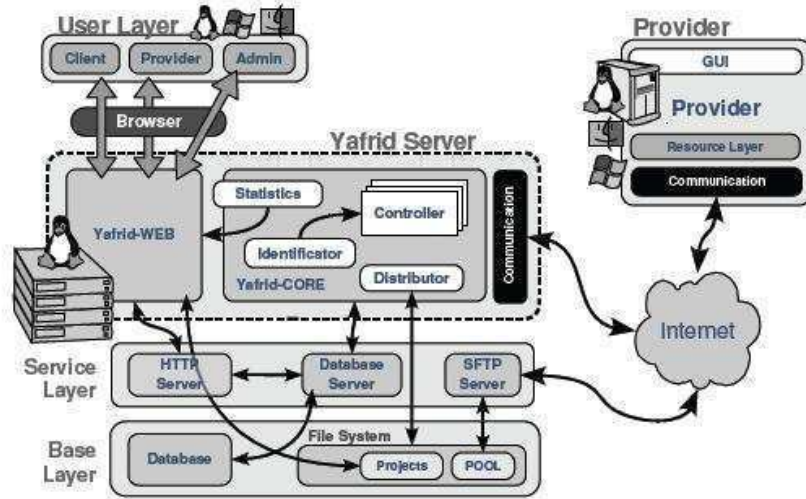


Figura 2. Yafrid General Architecture.

that appears in [4]. Those layers are '*Resource Layer*', '*Service Layer*', '*Yafrid Server*' and '*User Layer*' from lowest to highest level of abstraction.

Resource Layer. This layer has the lowest abstraction level and it is the most related with the operating system issues. Resource layer has the following components:

- **Database system.** It is in this database where the necessary tables for the correct operation of the system are maintained. Some of these tables are used to obtain statistics about the system performance while other ones store the data associated to users, groups, projects, etc. This database is accessed from the levels above by means of a database server. The current implementation uses MySQL.
- **Filesystem.** Sometimes, it is necessary to access directly to the filesystem from the layers above. Basically, the system distinguishes two types of directories. There are some directories which are used to store the workunits of the launched projects that will be accessed via SFTP by providers. Those directories compose the workunits POOL. The other category of directories is composed by those ones that contains the information about the users and their projects.
- **Network system.** The module dedicated to the communications that belongs to the main layer hides the utilization of the network resources of the computer by using a middleware (the current implementation uses ICE).

Service Layer. Basically, this layer contains the different servers that allow modules from the layers above to access the resources that belongs to the layer under this one. There are the following servers in this level:

- **HTTP Server.** Yafrid-WEB module is established over this server. As Yafrid-WEB has been developed using dynamic web pages written in a web-oriented scripting language (the current implementation has been done using PHP), the web server has to have support for this language. It is also necessary to have support for composition of graphics and for accessing to the database.
- **Database server.** This server is used by the different modules of Yafrid to access the indispensable data for the system operation.
- **SFTP server.** Accessed by the service providers to obtain the necessary files to carry out the rendering of the workunits. Once the rendering has finished, the SFTP server will be used to send to the Yafrid server the resultant image.

Yafrid Layer This is the main layer of the server and it is composed by two different modules (Yafrid-WEB and Yafrid-CORE) which work independently one of the other. **Yafrid-WEB** is the interactive module of the server and it has been developed as a set of dynamic web pages written using HTML and a web-oriented scripting language. **Yafrid-CORE** is the non-interactive part of the server. This module has been mainly developed using Python and . Yafrid-CORE is composed by three submodules; Distributor, Identificator and Statistics.

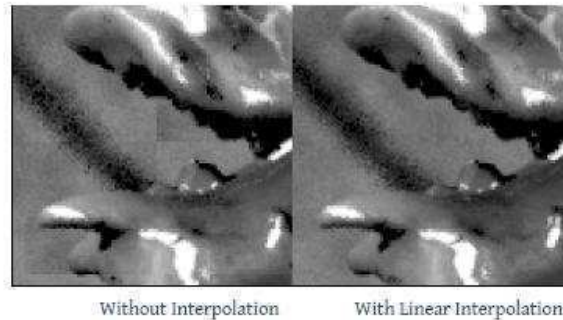


Figure 3. Artifacts without interpolation between workunits.

The **distributor** is the active part of the server. It implements the main algorithm that is aimed to do the following indispensable tasks such as generating the workunits, assigning the generated workunits to providers, controlling the Timeout, finishing projects and composing the results. With the results generated by the different providers, the distributor has to compose the final image.

This process is not trivial because slight differences between fragments can be distinguished when obtained from distinct computers due to the random component of Monte Carlo based methods (like Pathtracing). For that reason, it is necessary to smooth the joint between fragments which are neighbors using a lineal interpolation mask. We define a zone in the workunit that is used to combine with other workunits in the server. In Figure 3 on the left, we can see problems when joining the workunits if we don't use a blending method.

The passive part of Yafrid-CORE is called the **Identificator** module whose mission consists in waiting for the communications from the providers. The first time a provider try to connect to the Yafrid server the Identificator generates an object, the provider controller, and returns a proxy to this object. Each provider has its own controller.

Provider. The provider is the software that the users who wants to give CPU cycles to be used by the grid in rendering tasks must to be installed in the computer. It can work in both visual and non-visual mode. The first thing that a provider has to do is to connect to the grid. Once activated, the provider waits until the server sends a workunit to be processed. After finishing the rendering, the provider sends the file via SFTP and tell the controller the work is done.

4. MAgArRO: Distributed Intelligent Optimization

According to [12], an agent is a computer system that is situated in some environment and that is capable of autonomous action ini this environment in order to meet its design objectives. MAgArRO uses the principles, techniques and concepts known from the area of multi-agent systems, and is based on design principles of the FIPA [21] standards.

MAgArRO is also developed using the ICE middleware [25]. The location service IceGrid is used to indicate in which computer the services reside. Glacier2 is used to solve the difficulties related with hostile network environments, making available agents connected through a rotuer and a firewall.

4.1. Architectural Overview

As mentioned, the overall architecture of MAgArRO is based on the design principles of the FIPA standard. In figure 4 the general workflow and main architectural roles are shown. in addition to the basic FIPA services, MAgArRO includes specific services related to Rendering Optimization. Specifically, a service called Analyst studies the scene in order to enable the division of the rendering task. A blackboard is used to represent some aspects of the common environment of the agents. Finally, a master service called *Master* handles dynamic groups of agents who cooperate by fulfilling subtasks.

Figure 4 also illustrates the basic workflow in MAgArRO (the circled numbers in this figure represent the following steps). **1** – The first step is the subscription of the agents to the system. This subscription can be done at any moment;

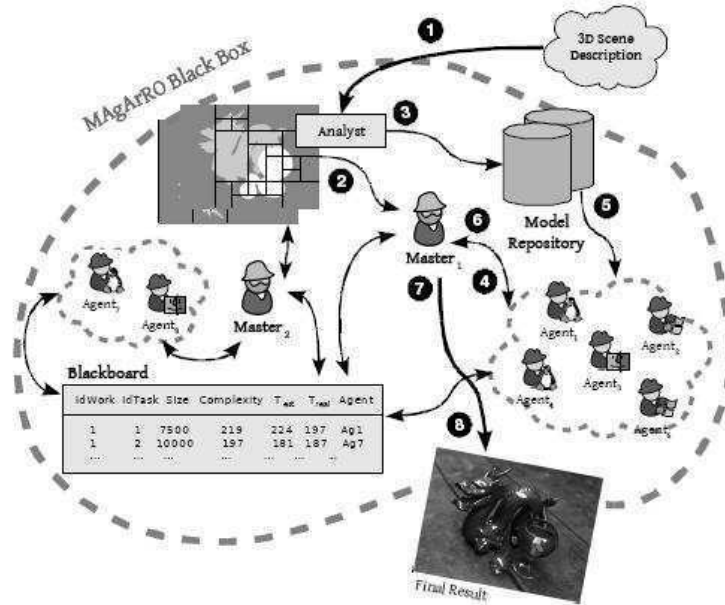


Figure 4. General workflow and main architectural roles.

the available agents are managed dynamically. When the system receives a new file to be rendered, it is delivered to the Analyst service. **2** – The Analyst analyzes the scene, making some partitions of the work and extracting a set of tasks. **3** – The Master is notified about the new scene which is sent to the Model Repository. **4** – Some of the agents available at this moment are managed by the Master and notified about the new scene. **5** – Each agent obtains the 3D model from the repository and an auction is started. **6** – The (sub-)tasks are executed by the agents and the results are sent to the Master. **7** – The final result is composed by the Master using the output of the tasks previously done. **8** – The Master sends the rendered image to the user. Key issues of this workflow are described in the following.

Analysis of the Scene using Importance Maps. MAGArRO employs the idea to estimate the complexity of the different tasks in order to achieve load-balanced partitioning. Complexity analysis is done by the Analyst agent prior to (and independent of) all other rendering steps.

The main objective in this partitioning process is to obtain tasks with similar complexity to avoid the delay in the final time caused by too complex tasks. This analysis may be done in a fast way independently of the final render process.

Once the importance map is generated, a partition is constructed to obtain a final set of tasks. These partitions are formed hierarchically at different levels,

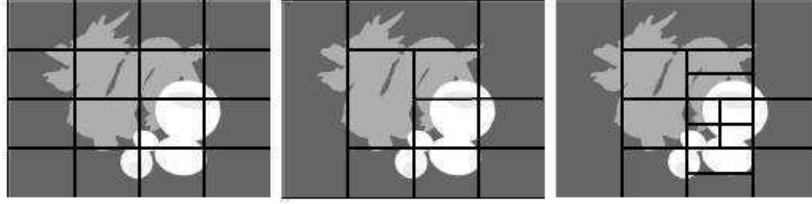


Figure 5. Importance maps. *Left:* Blind partitioning (First Level). *Center:* Join zones with similar complexity (Second Level). *Right:* Balancing complexity/size ratio (Third Level).

where at each level the partitioning results obtained at the previous level are used. At the first level, the partition is made taking care of the minimum size and the maximum complexity of each zone. With these two parameters, the *Analyst* makes a recursive division of the zones (see Figure 5). At the second level, neighbor zones with similar complexity are joined. Finally, at the third level the *Analyst* tries to obtain a balanced division where each zone has nearly the same complexity/size ratio. The idea behind this division is to obtain tasks that all require roughly the same rendering time. As shown below in the experimental results, the quality of this partitioning is highly correlated to the final rendering time.

Using Expert Knowledge. When a task is assigned to an agent, a fuzzy rule set is used in order to model the expert knowledge and optimize the rendering parameters for this task. Fuzzy rule sets are known to be well suited for expert knowledge modeling due to their descriptive power and easy extensibility [13]. The output parameters (i.e. the consequent part of the rules) are configured so that the time required to complete the rendering is reduced and the loss of quality is minimized. Each agent may model different expert knowledge with a different set of fuzzy rules. For example, the following rule is used (in a set of 28 rules) for describing the rendering parameters of Pathtracing method:

R_1 : **If** C is $\{B,VB\} \wedge S$ is $B,N \wedge Op$ is VB **then** Ls is $VS \wedge Rl$ is VS

The meaning of this rule is “If the Complexity is Big or Very Big and the Size is Big or Normal and Optimization Level is Very Big then the number of Light Samples is Very Small and the Recursion Level is Very Small”. The Complexity parameter reepresents the complexity/size ratio of the task, the Size represents the size of the task in pixels, the Optimization Level is selected by the user. The output parameter Recursion Level defines the global recursion level in raytracing (number of light bounces) and the Light Samples defines the number of samples per light in the scene (the biggest, the more quality and the higher rendering time).

5. Experimental Results

In order to test the behaviour of the systems, 8 computers with the same characteristics were connected to Yafrid and MAgArRO. These nodes (Intel Pentium Centrino 2 GHz, 1GB RAM) were used in both systems during the execution of all tests. The test scene contained more than 100.000 faces, 5 levels of raytracing recursion in mirror surfaces (the dragon), 6 levels in transparent surfaces (the glass), and 128 samples per light source was rendered using the free render engine Yafray [23]. In addition, 200.000 photons were shot in order to construct the Photon Map structure. With this configuration, the rendering on a single machine without any optimization took 121:17 (121 minutes and 17 seconds).

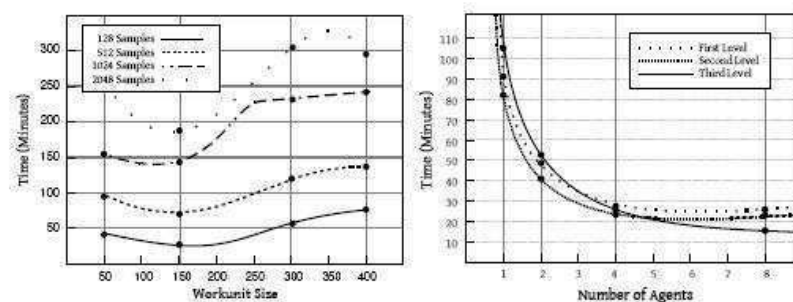


Figure 6. *Left:* Yafrid. Rendering time related to Workunit size. *Right:* MAgArRO. Different levels of partitioning with Normal optimization level.

In the case of Yafrid, as we can see in Figure 6 (*Left*), the rendering time in the best case is near seven times better using the grid and less than twice in the worst case. With these results it is clear the importance of choosing an appropriate workunit size. This occurs because there are complex tasks that slow down the whole rendering process even if the number of nodes is increased.

As we mentioned, MAgArRO uses *Importance Maps* to estimate the complexity of the different tasks. Figure 6 (*Right*) shows the time required by using different partitioning levels. Using a simple first-level partitioning (similar to the Yafrid approach), a good render time can be obtained with just a few agents. However, when the number of agents (processing nodes) grow, the overall performance of the system increases because the differences in the complexity of the tasks are relatively small.

As a final remark, note that intelligent optimization may result in different quality levels for different areas of the overall scene. This is because more aggressive optimization levels (Big or Very Big) may result in a loss of detail.

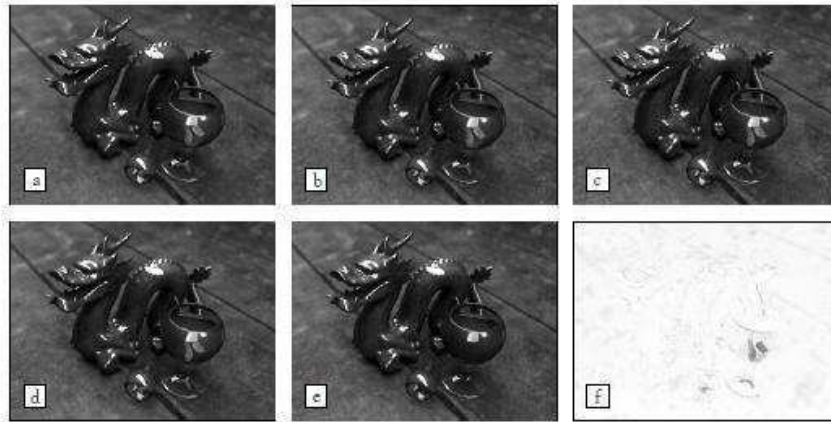


Figure 7. Result of the rendering using different optimization levels. (a) No optimization and render in one machine. (b) Very Small (c) Small (d) Normal (e) Very Big (f) Difference between (a) and (e) (the lighter colour, the smaller difference).

For example, in Figure 7.e, the reflections on the glass are not so detailed as in Figure 7.a. The difference between the optimal render and the most aggressive optimization level (Figure 7.f) is minimal.

6. Discussion and Conclusion

The computational requirements of photo-realistic rendering are huge and, therefore, to obtain the results in a reasonable time and on a single computer is practically impossible (even more difficult in the case of animations). Several approaches based on different technologies have been exposed in this paper.

Our cluster based on the **Oscar** system has some interesting characteristics:

- Very good throughput in the case of animations. The system divides each frame of the animation in different nodes of the cluster. The fine-grained approach needs the programming of new features in the main server.
- The processing nodes are used during the idle time (at night).
- The latency due to the file transfer is minimal (thanks to the use of a Fast Ethernet network).

Otherwise, the cluster can only be used submitting tasks to the main server into the same organization.

To solve some of these problems, the **Yafriad** approach was designed. This computational grid has some important advantages:

- There is no cluster; the providers can be heterogeneous (software and hardware) and can be geographically distributed.

- With the fine-grained approach, we can make local optimizations in each frame.
- One of the main advantages of this distributed approach is the scalability. The performance perceived by the user depends on the number of subscribed providers.

Some enhancements would be done to improve the performance of Yafrid. Some of them was added to **MAGArRO**:

- MAGArRO enables importance-driven rendering through the use of importance maps.
- It allows the application of expert knowledge by employing flexible fuzzy rules.
- It applies the principles of decentralized control and local optimization. The services are easily replicable, thus possible bottlenecks in the final deploy can be minimized.

There are many future research lines. In our current work, we concentrate on the combination of the best characteristics of Yafrid and MAGArRO to integrate the new system (called YafridNG) in the official Blender branch [15]. The source code of these systems, distributed under GPL license, can be downloaded from [24].

7. Acknowledgments

This work has been funded by the *Consejería de Ciencia y Tecnología* and the *Junta de Comunidades de Castilla-La Mancha* under Research Projects PAC-06-0141 and PBC06-0064. Special thanks to Javier Ayllon for his support from the Supercomputation Service (University of Castilla-La Mancha).

Referencias

1. Anderson, D.P., Fedak, G.: The Computational and Storage Potential of Volunteer Computing. Sixth IEEE International Symposium on Cluster Computer and the Grid (CCGRID '06). 73–80. May 2006.
2. Buck, I., Foley, T., Horn, D., Sugerman, J., Fatahalian, K., Houston, M., Hanrahan, P.: Brook for GPUs: Stream Computing on Graphics Hardware. Proceedings of SIGGRAPH '04, 777–786.
3. Chalmers, A., Davis, T., Reinhard, E.: Practical Parallel Rendering. Ed. A. K. Peters, 2002. ISBN: 1-56881-179-9.
4. Foster, I., Kesselman, C., Tuecke, S.: The Anatomy of the Grid: Enabling Scalable Virtual Organizations. International Journal of Supercomputing Applications 15, 3(2002).
5. Hachisuka, T.: High-Quality Global Illumination Rendering using Rasterization. GPU Gems 2: Programming Techniques for High Performance Graphics and General-Purpose Computation. Addison-Wesley Professional, 2005.

6. Kajiya, J. T.: The rendering equation. *Computer Graphics* 20(4): 143–150. Proceedings of SIGGRAPH '86.
7. Kuoppa, R. R., Cruz, C. A., Mould, D.: Distributed 3D Rendering System in a Multi-Agent Platform. Proceedings of the Fourth Mexican International Conference on Computer Science, 8, 2003.
8. Rajagopalan, R., Goswami, D., Mudur, S. P.: Functionality Distribution for Parallel Rendering. Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS'05), 18–28, April 2005.
9. Reinhard, E., Kok, A. J., Jansen, F. W.: Cost Prediction in Ray Tracing. *Rendering Techniques '96*, 41–50. Springer-Verlag, June 1996.
10. Veach, E., Guibas, L. J.: Metropolis light transport. Proceedings of SIGGRAPH '97, 65–76. New York, USA: ACM Press - Addison Wesley Publishing Co.
11. Whitted, T.: An improved illumination model for shaded display. Proceedings of SIGGRAPH '79, 14. New York, USA: ACM Press.
12. Wooldridge, M. J.: An introduction to multiagent systems. John Wiley & Sons, 2002. ISBN: 0-471-49691-X
13. Zadeh, L. A.: The concept of a linguistic variable and its applications to approximate reasoning. *Information Science*, 1975.
14. Beowulf: Open Scalable Performance Clusters. <http://www.beowulf.org>
15. Blender: Free 3D content creation suite. <http://www.blender.org>
16. BURP: Big Ugly Rendering Project. <http://burp.boinc.dk/>
17. Dr. Queue.: OS Software for Distributed Rendering. <http://www.drqueue.org/>
18. OSCAR: Open Cluster Group. <http://www.openclustergroup.org/>
19. Thin-Oscar: <http://thin-oscar.sourceforge.net/>
20. Virtual Tour ESI UCLM. http://www.inf-cr.uclm.es/virtual/index_en.html
21. FIPA. Foundation for Intelligent Physical Agents. <http://www.fipa.org>
22. Virtual Visit - Hospital Ciudad Real. <http://dev.oreto.inf-cr.uclm.es/www/vvhosp>
23. Yafray: Yet Another Free Raytracer <http://www.yafray.org>
24. Yafrid Next Generation. <http://www.yafridng.org>
25. ZeroC ICE Middleware <http://www.zeroc.com>

Servicios de información sobre la EPSJ basados en entornos de publicación

Elisabet Parras-Gutiérrez, Víctor M. Rivas y M^a José del Jesus

Departamento de Informática, Universidad de Jaén
{eparrasg,vrivas,mjjesus}@ujaen.es
<http://wwwdi.ujaen.es>

Resumen El proyecto presentado hace uso de tecnologías y aplicaciones propias del software libre tales como XML, XSL y el entorno de publicación Cocoon. A través de ellas se muestra la versatilidad que ofrecen para adaptar el contenido de un sitio web a distintos clientes, habiendo sido aplicado a un ejemplo real. El proyecto está motivado por la creciente demanda de nuevos servicios Web que obligan a extender Internet y sus servicios a nuevas formas y múltiples modos de interacción. Surgen así los entornos de publicación Web basados en XML y XSL, que combinan las ventajas de ambas tecnologías. El resultado final del proyecto ha consistido en la utilización del entorno de publicación Cocoon para servir la información de la forma más adecuada al cliente solicitante, para lo cual se han adaptado técnicas de Ingeniería del Software al problema tratado.

Palabras clave: Proyecto fin de carrera, Cocoon, XML, entornos de publicación

1. Introducción

En este trabajo se presenta el proyecto fin de carrera realizado para la obtención del título de Ingeniería en Informática en el curso 2005-2006. Está motivado por la demanda creciente de nuevos servicios en la Web surgiendo la necesidad de extender Internet y sus servicios a nuevas formas y múltiples modos de interacción que permitan acceder a los contenidos Web de diversas maneras, desde cualquier lugar, a cualquier hora y desde cualquier dispositivo.

Existen grandes diferencias entre los distintos dispositivos que pueden acceder a Internet en cuanto a la forma de presentar y recuperar la información. Estas diferencias vienen dadas por las capacidades y funcionalidades soportadas por cada dispositivo, como ancho de banda, tamaño de las pantallas, memoria o tipo de interfaces. Sumando estas capacidades tan divergentes con la personalización de contenidos, la tarea de construir aplicaciones a las que poder acceder desde diversos dispositivos se vuelve compleja. Este hecho lo constatan la inmensa mayoría de páginas web existentes en la actualidad, las cuales sólo pueden ser usadas desde navegadores web instalados en ordenadores, y siendo inaccesibles para otro tipo de dispositivos como teléfonos móviles o PDAs.

Dado que la solución no pasa por una aplicación para cada dispositivo, se hace imprescindible tener una clara separación entre el contenido o datos, la presentación de los mismos y el control de una aplicación; esto es, una arquitectura de tres capas que separe la presentación del procesamiento y a su vez de los datos. Esta separación posibilita la modificación de cualquiera de ellas sin alterar a las otras, lo que permite cierta independencia y flexibilidad.

Para solucionar esta problemática, aparecen entornos de publicación Web desarrollados mediante la filosofía del código abierto y normalmente basados en XML (*Extensible Markup Language*, Lenguaje de Marcas Extensible) y XSL (*Extensible Stylesheet Language*, Lenguaje Extensible de Hojas de Estilo). De esta forma se reúnen las ventajas del desarrollo de código abierto y la flexibilidad de la tecnología XML, la facilidad en la transformación con el apoyo de la tecnología XSL, la separación total entre datos y presentación de los mismos, la separación entre el rol del programador y el rol del diseñador (y por lo tanto más productividad, menos costos y más paralelismo de trabajo), mejor y más fácil tratamiento al mantenimiento y compatibilidad con el resto de tecnologías [1].

En este trabajo presentamos la forma en que se ha utilizado el entorno de publicación Cocoon para, a partir de los datos almacenados en una base de datos propietaria gestionada por una aplicación también propietaria, crear una estructura paralela basada en software libre y capaz de ser accedida desde distintos tipos de dispositivos.

La implementación del sistema final se ha realizado utilizando los datos y formatos del sitio web de la Escuela Politécnica Superior de Jaén¹, construyendo un sistema completamente funcional en el que se muestran las capacidades del software de código abierto tanto para la presentación como para el mantenimiento de los datos.

Por tanto, el objetivo principal de este proyecto fin de carrera ha sido desarrollar una aplicación para la Escuela Politécnica Superior de Jaén capaz de ofrecer servicios de información mediante entornos de publicación. Dichos servicios podrán ser utilizados tanto desde un navegador web convencional, como desde clientes dotados con tecnología WAP (*Wireless Application Protocol*) [3].

Este objetivo puede ser descompuesto en los siguientes objetivos más específicos:

- Analizar diferentes entornos de publicación para ofrecer los servicios de información, seleccionando el más adecuado.
- Analizar la herramienta de entorno de publicación que se va a utilizar finalmente.
- Análisis de la aplicación y servidor actuales existentes.
- Determinación de la información a publicar mediante el entorno de publicación.
- Determinación de los posibles servicios de valor añadido a implementar.

¹ Dirección de web de EPSJ: <http://eps.ujaen.es/>

- Estudiar y aprender los lenguajes XML y XSLT (Extensible Stylesheet Language Transformations, o Transformaciones XSL) que se van a ser utilizados para la aplicación.

Dado que el sistema que actualmente gestiona el sitio web de la EPSJ deseaba ser conservado por sus administradores, el desarrollo del proyecto ha debido hacerse teniendo en cuenta una serie de restricciones impuestas por el sistema actual. Así, para la realización del proyecto se ha trabajado manteniendo la BBDD existente en formato Access y la aplicación de gestión de la misma, existiendo las limitaciones correspondientes al tener que respetar dichas aplicaciones sin poder realizar ninguna modificación o adecuación para dicho proyecto. Así pues, no solo era necesario exportar los datos a XML una sola vez, sino que debía de hacerse periódicamente para poder mantener una correcta sincronización de los datos contenidos en la BBDD Access. Además, el sistema operativo que daba soporte al servidor era Windows NT, por tanto el sistema tuvo que montarse considerando dicha arquitectura.

El resto del trabajo está organizado como sigue: la sección 2 introduce las distintas herramientas que se han utilizado (en su inmensa mayoría desarrolladas bajo la filosofía del código abierto); la sección 3 describe las distintas fases del proceso de ingeniería de software llevadas a cabo para la resolución del proyecto; la sección 4 expone brevemente el producto final conseguido y destaca las ventajas que aporta sobre el modelo que existía anteriormente; finalmente, la sección 5 resume las conclusiones más destacadas de este trabajo.

2. Herramientas utilizadas

A continuación se hace una breve exposición de las herramientas que se han utilizado para llevar a cabo el proyecto. Salvo la aplicación ABC Amber Access Converter, el resto pertenecen al campo del software libre.

2.1. XML

XML o *Extensible Markup Language* (Lenguaje de Marcado Extensible) [7] es un lenguaje de marcado que puede usarse para la presentación de contenidos en Internet. De forma resumida, es un conjunto de reglas que permiten definir etiquetas semánticas para organizar un documento en diferentes partes. Creado por el Consorcio para la World Wide Web (W3C) trata de superar las limitaciones de HTML [9].

XML tiene múltiples aplicaciones; desde la simple publicación de contenidos, pasando por publicación de contenidos complejos que se adaptan al cliente en el que se va a presentar, hasta un nuevo paradigma de programación, denominada programación orientada a documentos, en el cual el elemento fundamental no es el programa, sino el documento. XML aglutina a su alrededor un grupo de tecnologías (como las hojas de estilo transformadoras, XSLT, o el lenguaje

XPath), así como entornos completos de codificación que permiten aplicar diferentes transformaciones a un documento XML hasta que se presente al usuario final.

De forma genérica, el trabajo que se lleva a cabo con un documento escrito en XML se realiza a lo largo de tres fases [8]:

1. Creación de contenidos XML: llevada a cabo por el usuario bien usando un editor de texto o mediante algún otro mecanismo más sofisticado.
2. Servicio del fichero XML: realizado por un entorno de publicación (como Cocoon), y que puede incluir la ejecución de los programas incrustados (si se trata de un XSP) en dicho fichero.
3. Creación del documento final aplicando las hojas de estilo: lo hacen programas terceros que formatean el documento al tipo de salida requerido, esto es, HTML, XHTML, WML, PDF, o XML.

La principal característica de XML es que es un estándar, no pertenece a ninguna compañía y su utilización es libre. Además es ampliable, al no estar las etiquetas predefinidas, si bien su sintaxis es estricta facilitando el trabajo de procesamiento de los ficheros. Finalmente, es un sistema independiente de la plataforma que no usa un lenguaje específico, de hecho, los documentos XML son archivos de texto plano por lo que no requieren un software propietario para interpretarlos.

2.2. XSL

XSL, *eXtensible Style Language* (Lenguaje de Estilo eXtensible), es una tecnología XML de hojas de estilos que sirve para dar formato de presentación a los documentos XML, en un documento XSL se describe un conjunto de reglas para aplicarse en documentos XML, reglas encaminadas a la presentación del documento XML en distintos formatos tales como HTML, WML, texto simple, PDF e inclusive en otro documento XML.

Programar con hojas de estilo, es la alternativa más eficiente cuando se quiere adaptar un contenido descrito con XML a diferentes clientes y la mejor alternativa cuando se quiere procesar documentos XML. Esto se debe a que separan la información (almacenada en un documento XML) de su presentación, usando en cada caso las transformaciones que sean necesarias para que el contenido aparezca de la forma más adecuada en el cliente.

El lenguaje XSL está compuesto de varias partes entre las que destacan XSLT [2] y XPath, que se describen a continuación.

2.3. XPath

XPath (abreviación de *XML Path Language*), es un lenguaje que se utiliza como herramienta de navegación para buscar elementos XML, es decir, permite buscar y seleccionar contenido en función de la estructura jerárquica de XML.

XPath obtiene su denominación por el uso que hace de una notación de caminos, como en los URLs, para navegar a través de la estructura jerárquica de un documento XML.

Todo el procesamiento realizado con un fichero XML está basado en la posibilidad de direccionar o acceder a cada una de las partes que lo componen, de modo que podamos tratar cada uno de los elementos de forma diferenciada. Por tanto, el objetivo principal de XPath es direccionar partes de un documento XML, aunque como soporte para este objetivo principal, también proporciona facilidades básicas para manipulación de cadenas, números y booleanos.

Además de su uso para direccionar, XPath contiene un subconjunto natural que puede usarse para comprobar si un nodo encaja con un patrón o no; este uso de XPath está descrito en XSLT.

XPath es a su vez la base sobre la que se han especificado nuevas herramientas para el tratamiento de documentos XML, herramientas tales como XPointer, XLink y XQL (el lenguaje que maneja los documentos XML como si de una base de datos se tratase). Así, XPath sirve para decir cómo debe procesar una hoja de estilo el contenido de una página XML, pero también para poder poner enlaces o cargar en un navegador zonas determinadas de una página XML, en vez de toda la página.

2.4. Cocoon

Cocoon ([5],[6]) es un entorno de publicación web basado en java, que soporta las últimas tecnologías del Consorcio WWW (W3C) tales como XML y XSL para proveer contenidos web. Cuenta con desarrollo total en Java por lo cual se puede ejecutar desde cualquier servidor que pueda contener Servlets; y al ser un Servlet cuenta con las ventajas de éstos, es decir, se ejecutan como hebras de forma simultánea en el mismo contexto y no tienen que llamar a métodos auxiliares como lo hacen tecnologías del estilo CGI, por tanto son más rápidos de ejecutar, y además son persistentes, o sea que no hace falta cargarlos.

Cocoon es código abierto y bastante configurable y personalizable. El proyecto Cocoon, desarrollado por una parte del equipo de Apache XML, pretende cambiar la manera en que se crea, diseña y envía la información. Este nuevo paradigma, se basa en el hecho de que el contenido del documento, el estilo y la lógica del mismo, pueden ser creados por individuos o grupos de trabajo diferentes. Cocoon propone la separación completa de esas tres capas, para que sean creadas, diseñadas y mantenidas de forma independiente, reduciendo los costes de mantenimiento, aumentando la reutilización del trabajo y disminuyendo los tiempos de producción.

Funcionamiento

Cuando un usuario hace una solicitud, la tarea que realiza Cocoon se realiza en una serie de fases que consisten en:

1. El usuario solicita un documento de cualquier tipo al servidor.

2. La solicitud se analiza para concluir si se puede atender o no. Si no se puede atender se produce un mensaje de error.
3. Si se puede atender se analiza a qué productor XML corresponde. Se genera un documento XML con el cual se trabajará.
4. Se extraen las instrucciones del XML generado en el paso anterior y éstas se le pasan al procesador apropiado para que se le apliquen al XML. Al procesar el XML podría salir un XML con más instrucciones que serán tratadas en algún otro ciclo.
5. El XML procesado se le pasa al elemento que aplica el formato. Si el documento es un documento final, XML aplica el formato y le envía el documento formateado al cliente. En el caso que el documento XML procesado sea código que deba ejecutarse, éste se pasa como productor de XML y se vuelve a procesar hasta que se llega a un documento XML final.

Estructura y Arquitectura

Estructuralmente hablando y como se ve en la figura 1, Cocoon está compuesto de:

- Productores: son los ficheros fuente de donde proviene el XML. Un productor se encarga de transformar los datos del fichero en eventos.
- Procesadores: son el proceso principal del *pipeline* y atrapan el XML de los productores para aplicarle diversos procesos, como por ejemplo, conectividad a una base de datos, transformaciones XSL, etc. El transformador más común es XSLT.
- Reactor: es la central estructural. Extrae del XML del productor las instrucciones para determinar qué procesadores actuarán en el documento.
- Formateadores: son el punto final de un *pipeline*. Recogen la representación interna del XML resultante y la preparan para enviar como respuesta al cliente en el formato adecuado.

La arquitectura de Cocoon, como se muestra en la figura 2, es como sigue:

- Núcleo: se encuentra un entorno para el control de sesiones, ficheros de configuración, manejo de contextos, aplicar mecanismos de caché, *pipeline*, generación, carga y ejecución de programas.
- Componentes: en esta capa se encuentran generadores y transformadores de XML, matchers de ficheros y serializadores para formatear los ficheros.
- Hojas lógicas: son hojas lógicas que necesita Cocoon para ficheros como sitemap, xsp, esql, request, response.
- Sitio específico de configuración, componentes, hojas lógicas y contenido: es el nivel más externo en el cual un desarrollador puede hacer configuración, creación de componentes, creación de hojas lógicas y contenido definido por el usuario de Cocoon para su aplicación

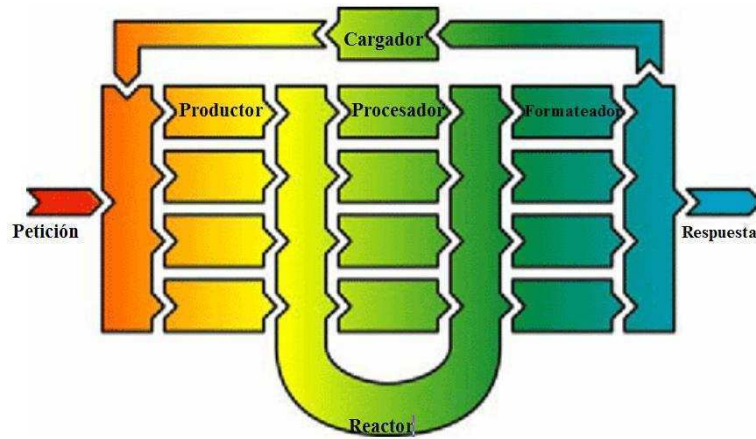


Figura 1. Estructura de Cocoon

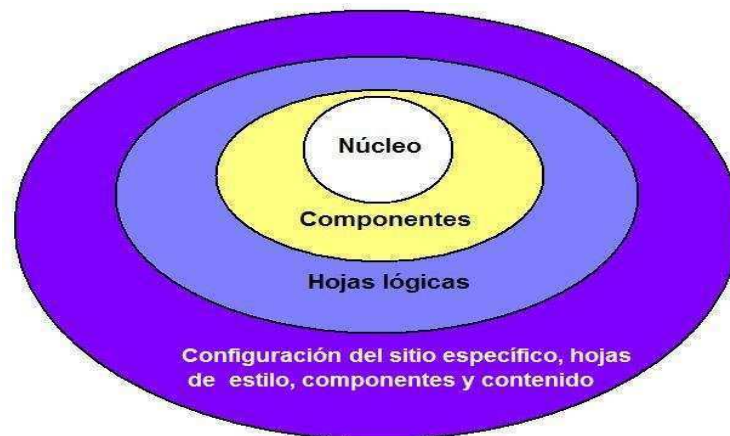


Figura 2. Arquitectura de Cocoon

El fichero de configuración *sitemap.xmap*

Los ficheros de configuración *sitemap.xmap* (o simplemente *sitemap*) indican a Cocoon la forma en que cada petición debe ser resuelta. El fichero *sitemap* tiene dos funciones básicas:

- Declarar los componentes que serán utilizados por cualquier *pipeline*.
- Declarar los *pipelines* necesarios para el funcionamiento de las aplicaciones.

Un pipeline (o tubería) es la secuencia de etapas por las que pasa una instrucción hasta ser terminada. Consiste en una entrada (por lo general uno o más ficheros XML), seguida de un conjunto de procesos de tratamiento de la misma, hasta obtener una salida. Es un concepto sencillo pero potente que hace que la programación sea más fácil y escalable. La idea es dividir el procesamiento XML en varios pasos más elementales.

Los *sitemap* deben ser jerárquicos, de modo que los directorios inferiores hereden de los superiores y definan solo lo necesario y viceversa.

El *sitemap* tiene tres partes básicas. La primera es la declaración del espacio de nombres, la segunda la declaración de los componentes y la tercera es la declaración de los *pipelines*.

Estructura del fichero sitemap.xmap

```
<map:sitemap
  xmlns:map="http://apache.org/cocoon/sitemap/1.0">
  <map:components>
    <map:generators/>
    <map:readers/>
    <map:transformers/>
    <map:actions/>
    <map:serializers/>
    <map:matchers/>
    <map:selectors/>
  </map:components>

  <map:\emph{pipelines}>
  </map:\emph{pipelines}>

</map:sitemap>
```

Es muy común tener varias aplicaciones bajo Cocoon, en cuyo caso es recomendable tener ficheros de configuración aparte para el momento en el que se debe pasar el control a cada aplicación. Esto mejora la portabilidad y la escalabilidad de los productos. Para esto Cocoon provee una herramienta que se denomina SubSitemap. Un subsitemap no es más que un fichero *sitemap* para una parte en particular de una aplicación de Cocoon. Para poder utilizar esta técnica sólo se deben tener en cuenta dos cosas:

- Incluir en el *sitemap* general de Cocoon el subsitemap (y especificar a qué y en dónde se aplica ese subsitemap).
- Incluir el subsitemap en el lugar correcto.

2.5. ABC Amber Access Converter

ABC Amber Access Converter [4] es el programa que realiza la conversión o reestructuración del formato de los datos o información contenida en una base de datos Access, para que tenga formato XML.

La principal razón por la que se ha usado ABC Amber Access Converter es porque puede ejecutarse desde línea de comandos mediante el uso de parámetros. Esto es muy útil para realizar conversiones automáticamente. Para ello hay que introducir la siguiente línea:

```
"C:\Program Files\ABC Amber Access Converter\abcaccss.exe\"
["SourceFile"] ["TableName"] ["DestinationFile"] [DestinationIndex]
```

Siendo SourceFile y DestinationFile ficheros para la importación y exportación, y TableName el nombre de la tabla MDB que se va a convertir. DestinationIndex es el tipo de conversión que se va a realizar, dado que permite conversiones a los formatos CSV, XML, HTML, entre muchos otros.

3. Proceso de Ingeniería del Software

A continuación se relatan las distintas fases que se han seguido para llevar a cabo este trabajo.

3.1. Análisis

Análisis del sistema actual

Actualmente la Escuela Politécnica Superior de Jaén cuenta con un servidor de páginas web, una base de datos Access que contiene gran parte de la información necesaria para generar las páginas y una aplicación de gestión de esa base de datos, denominada *AM Mantenimientos EPS*, realizada por la empresa AMSystem, a través de la cual es posible realizar altas, bajas y modificaciones en dicha información. El resto de páginas del sitio web son páginas estáticas e incluso enlaces a ficheros PDF. Por tanto, la información almacenada en la base de datos es utilizada por el programa que gestiona y tiene actualizados todos los datos, y por el servidor de páginas web, que es el que se encarga de servir la información al exterior a partir de la consulta de un usuario.

Toda esta información hace referencia a la Escuela Politécnica Superior de Jaén, de modo que se puede consultar información a cerca de las titulaciones impartidas en la escuela, el profesorado, tutorías de los mismos, departamentos, prácticas y asignaturas impartidas, horarios y aulas en las que se imparten, fechas de exámenes, etc.

Habitualmente, esta información es consultada por usuarios de la escuela politécnica, que en su mayoría son los alumnos y profesores de la misma, y está disponible a través de la página de la escuela en la siguiente dirección <http://eps.ujaen.es/> mediante un PC o portátil. En la actualidad, no es posible acceder a la información mediante tecnologías, como WAP, ni acomodar el contenido a otras pantallas más pequeñas. Al consultar dicha dirección obtenemos la página principal del sitio, como se muestra en la figura 3.

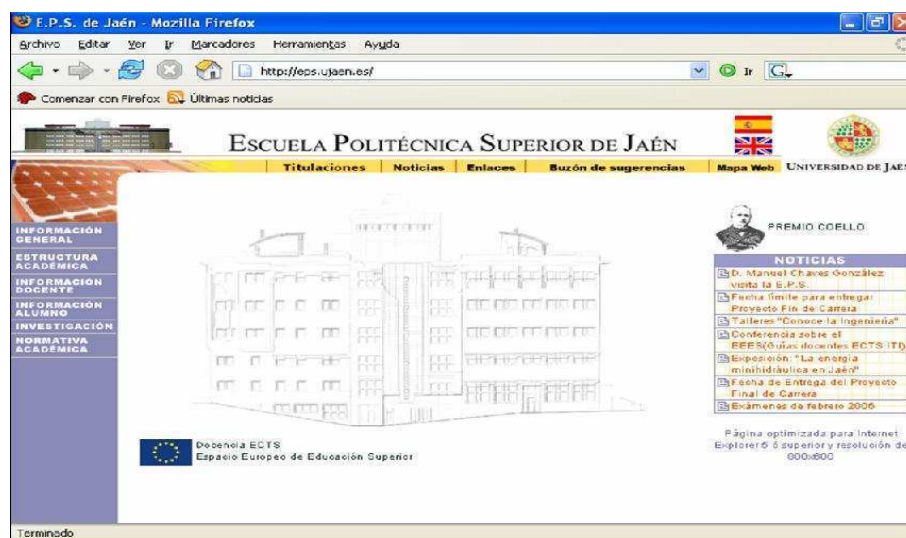


Figura 3. Pagina principal de la EPSJ

Análisis de nueva implementación

Para mostrar el uso de XML la aplicación a realizar incorpora los elementos de la web existente, pero además va a contar con un tablón formado por anuncios, en el que los alumnos podrán incorporar sus comentarios en forma de anuncios. Esta opción formará parte del menú principal del sitio web de la EPSJ, al mismo nivel que las opciones de Titulaciones, Noticias, Enlaces, etc.

Cada uno de los anuncios que formen parte del tablón estará compuesto por tres campos o apartados:

- Fecha: que indica la fecha en la que se realizó el anuncio.
- Asunto: que consistirá en un pequeño título que explique brevemente el anuncio para dar idea de su contenido.
- Texto: que será el anuncio propiamente dicho, es decir, donde se explique el tema del anuncio detalladamente.

Para que los usuarios puedan añadir anuncios a los ya existentes en el tablón, habrá un apartado en el que aportando la información necesaria se puedan enviar nuevos anuncios.

Casos de uso

Las figuras 4, 5 y 6 muestran el diagrama frontera del sistema y los casos de uso que componen el sistema.

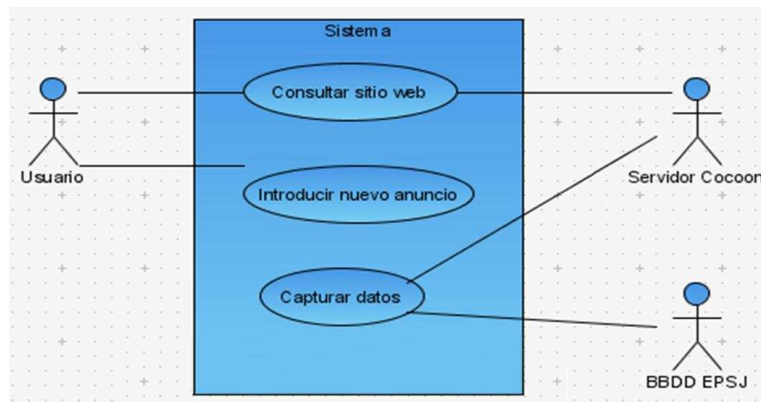


Figura 4. Diagrama frontera del sistema

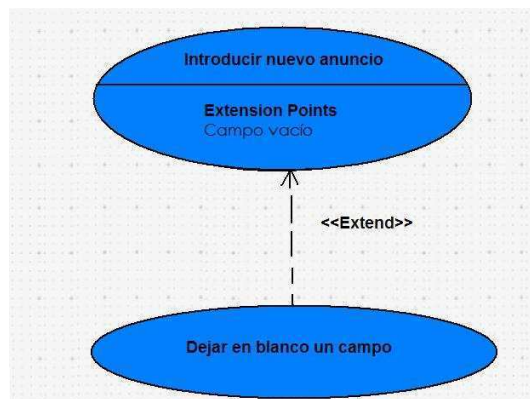


Figura 5. Caso de uso *Introducir nuevo anuncio*

El usuario interactúa con el sistema a través de dos posibles acciones: *Consultar sitio web* o *Introducir nuevo anuncio*.

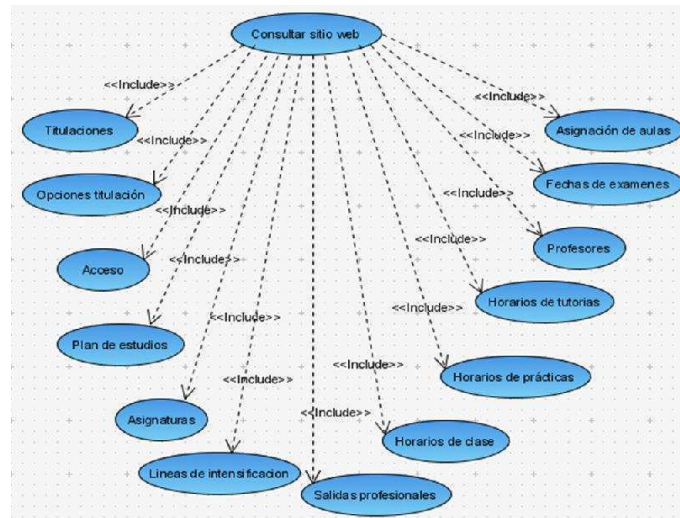


Figura 6. Caso de uso *Consultar sitio web*

Requisitos funcionales del sistema

- El principal requisito del sistema es la posibilidad de consultar los datos disponibles desde distintos clientes, es decir, se pueden hacer las mismas consultas desde clientes tan distintos como pueden ser el PC, un teléfono móvil, un PDA, etc.
- Será posible introducir nuevos datos a los ya existentes. Estos nuevos datos consistirán en los anuncios del tablón que los usuarios pueden introducir.
- Se debe poder separar contenido y formato de la presentación.

Requisitos no funcionales del sistema

Los requisitos no funcionales son los siguientes:

- Requisitos de recursos. El sistema no requiere de recursos especiales para su utilización. La información podrá ser consultada en cualquier dispositivo que disponga de conexión a Internet.
- Requisitos de interfaz con otras aplicaciones. El sistema contará con un programa (ABC Amber Access Converter) que tendrá acceso a la información de la base de datos de la Escuela Politécnica Superior de Jaén, con el que interactúa para convertir los datos que actualmente están en formato Access a formato XML. Estos ficheros generados estarán disponibles en el servidor de páginas web con acceso restringido, de donde serán capturados por el nuevo servidor cocoon.
- Requisitos de capacidad. El servidor cocoon deberá acceder a los ficheros generados en XML (antes mencionados) cada cierto tiempo, y deberá procesar estos ficheros con el objetivo de tener la información actualizada frecuentemente.

- Requisitos de documentación. La aplicación contará con un manual de la aplicación en el que se describa de forma detallada el uso del sistema.
- Requisitos de seguridad. En cuanto a la seguridad, será la que se viene utilizando en el sistema existente, es decir, nadie que no tenga permiso debería poder entrar al servidor y modificar a ningún dato que allí exista. Los servidores estarán protegidos con protocolos ssh, y también se dispondrán de medidas generales de protección como por ejemplo cortafuegos, antivirus, etc.
Los ficheros generados en XML tendrán restricciones de acceso y en general los archivos tendrán permisos de forma que, en principio, sea el administrador el que disponga de acceso total a los mismos.
- Requisitos de interfaz con el usuario.
 - En cuanto a los usuarios: debe de hacer más cómodo y flexible la consulta de información, ya que la presentación está separada del contenido en función del cliente que se utilice.
 - En cuanto al sistema operativo: el sistema debe funcionar bajo cualquier plataforma.
 - En cuanto al hardware: diferentes clientes para consultar la información, como pueden ser un teléfono móvil, un PDA, un ordenador portátil, etc.

3.2. Diseño

Diseño de los datos

En cuanto al diseño de datos, se ha trabajado respetando la estructura y el diseño de la base de datos existente, que contiene toda la información que actualmente se utiliza.

Diseño de nueva implementación

La aplicación cuenta con un tablón de anuncios en el que los miembros de la EPSJ (alumnos, profesores, PAS) pueden dejar comentarios y sugerencias. Se habilitará un espacio a través de un enlace para dicho tablón de anuncios, de manera que al acceder se puedan ver todos los anuncios que en ese momento haya en el tablón. Cada anuncio del tablón estará formado por tres campos o apartados con el siguiente formato:

- Fecha: que indica la fecha en la que se realizó el anuncio. Su formato será el siguiente: día (02), mes (09) y año (2006).
- Asunto: que consistirá en un pequeño título que explique brevemente el anuncio para dar idea de su contenido.
- Texto: que será el anuncio propiamente dicho, es decir, donde se explique el tema del anuncio detalladamente.

Además, para que el usuario pueda añadir nuevos anuncios al tablón se implementará un formulario, que estará formado por los tres campos correspondientes a las partes de un anuncio. Este formulario tiene como finalidad recoger dicha

información para poder agregar anuncios al tablón, por tanto, será obligatorio rellenar los tres campos de dicho formulario, que se corresponden con la fecha, asunto y texto del nuevo anuncio. En el caso de que el usuario deje alguno de los tres campos sin rellenar el sistema avisará al usuario mediante un mensaje de error, explicando el motivo de éste. El anuncio quedará correctamente almacenado, si todo va bien, cuando se envíe mediante algún botón de aceptar o enviar que disponga el formulario.

Diseño arquitectónico: Carta de estructura

La figura 7 muestra, a modo de ejemplo, parte del diseño arquitectónico que sigue el sistema. La programación de los módulos que la componen está dividida en dos partes: la primera, correspondiente a los módulos del nivel superior, está contenida en el fichero *sitemap.xmap*; por su parte, la segunda está recogida en módulos independientes programados como código XLST (de ahí la extensión *.xls*), que son utilizados por dicho fichero *sitemap*.

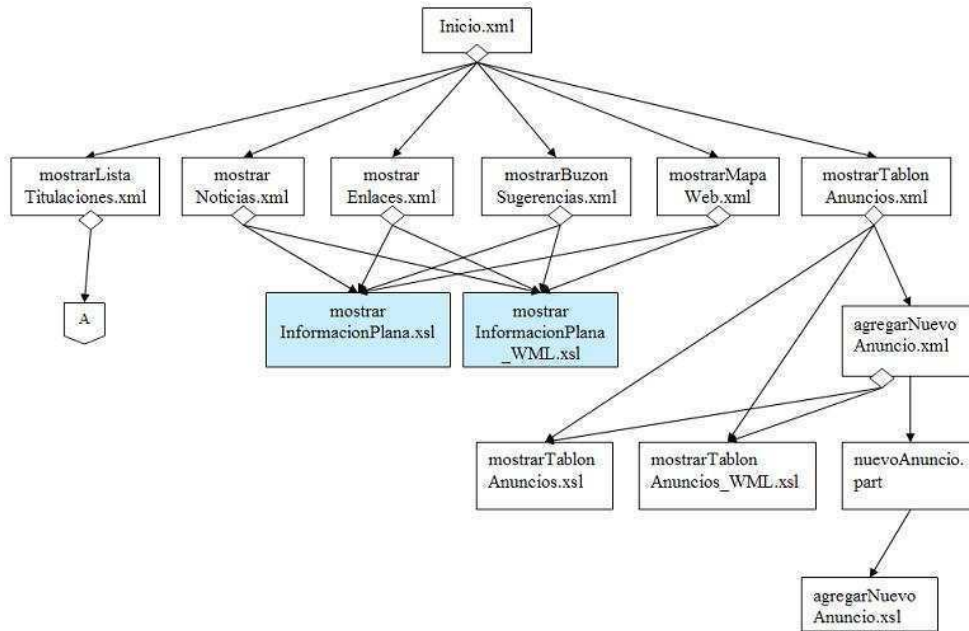


Figura 7. Carta de estructura del sistema

Destaca el hecho de que la mayoría de los módulos son compartidos a lo largo de la implementación del sistema, siendo únicamente exclusivos aquellos que se encargan de mostrar el resultado final dependiendo del cliente. De esta forma, si se desea añadir a posteriori un nuevo cliente con el que visualizar la información

del sistema, compartirá todo el proceso de recogida y transformación de los datos, centrando el esfuerzo únicamente en visualizar la información adaptada a dicho cliente.

3.3. Pruebas

Para la realización de las pruebas se ha utilizado una metodología de tipo directo mediante la aplicación de pruebas de caja negra, con las que se persigue un objetivo claro: comprobar la funcionalidad de cada una de las opciones implementadas en la aplicación, mediante el estudio de las entradas y las salidas asociadas a las mismas, sin entrar en detalle de la estructura lógica interna del software.

Los casos de prueba recogen entradas válidas e inválidas para cada uno de ellos observando si las respuestas son las esperadas en todo momento para cada uno de los casos.

4. Resultados

La aplicación obtenida cumple con todos los requisitos de partida. Por un lado, se respeta la interfaz habitual del sitio web que se está reimplementando, de modo que aquellos usuarios que acceden a través de un navegador no perciben ninguna diferencia externa en cuanto a la versión actual y la previamente existente. Además, se ha incorporado el tablón de anuncios que es completamente gestionado a través de XML, XSLT y el entorno de publicación Cocoon. Finalmente, el sitio web puede ser ahora consultado y actualizado a través de otro tipo de clientes, tales como teléfonos móviles o cualquier dispositivo que cuente con tecnología Wap. Dado que la información que se publica no está duplicada, se garantiza que los datos son consistentes con independencia del cliente en que se visualicen.

Las ventajas sobre el modelo anterior, basado en software propietario tanto para el acceso como para la modificación de los datos, es la siguiente:

- Existe una clara separación entre los datos o información y su representación.
- Dicha separación facilita su representación de múltiples formas.
- Es fácilmente ampliable, de manera que si a posteriori se quisiera incluir un nuevo cliente sería una tarea sencilla.
- El sistema no depende de ningún software propietario.
- Permite un código más ligero y el intercambio de datos es mucho más sencillo.
- Se simplifica el mantenimiento.

5. Conclusiones

En este trabajo se ha presentado un proyecto fin de carrera en el cual se ha reimplementado un sitio web basado en aplicaciones propietarias mediante la

utilización de aplicaciones y tecnologías pertenecientes a la filosofía del código abierto.

Dado que existía un software que debía seguir utilizándose, especialmente para la gestión de la información almacenada en la base de datos, el proyecto se ha desarrollado atendiendo a un conjunto estricto de restricciones, adaptando continuamente el proyecto a las imposiciones del software ya existente.

Las ventajas de la nueva aplicación se pueden resumir en que no sólo ofrece información de forma dinámica y adaptada al cliente, sino que también permite almacenar nueva información. Esto muestra la capacidad real que ofrecen las tecnologías de código abierto para el desarrollo de sistemas de información profesionales. Además, permite avanzar hacia una web semántica en la que el contenido esté totalmente dissociado de su presentación.

Por otro lado, se ha debido adaptar el proceso de ingeniería del software a las características propias del desarrollo con Cocoon y XML. Especial relevancia ha tenido el diseño de la carta de estructura en la que se ve reflejada como la lógica de la nueva aplicación se reparte entre los ficheros de configuración (*sitemap*) del entorno de publicación, y los ficheros XSLT que modifican el contenido XML.

Finalmente, destacamos que, exceptuando la aplicación *ABC Amber Access Converter* (cuya licencia tiene un coste de 20\$), todas las herramientas utilizadas son código abierto, lo cual redundará en la capacidad de introducir continuas mejoras, de abaratar su mantenimiento y de permitir futuras migraciones a nuevos sistemas que puedan surgir.

Referencias

1. Herraiz Pérez, J.A. (Tutor: Vicente Luque Centeno): "PFC: Desarrollo de un portal Web integrador de servicios de correo electrónico basado en tecnología XML". Universidad Carlos III de Madrid, 2004.
2. Doug Tidwell, "XSLT". O'Reilly & Associates, Inc., 2001
3. Huidobro, J.M.: "Tecnologías avanzadas de telecomunicaciones".^{Ed.} Thomson-Paraninfo, 2003.
4. <http://www.abcdatos.com/programas/programa/19940.html>
5. <http://cocoon.apache.org/>
6. <http://es.tldp.org/Tutoriales/APACHE-COCOON-2/multiple-html/index.html>
7. E.R. Harold, W.S. Means, "Xml: Imprescindible", O'Reilly, 2005.
8. P. Aiken, M.D. Allen, "XML in data management", Morgan Kaufmann, 2004.
9. <http://www.w3.org/TR/1998/REC-xml-19980210>

El uso de software libre en los sitios *web* universitarios españoles

Carlos G. Figuerola, José L. Alonso Berrocal, Ángel F. Zazoy Emilio Rodríguez

Grupo de Investigación REINA, Universidad de Salamanca
reina@usal.es <http://reina.usal.es/>

Resumen El desarrollo de la tecnología *web* ha convertido a éste en uno de los principales medios de comunicación de muchas entidades. Éste es el caso de universidades y otras instituciones académicas, que han hecho del *web* el instrumento probablemente más importante de difusión de información.

En el presente trabajo se analizan datos exhaustivos obtenidos de los sitios *web* de 72 universidades españolas, y se ofrece una visión detallada sobre los sistemas operativos, versiones más usadas, distribuciones preferidas; y se ponen en relación con el tamaño real de los servidores *web* soportados. Otro tanto se hace con el software servidor *web*. Los lenguajes de programación *web* utilizados son analizados desde el punto de vista de la mayor o menor penetración del software libre, al igual que los distintos formatos utilizados para los diferentes tipos de información. Igualmente, y en la medida en que hay datos disponibles, al menos para algunas universidades, referidos a años anteriores, se intenta un estudio diacrónico que muestra la evolución de varios de los elementos estudiados. Todo ello permite ofrecer una visión bastante pormenorizada sobre la difusión y uso del software libre en los sitios o sedes *web* de las universidades españolas, así como su evolución reciente.

Palabras clave: software libre, sitios *web*, universidades

1. Introducción

El desarrollo de la tecnología *web* ha convertido a éste en uno de los principales medios de comunicación de muchas entidades. Éste es el caso de universidades y otras instituciones académicas, que han hecho del *web* el instrumento probablemente más importante de difusión no sólo de informaciones administrativas, sino también de materiales docentes y resultados de la investigación. La facilidad de realización y publicación de páginas *web* ha contribuido sin duda a ello, propiciando la proliferación de servidores y sitios *web*.

De otro lado, parece de interés conocer datos acerca de la penetración y uso real del *software libre* en distintos ámbitos. Diversos trabajos han avanzado en esta línea, aplicando diferentes enfoques y metodologías; entre ellos, y por citar sólo algunos, cabe mencionar el conocido como *Libro Blanco* (2), referido sobre todo al ámbito empresarial; el informe sobre el *software libre* en Cataluña y en

España (15) o el informe REINA (7) (nada que ver con nuestro Grupo de Investigación, la homonimia es mera coincidencia) sobre el uso de la tecnología en la Administración Pública española. Algunos otros trabajos sobre distintos aspectos del *web* ofrecen también información sobre el uso del *software libre*, como el Informe sobre el *web* español de Baeza-Yates (4), o, anterior y más limitado, el estudio cibernético sobre los *webs* de algunas instituciones universitarias o relacionadas con la investigación científica (3).

En el transcurso de un estudio realizado durante 2006, subvencionado por el MEC, tuvimos ocasión de explorar de forma automática los dominios *web* de 72 universidades españolas (existen 74), prácticamente en su totalidad. Esto nos permitió recopilar abundantes datos sobre el desarrollo del *web* en las universidades españolas. El estudio cibernético citado antes, realizado en 2003, aunque mucho menos exhaustivo, también nos permite disponer de datos para abordar el uso y penetración del *software libre* en las universidades españolas.

Este trabajo está organizado como sigue: en la siguiente sección se exponen algunos problemas de orden metodológico encontrados en la realización de este estudio, al tiempo que se ofrecen los datos de tipo general en que se basa nuestro estudio. A continuación se analizan los sistemas operativos utilizados mayoritariamente, haciendo especial distinción entre los que son *soft libre* y los que no lo son. En la sección siguiente se analiza en parecida forma el *software* servidor *web*; y a continuación, los lenguajes de programación *web* empleados, para hacer, en la sección siguiente, un análisis de los formatos de ficheros más utilizados, siempre desde la perspectiva de observar el uso y penetración del *soft libre* y de los estándares abiertos. Finalmente, se extraen una serie de conclusiones.

2. Metodología

Como se ha dicho, los datos base proceden de la exploración automática del *web*. Dicha exploración se llevó a cabo mediante la utilización de un par de *spiders* de elaboración propia; cae fuera del ámbito de este trabajo la discusión de los problemas de diversa índole relacionados con el diseño y trabajo de los *spiders*; una discusión de este tipo puede encontrarse en (10). Pero sí cabe afirmar que se trata no de un muestreo más o menos amplio, sino de una recogida de datos cercana a la totalidad del *web* universitario español.

Universidades exploradas	72
Número total de hosts	6 392
Número total de páginas	4 199 081

Cuadro 1. Datos globales obtenidos en 2006

Los *spiders* recogen páginas *web*, de las cuales pueden extraerse una serie de datos de interés. Pero buena parte de la información interesante para nuestro propósito procede no de las páginas en sí, sino de las cabeceras devueltas por

los diferentes servidores en respuesta a la solicitud de las distintas páginas por parte de los clientes (navegadores o *spiders*). Estas están reguladas por el protocolo correspondiente ((8) y (14)) y, de forma opcional, ofrecen informaciones de diverso tipo, en especial en la cabecera *Server*. Hay que advertir, sin embargo, que muchas de estas informaciones no son siempre fiables; en muchos casos, simplemente no existen; y, en otros, no siempre dicen la verdad (9), por diversas razones.

De otro lado, la falta de uniformidad en algunas de las informaciones más interesantes para nosotros conducen a una dispersión difícil de tratar. Esto hace que no siempre podamos establecer determinados aspectos (por ejemplo, el Sistema Operativo del *host*). Pero la exhaustividad de los datos recolectados puede suplir estas carencias.

3. Sistemas Operativos

Una de las informaciones interesantes es la referente al Sistema Operativo de los *hosts* que conforman los *webs* de las diferentes universidades. La dispersión de datos derivada de la variedad de sistemas y versiones es grande, pero algunos grandes bloques pueden distinguirse. Igualmente, hay un número importante de *hosts* que no ofrecen datos sobre este particular, así como de máquinas que ofrecen datos demasiado genéricos. Éste es el caso de los que señalan como Sistema Operativo Unix, sin mayor precisión; este dato es importante porque podemos sospechar que muchos de ellos son Linux, aunque no tenemos constancia fehaciente; también, porque determinadas variedades de Unix no pueden considerarse *software libre*.

Los bloques de Sistemas Operativos que hemos podido distinguir pueden verse en el Cuadro adjunto. De ellos son *soft libre* todos los del bloque Linux, que es el más numeroso. También lo es FreeBSD, aunque éste tiene una presencia testimonial. También lo son, probablemente, parte de los etiquetados como Unix, sin más, pero éstos no pueden ser cuantificados.

De esta forma, si atendemos al número de *hosts* que utilizan cada variedad de sistema operativo, dejando de lado aquéllos de los que no tenemos datos precisos, podemos apreciar una igualdad entre *software libre* y propietario.

Linux	1 856
Windows	1 766
Unix	1 344
MacOS / Darwin	77
Sun Solaris	45
OpenVMS	43
OS/2	2
FreeBSD	9
Otros / sin datos	1 250

Cuadro 2. Sistemas Operativos por número de hosts

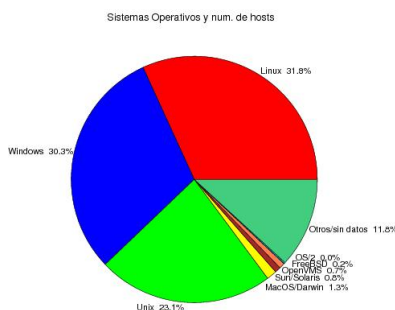


Figura 1. Sistemas Operativos por número de *hosts*

Sin embargo, si tomamos en cuenta no el número de *hosts*, sino el de páginas que albergan dichos *hosts* apreciaremos diferencias notables. La cantidad de páginas servidas en máquinas con Linux es abrumadoramente mayoritaria, y eso si no tomamos en cuenta que, como se ha dicho, parte de las máquinas son sistema operativo declarado como Unix probablemente son Linux también.

El seguidor más inmediato, el de los sistemas Windows, hospeda menos de la tercera parte que Linux. El resto de sistemas, de otro lado, tiene una presencia reducida, haciendo excepción de los *hosts* con sistema operativo de Sun; aunque siempre muy por debajo de los mayoritarios.

Linux	1 108 406
Unix	1 560 744
Windows	337 029
Sun-Solaris	64 912
MacOS-Darwin	4 634
OpenVMS	5470
FreeBSD	626
OS2	509
Otros/ s.d.	1 116 751

Cuadro 3. Sistemas Operativos por número de Páginas

Adicionalmente, siendo Linux el Sistema Operativo mayoritario, y la opción más importante dentro del *software libre* de este tipo, podemos preguntarnos acerca de las distintas variantes o distribuciones utilizadas por los servidores universitarios. Hay, como podía esperarse, también una dispersión notable, debido a la multiplicidad de versiones; pero podemos apreciar claramente que la distribución más utilizada es Debian, seguida a corta distancia de Red Hat/Fedora. Las demás se encuentran ya a considerable distancia.

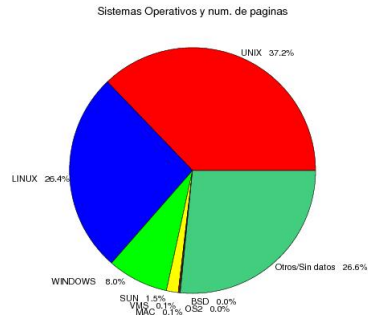


Figura 2. Sistemas Operativos por número de páginas

Debian	767
Red Hat/Fedora	721
Suse	199
Mandrake/Mandriva	79
Ubuntu	59
Gentoo	9

Cuadro 4. Distribuciones Linux por número de hosts

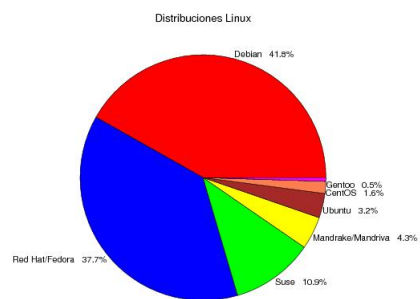


Figura 3. Distribuciones Linux y número de *hosts*

4. Servidores *Web*

Otra de las informaciones útiles que podemos obtener es el *software* servidor *web* que utilizan los diferentes *hosts* de las universidades. Como en otros elementos analizados, la dispersión de datos es importante, pero a pesar de ello podemos observar tendencias generales, así como el mayor o menor grado de implantación de *software libre* frente a propietario.

Apache	4 170
Microsfot IIS	1 425
Netscape	144
Oracle	66
Zope	63
WASD OpenVSM	43
Sun-ONE-Web-S	41
Roxen	25
Lotus-Domino	24
SAMBAR	21
AOLserver	14
Otros	356

Cuadro 5. Software servidor *web* por número de *hosts*

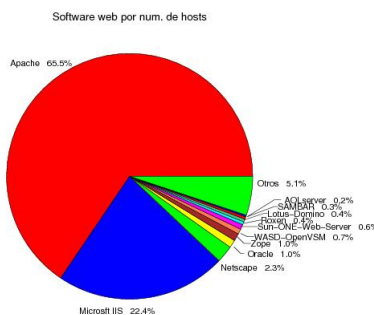


Figura 4. Software Servidor *Web* y número de *hosts*

Desde este punto de vista, cabe resaltar el dominio de Apache sobre todos los demás servidores. El más cercano, IIS de Microsoft, queda a gran distancia, y los demás a mucha mayor distancia todavía. Parece que, en lo que a servidores se refiere, las dos grandes opciones son Apache (*soft libre*) y MS IIS (propietario), con clara ventaja para el primero.

Si observamos el *soft* servidor desde el punto de vista de la cantidad de páginas albergadas, en lugar del número de *hosts*, la diferencia entre Apache y

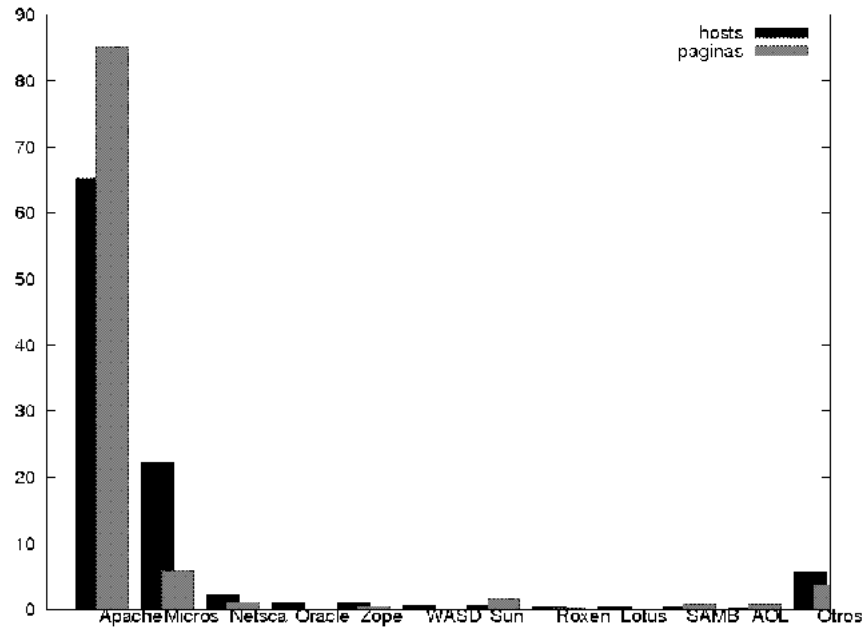


Figura 5. Servidores según número de *hosts* y páginas

Apache	3 574 881
Microsoft IIS	248 217
Netscape	47 836
Oracle	697
Zope	22 598
WASD OpenVSM	5 470
Sun-ONE-Web-S	64 912
Roxen	7 932
Lotus-Domino	766
SAMBAR	34 929
AOLserver	32 118
Otros	158 725

Cuadro 6. Software servidor *web* por número de páginas servidas

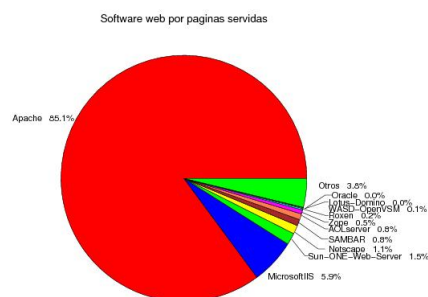


Figura 6. Software Servidor *Web* y número de páginas

MS IIS se acentúa notablemente a favor del primero. Parece claro que el grueso de las páginas que conforman el *web* universitario español está servido a través de Apache.

De otra parte, si relacionamos estos datos con los que vimos antes sobre sistemas operativos, parece claro que una parte importante de las instalaciones de Apache lo son sobre máquinas Windows.

Apache, de otro lado, tiene numerosas versiones y variantes, como es sabido. Apache 1.3 parece ser el más difundido; incluso más que Apache2; tal vez la mayor complejidad de éste último, junto con la estabilidad y buen rendimiento del primero, pueden explicar esto.

Apache 2.2	49
Apache 2.1	2
Apache 2.0	1 555
Apache 1.3	1 957
Apache 1.2	12
Apache Coyote	80
Otros	38
Sin especificar	477

Cuadro 7. Versiones de Apache

Otro asunto es el tamaño de los servidores, medido con las datos de que disponemos: el número de páginas. Aunque ya hemos comentado una visión global, que nos muestra a Apache como dominador absoluto, quizá una visión algo más pormenorizada pueda enseñarnos algo más. Para ello, hemos considerado varios grupos de servidores en función de su tamaño; a partir de ahí, vemos que, por lo que se refiere a Apache, más del 80% de los servidores que lo usan tienen menos de 100 páginas. Sin embargo, para Microsoft IIS, esta cifra desciende a casi el 60%.

Para el resto de los tamaños los datos siguen parecida tónica. Esto parece indicar que Microsoft se usa para servidores grandes con más asiduidad que Apache; tal cosa no debería extrañar, puesto que parece menos probable que alguien pague licencias de *soft* propietario para ofrecer pocas páginas.

	menos de 100	entre 100 y 1 000	entre 1 000 y entre 10 000	más de 10 000
Apache	81.01	13.6	4.36	1.03
Microsoft	59.92	28.69	9.21	2.18
Netscape	82.64	0.29	9.03	0
Oracle	96.97	3.03	0	0
Zope	69.84	20.63	9.52	0
WASD	0	81.82	18.18	0
Sun-ONE	85.37	7.32	0	7.32
Roxen	60	28	12	0
Lotus	92	8	0	0
Sambar	52.38	38.1	0	9.52
AOLServer	78.57	7.14	7.14	7.14

Cuadro 8. Software servidor *web* por tamaño de servidores (en %)

En este sentido, cabe destacar el caso del servidor WASD, cuyas pocas instalaciones parecen dedicarse a servidores de muchas páginas. Este caso, no obstante, es poco significativo, si tenemos en cuenta que este servidor se usa en sólo dos universidades; éstas parecen haber optado claramente por este servidor, puesto que contabilizan hasta 43 *hosts* entre ambas. Probablemente estamos ante máquinas relativamente potentes con varios *hosts* virtuales. WASD es un paquete para máquinas con VMS, que se distribuye mediante licencia GPL (6).

También podemos fijarnos en el servidor Roxen, usado en un par de universidades, pero con 25 *hosts*. Roxen, sin embargo, es *software libre* que corre en multitud de sistemas y arquitecturas, y que acompaña a un potente Sistema de Gestión de Contenidos (1).

Apache	67
Microsoft IIS	61
Netscape	18
Oracle	18
Zope	22
WASD OpenVSM	2
Sun-ONE-Web-S	15
Roxen	2
Lotus-Domino	8
SAMBAR	6
AOLserver	5

Cuadro 9. Software servidor *web* por universidades

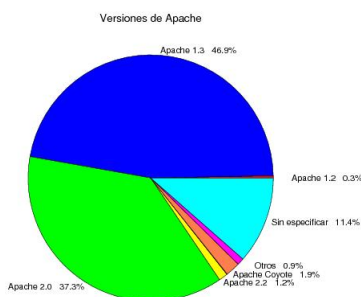


Figura 7. Versiones de Apache

Si observamos los datos recogidos en 2003 sobre esta particular, referidos sólo a número de hosts con cada *software* servidor, veremos que ya entonces el asunto se dirimía básicamente entre Microsoft y Apache, con clara predominancia de éste. Los servidores de Netscape, en sus distintas variantes, eran entonces bastante utilizados, puesto que más de un 13 % de los *hosts* lo utilizaban.

Apache	56.25
Microsoft IIS	16.86
Netscape	13.56
Oracle	0.65
Zope	0.13
Sun-ONE-Web-S	0.26
Roxen	0.13
Lotus-Domino	0.78
SAMBAR	1.84

Cuadro 10. Software servidor *web* en 2003 (porcentajes en num. de *hosts*)

5. Programación *web*

Una de las formas de aportar interactividad a las páginas *web* es el uso de *scripts* que corren en el servidor, y que reciben datos a través de formularios y similares. La tecnología clásica de este tipo se basa formularios CGI; es difícil, con los datos que tenemos, saber qué lenguajes hay del otro lado de esos CGI. En ocasiones los programas o *scripts* tienen la extensión *.pl*, pero todo el mundo sabe que Perl es uno de los lenguajes favoritos de los CGI. Los programas sin extensión, o con la extensión *.cgi* son, de lejos, los más abundantes, de manera que, por esa vía, no hemos podido avanzar demasiado.

Más moderno es el uso de intérpretes como PHP o ASP. Como es bien sabido, el primero es *soft libre* (12) mientras que el segundo es un ejemplo típico de *software propietario* de Microsoft (5).

Gracias a la formación de los URL, podemos comparar la cantidad de enlaces a *scripts* de uno u otro tipo. La diferencia es bastante clara a favor de PHP; naturalmente, puede pensarse en una asociación entre el *software* Servidor *Web* y el lenguaje utilizado, pero, en cualquier caso, la tendencia hacia el uso de soluciones de código libre es muy marcada en este campo.

	PHP	ASP
Núm. de Páginas	14 012 783	544 750

Cuadro 11. Uso de *scripting* en servidor

6. Formatos de ficheros

Las páginas *web* enlazan recursos de muy diverso tipo, y los formatos de dichos recursos son un exponente claro de la penetración o no de los formatos abiertos. Estos recursos enlazados van desde información institucional (planes de estudio, procedimientos administrativos, etc.) a material docente de las diversas asignaturas, pasando por resultados de la investigación, convocatorias, etc.

En su mayor parte, de trata de recursos que se desea difundir de forma general. Cuando no es así, se utilizan protecciones con contraseñas u otros métodos y, en consecuencia, caen fuera del ámbito analizado por nosotros. Naturalmente, tales recursos pueden haber sido elaborados con programas diversos, pero lo importante para este trabajo son los formatos con que se difunden.

En este sentido, la variedad de recursos y de formatos nos lleva nuevamente a una dispersión poco útil. Pero si nos centramos en algunos de los casos más comunes, como es el caso de los documentos de tipo básicamente textual (aunque puedan incorporar imágenes, etc.), veremos que PDF es el formato favorito, sin lugar a dudas; éste es un formato abierto (13), y lo mismo puede afirmarse del texto plano. Cosa distinta es el caso del RTF (16), que, sin ser abierto en sentido estricto, sí permite una relativa interoperabilidad. El formato del procesador de textos de Microsoft, plenamente propietario (*.doc*), queda en último lugar.

La comparación con los datos obtenidos en 2003 no es sencilla, toda vez que entonces no se recogieron datos sobre el uso de RTF, y sí sobre otros formatos, hoy de escasa significancia. Pero si tenemos en cuenta solamente los tres tipos (DOC, TXT y PDF), podremos extraer alguna conclusión. En primer lugar, el descenso de la proporción de documentos en texto plano, lo cual parece lógico, si tenemos en cuenta su pobreza visual. Más importante, la confirmación clara del decaimiento del formato propietario DOC en favor del PDF; hay razones técnicas para ello (por ejemplo, la no posibilidad de modificación de los PDF

colgados en la red), pero queremos ver también un escoramiento claro hacia el uso preferente de formatos abiertos.

Es preciso mencionar el detalle de que el número alto de ficheros RTF se debe en buena parte a una única universidad.

Por contra, recursos en formatos *Open Document* (11) son muy escasos, prácticamente testimoniales. En sentido contrario, es relativamente abundante el uso de PPT *MS Power Point*.

Por lo que se refiere a formatos de imagen, la abundancia de JPEGs era, probablemente, esperada, como también la de GIFs. Ambos formatos, sin embargo, tienen usos diferentes, por lo que tal vez sea más oportuno comparar GIF (propietario) con PNG (formato abierto). En este sentido, parece clara la preponderancia de GIF.

pdf	993 536
doc	111 437
txt	278 255
rtf	189 628
xls	7 870
ppt	16 718
opendoc	129
jpg	494 186
gif	309 358
png	26 423

Cuadro 12. Formatos de ficheros enlazados

	2003	2006
.doc	23.83 %	8.06 %
.pdf	45.32 %	71.83 %
.txt	31.35 %	8.06 %

Cuadro 13. Evolución de formatos de texto más importantes

7. Conclusiones

Se han analizado datos obtenidos tras la exploración automática de la mayor parte del espacio *web* universitario español. A través de estos datos, hemos visto que el *software libre* tiene un grado notable de implantación a nivel de servidores o *hosts*. En lo que respecta a Sistemas Operativos, Linux es el más utilizado al igual que diferentes versiones de Unix sin identificar. En *software servidor*, el dominio del *soft libre* es patente, en especial a través de Apache, así como

en programación *web* la supremacía es para PHP, la opción libre, de forma abrumadora.

Las opciones propietarias están básicamente representadas por productos Microsoft (otras opciones parecen minoritarias); mientras que del lado del *soft libre* las opciones parecen concentrarse en Linux (especialmente Debian), Apache y PHP.

Esto, por lo que se refiere a servidores. Sin embargo, en lo que se refiere a formatos de ficheros, el Open Document resulta marginal. Se trata de especificaciones muy recientes, lo cual influye en su poco uso, pero cabe preguntarse si, en general, el uso de la *suite* ofimática propietaria de Microsoft sigue siendo mayoritaria, y esto repercute en los formatos de los ficheros que se cuelgan en la red. En sentido contrario, sin embargo, el aumento de PDFs y la disminución de .doc puede apuntar a una mayor conciencia acerca del uso de estándares abiertos para difundir información, independientemente de que, para su elaboración, se haya utilizado *software* no abierto.

De otro lado, a pesar de la precariedad de datos para épocas anteriores, la comparación con lo recogido en 2003 muestra que la situación actual, en lo que a penetración y uso de *software libre* no es sino una continuación de las tendencias apuntadas entonces, en el sentido de una mayor implantación general de las alternativas libres frente a las propietarias.

Bibliografía

- [1] Roxen Internet Software AB. Roxen web server, 2007. <http://www.roxen.com/products/webserver/>.
- [2] Alberto Abella García and Miguel Angel Segovia. Libro blanco del software libre, 2006. http://libroblanco.com/joomla/document/III_libro_blanco_del_software_libre.pdf.
- [3] José Luis Alonso Berrocal, Carlos G. Figuerola, and Ángel F. Zazo. *Cibernetría: Nuevas Técnicas de Estudio Aplicables al Web*. Trea, Gijón, 2004. ISBN: 84-9704-114-3.
- [4] Ricardo Baeza-Yates, Carlos Castillo, and Vicente López. Características de la Web de España. *El profesional de la información*, 15(1):6–17, enero-febrero 2006. http://www.catedratelefonica.upf.es/webes/2005/Estudio_Web_Espana.pdf.
- [5] Microsoft Corporation. The official microsoft asp.net 2.0 site, 2006. <http://www.asp.net>.
- [6] Mark G. Daniel. WASD. The only web environment implemented expresely for VMS, 2007. <http://wasd.vsm.com.au/>.
- [7] Consejo Superior de Administración Electrónica. Informe REINA (Recursos Informáticos de la Administración del Estado), 2006. http://www.csi.map.es/csi/iria2006/iria_2006.pdf.
- [8] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. RFC 2616: Hypertext Transfer Protocol–HTTP/1.1, 1999. <ftp://ftp.rfc-editor.org/in-notes/rfc2616.pdf>.
- [9] Carlos G. Figuerola, José Luis Alonso Berrocal, Angel F. Zazo, and Emilio Rodríguez Vázquez de Aldana. Web page retrieval by combining evidence. In C. Peters, F. Gey, J. Gonzalo, H. Mueller, G.J.F. Jones, M. Kluck, B. Magnini, and M. de Rijke, editors, *Accessing Multilingual Information Repositories: 6th Workshop of the Cross-Language Evaluation Forum, CLEF 2005, Vienna, Austria, 21-23 September, 2005, Revised Selected Papers*, volume 4022 of *Lecture Notes in Computer Science*, pages 880–887. Springer, 2006.
- [10] Carlos G. Figuerola, José Luis Alonso Berrocal, Ángel F. Zazo Rodríguez, and Emilio Rodríguez Vázquez de Aldana. Diseño de spiders. Technical Report DPTOIA-IT-2006-002, Departamento de Informática y Automática - Universidad de Salamanca, March 2006. <http://reina.usal.es/pub/figuerola2006diseno.pdf>.
- [11] Organization for the Advancement of Structured Information Standards. Open document format for office applications (opendocument) v1.1, 2007. <http://docs.oasis-open.org/office/v1.1/OS/OpenDocument-v1.1-html/OpenDocument-v1.1.html>.
- [12] The PHP Group. Php: Hypertext processor, 2007. <http://www.php.net>.
- [13] Adobe Systems Incorporated. Porqué pdf?, 2007. <http://www.adobe.com/es/products/acrobat/adobepdf.html>.

- [14] R. Khare and S. Lawrence. RFC 2817: Upgrading to TLS within HTTP/1.1, 2000. <ftp://ftp.rfc-editor.org/in-notes/pdfrfc/rfc2817.txt.pdf>.
- [15] Meritxell Roca Sales. El software libre en Catalunya y España, 2006. <http://portal.uoc.edu/west/media/F-1214-1272.pdf>.
- [16] Microsoft Technical Support. Microsoft MS-DOS, Windows, Windows NT, and Apple Macintosh Applications. Rich Text Format (RTF Specifications, 2001. <http://www.snake.net/software/RTF/RTF-Spec-1.7.pdf>.

Libre software for research

Israel Herraiz, Juan José Amor, Álvaro del Castillo

Grupo de Sistemas y Comunicaciones
{herraiz, jjamor, acs}@gsyc.escet.urjc.es
<http://libresoft.urjc.es>

Abstract. Traditionally, research projects are highly opaque, showing to the public only selected deliverables, but not any internal information. No information about how the research project is working is usually available as public data. Even, among the partners of the project, no information about how the rest of the partners is working, is available neither. In this sense, research projects are similar to traditional software development projects. Research projects in the field of Information Society Technologies share some features with libre (free / open source) software projects, such as the global distributed development, and the possibility of teleworking. Because of this reason, we present in this paper a proposal to manage research projects, adopting methods used in the libre software community, and using libre software tools. Our methodology makes possible the communication flows between the different partners of the project, even if they are in different locations, and makes also possible the sharing of selected internal information with the general public. Furthermore, by adopting these methodology, several additional possibilities arise. Among them, automated and public activity reports, evolution studies of the project, and technological surveillance and prospective techniques. We strongly think that these new approach of managing research projects present several advantages over traditional organization methods, and may boost the performance of the research projects.

1 Introduction

In the scope of the 6th Framework Programme (6FP), libre (free / open) source has gained attraction, and several projects have studied or are studying the phenomenon, in order to gain knowledge and improve the development of software. There are certainly good practices within libre software projects that could be adapted to manage complex environments. We think that one of these complex environments could be research projects themselves.

In a research project, people from different countries work in a coordinated way to get the work and the goals of the project done. These people need to work on the same documents, or on the same pieces of software, without sharing the same geographical location, and have to be aware of the work of the rest of partners, in order to be efficient in the deployment of the work.

However, traditionally, research projects are highly opaque. The partners are not fully aware of what the rest of partners is doing, the general public may access only selected documents, called *deliverables*, and even some deliverables are not intended for the public.

This is a serious problem. First of all, research projects, at least in the scope of the 6th Framework Programme, are publicly funded. Therefore all the results (not only the final deliverables but all the work done in the project) should be available to those who are paying the project.

Furthermore, at least in the 6th Framework Programme's projects about libre software (and probably in other fields), several projects partially share the same goals, and need the same sources of information. These projects could gain from the other projects if they could access to the internal documents and information generated by each project. Think of the analogy with the libre software world: if a developer knows that she can reuse a piece of source code available in any other project, she can just take it and adapt for her own goal.

Because of all these issues we propose a methodology to adopt the practices found in the libre software community in the management of research projects. Our methodology is intended to be adopted by each one of the partners of the project.

The rest of the paper is as follows. Next section describes the characteristics of a typical research project. Section 3 describes the needs of a research project, and a proposal of tools that fulfill these needs. Section 4 explains how to organize the work and the environment of tools that support that work, based on the experience of our research work. Finally section 5 includes some conclusions.

2 Structure of a research project

In this section we describe the structure of a typical research project. We take as examples our experience in the participation of research projects in the scope of the 6th Framework Programme.

Research projects are proposed and developed by a set of partners coming from different countries. This arises the first problem found when working on a project: language. English uses to be the language chosen to communicate between the partners, and to write all the documents (both internal and public) of the project.

The work is divided in *workpackages*. One of the partners may lead one or more workpackages. All the partners participate at least in a workpackage. Within these workpackages, we find both *milestones* and *deliverables*. Milestones are key dates, when some work is due. Deliverables are documents (although it can be also a piece of software, a database, etc) with a part of the final output of the project. Some deliverables are public, some internal to be used by the own partners of the project, and some other are intended to be provided to the sponsor of the project (in the case of the 6FP, the European Commission).

The work needed to get the deliverables finished must be usually done by different partners in coordination.

Usually, one of the partners acts as coordination, taking care of all the economic aspects of the projects, and making sure that all the work due by each one of the partners is finished according to the workplan and in the proper dates.

The key aspect of a research project is coordination: the different partners have to work in coordination with the rest of partners, and it is very important that all the partners are aware of the work done by the others. Of course, each one of the partners must be aware of the work done by it, and the dates when the work is due.

3 Needs of a research project

To get the aforementioned work done, some tools are needed by the project. We first talk about generic concepts of what a research project needs, and then we include a proposal of libre software tools to cover those needs.

– Website

First of all, the dissemination duties of a publicly funded project should be covered by a website. It is usual to build a content management system (CMS), to make easier for the partners to publish documents, and for the general public to access to the documents.

From the Wikipedia page about CMS [1]:

A content management system (CMS) is a computer software system used to assist its users in the process of content management. CMS facilitates the organization, control, and publication of a large body of documents and other content, such as images and multimedia resources. A CMS often facilitates the collaborative creation of documents. A web content management system is a content management system with additional features to ease the tasks required to publish web content to Web sites.

This website should allow to distinguish among public and private documents, making private documents available only for selected users (typically, the partners of the project).

– Mailing list

Secondly, in order to communicate with other partners, a mailing list is needed. It is a good idea to set up two different mailing lists. One of the mailing lists would include all the people involved in the project. The other would include only the core of the group working on the project. In our opinion, some strategic decisions regarding the research project should only be discussed by a core group and not by anyone participating in the project.

If the group of people who work together is greater than 4 or 5, the need for a mailing list is clear. Furthermore, mailing lists provide other advantages such as archives of past messages, that can be useful when new members come to the group to work in the project once it has started. In this case, it

would be also a good idea to set up two mailing lists, and including all the people involved in the project and other one only with the core group. If the research group is small, it may be enough with only one mailing list.

– **Control version system**

Another need is a repository of working documents and software (files of any kind in general), with version control capabilities. This make possible to recover past version of the documents and to work in coordination with other people in the same documents. It is also a central point where anybody can find any document or file belonging to the project. This repository is not intended to publish the documents, but for the working process on the documents. The control version capabilities are crucial, because different people work on the same document, and it may be necessary to recover a past version of a document.

– **Wiki**

Another interesting tool are *wikis*, that may possible to work on documents using a web browser. From the Wikipedia page about wikis [2]:

A wiki [...] is a website that allows the visitors themselves to easily add, remove, and otherwise edit and change available content, typically without the need for registration. This ease of interaction and operation makes a wiki an effective tool for mass collaborative authoring.

Wikis allow to work on documents on the web, using only a web browser. It is intended for lightweight documents. In our opinion, it is not a good tool to write the deliverables, but it is good for the knowledge base of all the know-how of the research group.

– **Issue tracking system**

Finally, a *issue tracking system* maybe also useful. From the Wikipedia page about this topic [3]:

Issue tracking systems [...] are computer software packages that manage and maintain lists of issues, as needed by an organization. Issue tracking systems are commonly used in an organization's customer support call center to create, update, and resolve reported customer issues, or even issues reported by that organization's others employees. An issue tracking system often also contains a knowledge base containing information on each customer, resolutions to common problems, and other such data.

In the case of a research project, the tracking system can be used by the managers to assign tasks to people and other resources, and to follow the evolution of the work. This makes easier the life of the manager of the project, and allow to everybody to be aware of the work made by the rest of people in the group.

In our opinion, this is the fundamental set of tools that any group working on a research project should install. It makes easier to organize the work of the group, and it allows to keep a track of all the work done until the present day.

3.1 Tools to cover these needs

– Website

For the first requirement, a website with CMS capabilities, there exist lots of platforms available in the libre software community. A comprehensive list of libre software alternatives may be found at [5]. Most of them include the capabilities needed by a research group, such as document repository with different profiles (public, private and so on).

However, our recommendation is not included in that site. We recommend to use Plone [9].

– Mailing lists

Regarding mailing lists, we recommend to use *GNU Mailman*, which is a package for managing electronic mailing lists. It has a web interface to administrate the system, and offers the capability of archives of the list, with a web interface. More information about GNU Mailman can be found on its Wikipedia's page (see [4]).

– Control version repository

For the version control repository we recommend *Subversion* [6] (also known as SVN). The main reason is that Subversion has better integration with other tools, and can be accessed using standard Webdav clients, available in file navigators of almost all operating systems, although its better to use a Subversion client, which full support of all capabilities.

– Wiki

For wikis, in our opinion the most popular solution is MediaWiki. For instance, it is the system used by Wikipedia itself [7].

– Issue tracking system

Finally, for issue tracking systems, we recommend Trac [8]. What is even more interesting about Trac is that it can integrate a wiki, a subversion repository, a issue tracker, and a timeline for the planning of the project. For instance, when submitting a ticket, it can be associated to milestone in the planning of the project, to the different people participating in the project, to a given revision of a document in the SVN repository. The information is available in different formats besides web: text format and RSS. In particular, RSS allows an automatic treatment of the information, useful for technological surveillance and activity report systems.

There are however lots of alternatives for issue tracking systems. [10] include a comprehensive list of tracking systems, classified according to different categories.

4 Organization of the work

In this section we present how we used the mentioned tools in the previous section to cover the needs of our research projects when participating in some European projects.

First of all, this is the list of tools that we selected:

- Zope for our website.
- Mailman for the mailing lists.
- Subversion for the control version system.
- Trac for the wiki and the issue tracker. The SVN repository is integrated with Trac.

For the website, our developed its own solution, using Zope as framework. The website does not fulfill the mentioned requirements (document repository, profiles for different kind of users, etc). However, within the scope of the project, other solutions fulfilling these requirements were adopted. For instance, in some projects, Plone (which is based on Zope) was selected.

In the case of mailing lists, we have three different mailing lists for every project:

- A list where everybody working on the project is subscribed to.
- A list containing only the core group, who manages the project.
- A list where all partners all subscribed. This is useful when the trac covers only the work of a team, but the project has several teams from different institutions working on the project.
- A list of commit watchers. Every time a new commit is made to the version control system, a message with a summary of the commit is sent to this list. This allow everyone to be aware of the changes made to the repository.

For the mailing lists we use Mailman. The lists are usually configured as moderated for unsubscribed people, to avoid junk emails. Some lists, such as core or partners lists, could be also configured as private (nobody can subscribe or read the archives without authorization).

For the wiki and the issue tracking system, we adopted Trac. We integrated also the Subversion repository in Trac. We use the wiki for the knowledge base of the project, and the tracking to control, assign and monitor the work in the project. Also, the electronic mails generated by this tool (when issue tickets are created or closed) are sent to the list used by the working team.

When managing several projects at a time, each one with its own trac, it is very useful to integrate the activity tracking of all projects in a “planet” (a RSS aggregator ¹). The planet is very useful to see the recent activity of all projects in a web page, importing all RSS files which represent the timeline contents of each trac site.

Our team has modified Planet in order to integrate also *activity indicators*. An activity indicator is a funny face which represents the recent activity of a

¹ The most used, written in Python, is available at <http://www.planetplanet.org/>

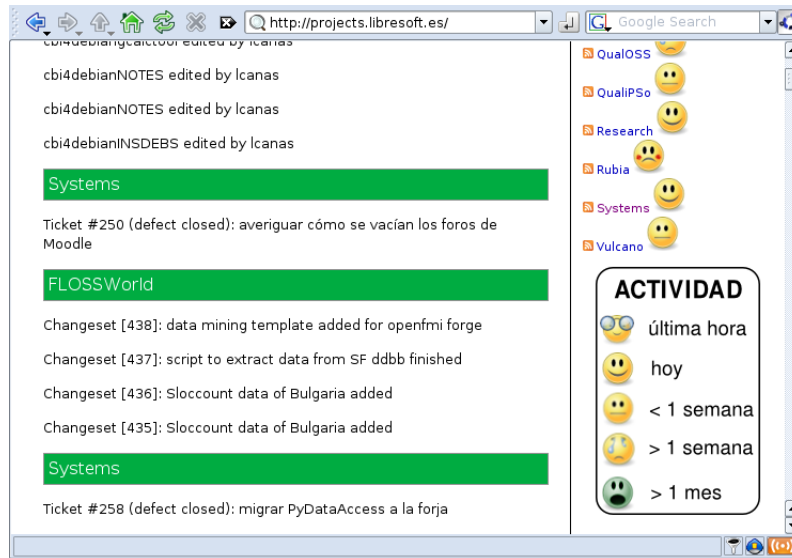


Fig. 1. Planet website, with recent activity in the different projects, and indicators about this activity

project. For example, if a project has registered activity in last hour, the face is showed laughing. But when the activity is registered only in last day, the face is showed more serious. There are several faces until the worst case, which represents the project when it has not registered activity in last month.

An example of this website is shown in figure 1. On the left part we find a list of recent events, classified by project. On the right part, we see a list of all the projects, with the indicator of the recent activity. Below the list of projects, a summary of activity indicators is shown.

The RSS feed from the Trac tool from different projects are integrated into one only website. This feed contains a entry for each event happening in the Trac. An even may be a ticket event (created, changed, etc), an event in the wiki (modification, adding or removal of a page), or an event in the Subversion repository (again modification, adding or removal).

This website has resulted to be very useful for the group. First of all because it allows to anyone working in the group to be aware of the recent work done in all the projects, and who did it. Secondly, because the activity indicators act as a “motivator” for the different subgroups working on each project. For instance, if one of the project is taking the lead in the activity indicators, other group may want to encourage their work in order to recover the leadership.

Summarizing, we have implemented all the mentioned tools. We obtained the full potential of using these tools when we integrated all of them. For instance, our Trac websites integrate wiki, Subversion repository and issue tracking system. Moreover, we have a mailing list which receives a message every time

a change happens in any of the repositories. As we are working on different projects, we join the information about the recent activity of these projects in one only website. This allow to everybody to be aware of the recent work done by the rest of the group, regardless which project they are working on. Furthermore, activity indicators resulted to be motivators to keep the activity level high, when comparing to other projects within our own research group.

However, we have to admit that due to external requirements, we can not fully open our tools to the rest of the world yet. So we are not taking advantage of sharing our knowledge with the rest of partners of the different projects. We are making efforts to achieve so though.

5 Conclusions

We present in this paper a methodology and a set of tools to organize a research project and the different groups working on the project. Our methodology is based on the methods and tools used to manage and organize libre software communities.

Research projects should be as open as libre software project, because of two reasons: they are usually publicly funded and so they should publicly available to anyone, and some projects may get benefit of collaborating with other research projects, making a more efficient use of the public funding.

The proposed methodology allows to make all the information publicly available. Not only the final deliverables but all the work done during the lifetime of the project.

The proposed tools allow to keep a track of all the work done during the whole lifetime of the project. These repositories of information about the research project open new directions to improve the efficiency of the research projects. For instance, automatic technological surveillance of research projects, based on the trails available in the repositories of the project (website, mailing list, version control system, issue tracking, etc).

In the other hand, the proposed tools and methods also allow to filter the information to the users and the public based on different profiles to access the information as well.

Another strong point of this methodology is that it makes possible to work remotely. As all the information is managed using the proposed tools, and all the tools can be available remotely. This would make possible to continue working when people is visiting other partners or universities. It allows also the coordinated work among different partners (they are usually located in different countries).

The only drawback of our proposal is that is only valid for Information and Communication Technologies. For instance, chemical or biological projects require people to work together in the same location. Although the tools may be used to organize the work anyway.

In further work we will use the trails of the repositories of the projects where we are working on to build a platform of technological surveillance of the research

done on libre software. We are planning also to build tools to automate activity and participation reports, based on the information provided by the repositories. In the near future, we are considering also to fully open our repositories, to make the information available to anyone. Right now, as we are working with other partners, that decision is in our hand. Anyway, all the results of our projects are offered under non-restrictive licenses, both for software and documents.

References

1. http://en.wikipedia.org/wiki/Content_management_system
2. <http://en.wikipedia.org/wiki/Wiki>
3. http://en.wikipedia.org/wiki/Issue_tracking_system
4. http://en.wikipedia.org/wiki/GNU_Mailman
5. <http://www.opensourcecms.com>
6. [http://en.wikipedia.org/wiki/Subversion_\(software\)](http://en.wikipedia.org/wiki/Subversion_(software))
7. <http://en.wikipedia.org/wiki/Mediawiki>
8. <http://en.wikipedia.org/wiki/Trac>
9. <http://plone.org>
10. http://en.wikipedia.org/wiki/Comparison_of_issue_tracking_systems

JavaVis: open source code for computer vision subjects^{*}

J.M. Perez, B. Bonev, P. Suau, D. Viejo y M. Cazorla

Dept. de Ciencias de la Computación e Inteligencia Artificial
University of Alicante, P.O. Box 99, E-03080 Alicante
{jmperez,boyan,pablo,dviejo,miguel}@dccia.ua.es

Resumen In this paper we present the open source project JavaVis, oriented to Computer Vision teaching. It consists of a framework with several features that make it useful for that purpose. Some of them are: a) image format, supporting frames and bands for sequence processing, b) code with many algorithms available, it contains implementations of some of the most widely used algorithms in computer vision, c) 3D data support, d) a visual desktop for visualizing partial results. It was designed to be easy to use: the user does not have to deal with internal structures, and adding a new algorithm is a simple task. J

We have developed three different modules, based on three different needs we have noticed in our subjects. The first one is a basic library for image processing. Besides the previously commented features, it supports geometrical data (edges, segments, points, etc.). The second module is based on the same working schema as the first one, but applied to 3D data. These two modules are enough for testing many well-known algorithms. They also suit the programming needs of students and teachers, as they can easily develop their own algorithms for the JavaVis framework. All JavaVis functions can be launched both from command line, as well as with the JavaVis Graphical User Interface. Finally we have extended JavaVis with a third module consisting of a visual desktop where different Computer Vision functions can be easily placed and connected. Its purpose is to visualize intermediate results in processes involving several functions, helping their better understanding.

JavaVis, as its name suggests, is implemented in Java.

1. Introduction

Computer vision is an important subject in computer science degrees. For laboratory lectures, we need a tool that is complete and easy to use. In this work we present a Java library which is oriented to teaching. This means that we have designed and built the library thinking in readability and understanding instead of efficiency.

^{*} This work has been partially supported by project GV06/134 from Generalitat Valenciana (Spain).

We have developed three different modules, based on three different needs we have discovered in our vision related subjects. The first one is a basic library to process images. It is the oldest one and it follows a special file and image format enabling easy sequence processing; it incorporates geometrical data (edges, segments, points, etc.) and it has implemented several well-known computer vision algorithms. The students can, easily, develop their own algorithms. The second module is based on the same working schema than the first one, but applied to 3D data. Finally, we have developed a visual desktop in order to visualize how the different algorithms work and the results of combining them.

Several libraries and systems have been developed for computer vision research and education. First of all, we will describe libraries written in any programming language, and we will conclude with Java libraries. Khoros library [7] is available for Unix platforms, Windows and MacOS. It incorporates a visual programming environment where programs are created by placing toolboxes: rectangular icons that represent operators, which are simply stand-alone programs written in C, C++, Java or a script language. Each operator performs on an input image or dataset, producing an output image. Connections that represent data flow between the toolboxes are created by clicking the mouse. To complete the visual programming capabilities, there are advanced programming language constructs such as loops, procedures and control structures. This library is currently not freeware nor open source. Open Computer Vision Library (OpenCV) [8] is available for Windows and Linux. It is distributed under Intel's licence for both commercial and non-commercial (research and teaching) purposes. The library includes over 300 image analysis and processing methods from morphology, geometry, image treatment, etc., up to the recently added methods for computing stereoscopic correspondence, face recognition or 3D tracking. Although OpenCV is one of the most complete and efficient computer vision libraries, it is not portable enough for other operating systems. Also, OpenCV does not incorporate a Graphic User Interface (GUI) to visualize and evaluate results. VIGRA [9] is a novel computer vision library which focuses on customizable algorithms and data structures. The library was built using generic programming as indicated by the Standard Template Library (STL). By writing a few adapters (image iterators) it is possible to use VIGRA's algorithms in computer vision applications. Nevertheless, the library does not have enough implemented algorithms, nor does it have a GUI.

On the other hand, there are a few libraries written in Java. Java Imaging and Graphics Library (JIGL) [4] was developed at Brigham Young University to make programming both course-level and research-level image-handling algorithms as easy as possible. JIGL extends standard Java image-handling capabilities. It is based on the Java Advanced Imaging (JAI) [3] development by Sun Microsystems. JAI is a base library which focuses primarily on web-based applications. Nevertheless, JAI is not useful for computer vision applications. This is why JIGL was developed. It is built around a set of image classes that support individual pixel access, image-wide operations and image-image operations. Nevertheless, the number of included computer vision algorithms is low.

Image Processing in Java (IPJ) [10] is another JAI-based library. The IPJ goal is to expand JAI's functionalities with computer vision algorithms. The methods included in the library are spatial filters, convolutions, compression, morphological filtering, boundary processing and chromatic light. Nevertheless, some of the more elaborated methods begin in lack as, for example, sequence processing, object tracking, 3D stereo vision, etc. Furthermore, IPJ does not have templates for adding new algorithms to the library, nor does it have a GUI. Java Vision Toolkit (JVT) [5] is based on the JAI library, adding computer vision algorithms for 2D and 3D images. JVT provides implementations for image-handling filtering, edge detection, segmentation, Hough transform, morphology and colour analysis. It also includes a GUI application, which makes JVT easy to use for end-users and developers alike. JVT is designed for students using the image-computation template provided to implement new algorithms. The main problem with JVT is the lack of sequential image handlers. ImageJ library [2] is based on three fundamental features. First, the use of macros, which allow users to automate tasks and create custom tools; second, it is possible to extend the capabilities of ImageJ by developing plug-ins; and finally, we can process an entire stack of related images using a single command. The library core includes several image handlers. All the related computer vision functions are incorporated as plug-ins. Also, ImageJ contains a GUI which allows end-users to launch the library algorithms and evaluate the results. This library is widely accepted in scientific areas, although the lack of templates for the development of new algorithms makes it less appropriate for educational purposes.

The rest of the paper is organized as follows: In Section 2, we describe the main features of JavaVis. Section 3 explains how the 3D GUI works and Section 4 explains the JavaVisDesktop. Finally, some conclusions are drawn in Section 5.

2. JavaVis

JavaVis was designed as an open source tool for computer vision teaching. This section will introduce its main features. We began to develop JavaVis following the same philosophy of work than Vista [1]. Vista was a computer vision library written in standard C which tried to simulate the object-oriented programming. The features of Vista were very attractive for developing algorithms in vision and, in fact, many researchers used it, but its creator left the project. Our objective was to create a similar library with a totally OO language. Most of the features of JavaVis are based on the operation of Vista.

In JavaVis we have defined both a file and an image format that allow to work with image sequences. Some computer vision algorithms have to manage image sequences. For example, optical flow algorithms use several images to estimate movement and stereo computation needs two or three images to work. Furthermore, images are usually composed by different bands, for example, a RGB color image can be stored in three different bands. In this way, JavaVis handles its own image format that enables it to manage image sequences and bands.

Each image in JavaVis is composed of one or several frames. A frame represents an image that can be of two types: bitmap or geometric. A bitmap frame is an image represented as a matrix in which each element is a pixel (picture cell). JavaVis has five types of bitmap frames: BIT, BYTE, WORD, REAL and COLOR. A brief explanation of frame types is showed in Table 1. In addition, each bitmap frame can be formed by one or several bands.

Cuadro 1. Bitmap frame types in JavaVis

Type	Description
BIT	Defines a binary image where a pixel value may be 0 or 1
BYTE	It represents the classic gray scale image, with pixel values in the range [0, 255]
WORD	Uses a range of [0, 65525]
REAL	It is the only format which allows negative values, useful in convolution and gradient computations. It is represented using the float type of Java
COLOR	Color image (R, G, B) formed by a three bands image, each band being of <i>BYTE</i> type

On the other hand, a geometric type frame manages information in a different way. For example, the *SEGMENT* frame type just stores the coordinates (x and y) of the initial and end defining points of a segment. A *SEGMENT* type frame may have several segment objects stored not into a pixel matrix but into a segment object list. So the representation of geometric frames is more compact and computations are faster. The available geometric types are showed in Table 2. This kind of frames are useful in several computer vision algorithms.

Cuadro 2. Geometric frame types in JavaVis

Type	Description
POINT	The simplest type, a 2 dimensional point.
SEGMENT	It represents segments. A segment consists of an initial and an end point.
EDGES	It represents a set of points forming an edge. They keep an adjacent relation, i.e. the first point is adjacent to the second, the second to the third and so on. Adjacency between the last point and the first one is not necessary.
POLY	It contains a set of points forming a polygon. The points form a circular sequence.

A sequence can be organized in two ways: several frames in a sequence, or several bands per frame, which can also form a part of a sequence. However, in order to insert several bands in the same frame, every band must have the same size and be of the same type whereas different frames can have different size, type and number of bands. In Fig. 1 we show an example of an image in JavaVis. Finally, we have developed a special file format to store all information from our image format. It's called JIP format or JIPZ for compressed files in order to save disk space. JavaVis can read both JIP and other file formats such as JPEG or GIF, converting them into its own image format to work with them. For storing source images and computer vision algorithm results the JIP or JIPZ formats can be used, an exporting to JPEG is also allowed.

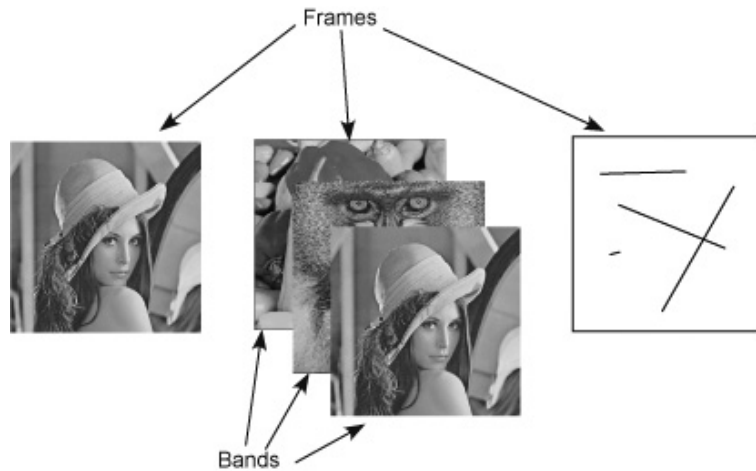


Figura 1. A sequence in JavaVis having three frames. The second one is formed by several bands and the third one is geometric.

An important point in JavaVis is the organization of its functions (implementation of algorithms), as it is important for other users to be able to easily implement their algorithms in a standardized way. A function in JavaVis is a Java class which implements an algorithm or method. A function inherits from an abstract class JIPFunction. In order to implement an algorithm, only the function code must be developed and input and output parameters specified. When a function is executed, neither the user nor the programmer has to check the parameters; JavaVis does it automatically.

There are three ways to execute a function. The first one is from command line: the Launch class can execute a function of the library, after the function name and its parameters are specified. This class checks the values entered (name, type and range of the parameters) and returns an error if something is wrong. Otherwise, the function is executed. The execution takes an input file and generates an output file. The output file has the same sequence as the input file,

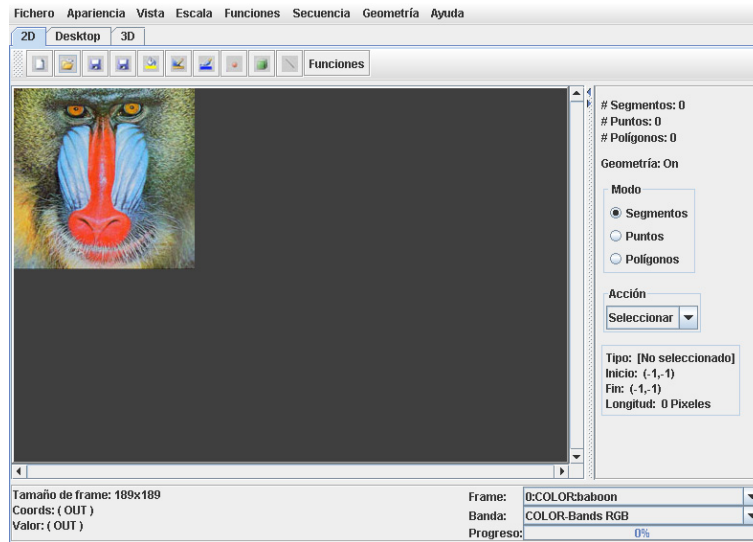


Figura 2. Graphic user interface used to visualize images and to apply functions.

where every frame has been processed with the function. The second way to execute a function is from the graphical user interface (see Fig. 2). A function can be executed by introducing the input parameters in a window and the parameters are checked too. Furthermore, the function can be applied to all the sequence or just to the current frame. The final way to use a function is from another function. Each function can use any existing function defined in the library. Note that despite the possibility of executing a function in three different ways, it is only defined once, and there is no need of additional configuration of the JavaVis environment but just adding the class for the function in a directory.

Another important issue regarding functions is the definition of parameters. A standard is defined for both input and output parameters for JIPFunctions. This is done by means of a class JIPParameter. Thus, the GUI can set and get parameters for any implemented JIPFunction.

Finally, regarding function implementation, it must be noted that Matlab linking is possible for intermediate results, as functions for transforming an image to a Matlab matrix are provided. JavaVis is mainly used in academic and research areas, so linking with other mathematical and Computer Vision related programs is a very usual situation.

Table 3 lists some of the functions already included in the distribution of JavaVis. Many of them are implemented by undergraduate students who used JavaVis for their projects and needed to implement new functions. The maintainers of JavaVis have revised their work before including it into the JavaVis distribution.

Cuadro 3. JavaVis JIPFunctions

Group	Functions
Transform	They allow image format conversion: FColorToGray , FGrayToGray , FGrayToColor , FRGBToHSB , FRGBToHSI , FRGBToYCbCr , FBinarize .
Adjustments	Image adjustment: FBrightness , FContrast , FGamma , FEqualize , FSharpen , FSharpenMore .
Smoothness	Image smoothness, reducing noise: FSmoothAverage , FSmoothMedian , FSmoothGaussian .
Convolution	Implement several ways to convolution: FUser3x3 , FUser5x5 , FConvolveImage , FConvolveAscii , FGabor .
Manipulation	Manipulate the image, rotating it, scaling it, and so on: FOpenWindow , FSkew , FFlip , FFuzzyKmeans , FKmeans , FMirror , FScale , FRotate , FCrop , FWaveHoriz , FWaveVert , FNegate , FNoise , FMaximum , FMinimum , FPixelate .
Geometry	Apply functions to the geometric data: FAddPoint , FAddSegment , FRandomPoint , FRandomSegment , FInterSegment , FGeoToGray , FHoughCirc , FHoughLine .
Edges	Calculate and operate with edges: FCanny , FLink , FSegEdges , FNitzberg , FSusan , FGrad , FMag , FPhase .
Math Morph	Implement morphological operators: FErode , FDilate , FClousure , FOpening .
Applications	Functions which use other functions and implement complex application: FCountCoins .
ImageBD	Makes image Data Base searching: FCalcHistoColor , FCalcHistoBD , FSearchImage , FSOM .
Ring Projection	Manages omnidirectional images: FRectifyOmnicam , FCleanOmnicam , FRingCreateMask
Other	Functions non included in previous groups: FHistogram , FSkeleton , FOp , FSegmentHSB , FBlobs , FManhattan , FCapture , FAnnealing, FPca , FPcaRecon, FFlow .

3. JavaVis3D

3D data is a special data type formed by a set of points where each point is described by 3 coordinates (usually X , Y and Z) and, sometimes, a gray or color level associated to this point. These data come from a stereo camera or a 3D laser. We have realized that others computer vision libraries do not manage 3D data and we think it could be useful to incorporate a tool for managing them.

Java provides a 3D API: Java3D. We have used this API in order to incorporate a GUI for showing 3D data (see Fig. 3). The GUI is built on a typical 3D scene in Java3D. Data is shown as points and they can be rotated, zoomed

in and out, and translated. We can also change several properties like aliasing, color and point size.

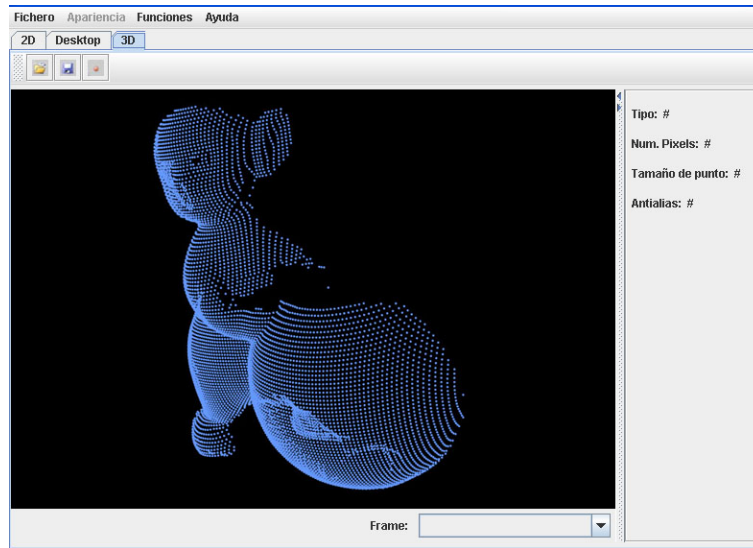


Figura 3. JavaVis 3D Gui.

File format is different from images. 3D data are represented by a list of coordinates and, if available, color information. File has a header with metadata information and then the list of points, one for line. It supports several frames (a sequence) like images, but not bands. Functions follow the same schema than previously: they receive and return an 3D data object. Several additional classes and methods have been implemented for data manipulation.

4. JavaVisDesktop

Another new feature we have recently incorporated to JavaVis is JavaVis Desktop (see Fig. 4). The goal was to build a tool to allow a better understanding of partial results when processing an image. Different algorithms may be used in order to build more complex routines. Fig. 5 shows an example of an algorithm that counts the number of coins in an image. The algorithm is built using several already implemented algorithms, like Hough transform [11] and Canny algorithm [12]. The Desktop utility allows us to check partial results, showing the images obtained applying an algorithm.

This tool allows to join a sequence of functions, like an automata. Each node in the sequence is a function (algorithm) from JavaVis. Each node shows the result of applying its algorithm as a thumbnail image. The full-size result of

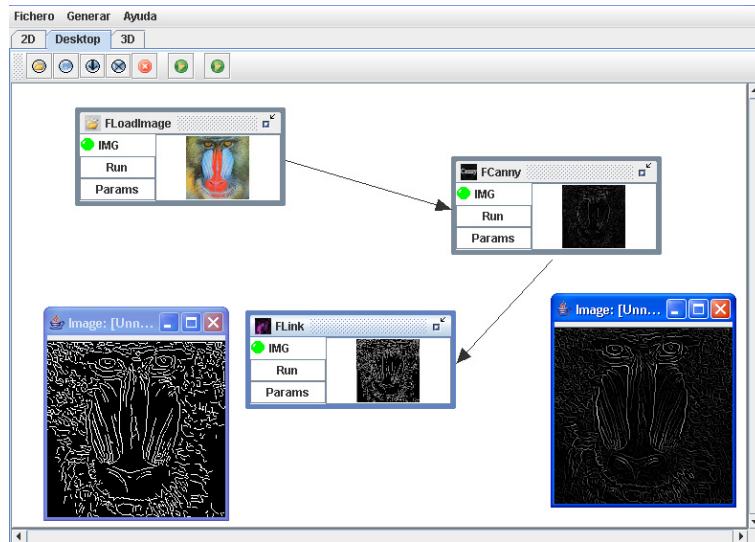


Figura 4. JavaVis Desktop.

each step can be seen by clicking on the thumbnail. The result is only available if the function has been successfully executed. Each node includes a button to execute its function, as well as a button for setting its specific parameters.

This new routine created as a sequence of other algorithms, can be easily added to JavaVis as a new function, with a button on the Desktop interface. Thus, the user can run this new function from the JavaVis interface, from command line, or to use it from a java program. Also this is a way to facilitate the sharing of new Computer Vision (CV) algorithms. The aim is to impulse the community to incrementally construct the CV library. Many CV algorithms are easy to implement if some basic functions are already developed, as seen in the previous coin counting example. Also, the sequence of function, together with the parameters of the functions is saved in XML format, for further editing or running it in the JavaVis environment.

5. Conclusions

In this paper we have introduced our open source framework for teaching computer vision written in Java: JavaVis. Its features include a new image format capable of handle images and geometrical data, with multiple bands and frames for sequence processing, several launching methods (including graphical interface) and a large quantity of implemented computer vision algorithms. We have described the recently new modules added to JavaVis: the desktop, useful to understand how final result is obtained from partial results, and Javavis3D to support 3D data. This tool have been used during past years in computer vision subjects in University of Alicante. Open source makes easier some tasks



Figure 5. Two examples of counting coins. The white circumferences are the coins detected. Note that the CD is not detected as it has a different color, and the inner circle of several coins is not detected too.

needed during our teaching experience: students can examine the code and see how algorithms are implemented, and as a consequence, how they work; and also, new functions can be easily added.

Our framework is under continuous development and improvement. New vision algorithms implemented by students and revised by teachers will be included. Currently we are developing a new version of JavaVis. It will be available by march 2007. Connectivity with Matlab could also be improved.

Referencias

1. 2006. The Vista website. <http://www.cs.ubc.ca/nest/lci/vista/vista.html>.
2. 2006. The imagej website. <http://rsb.info.nih.gov/ij>.
3. 2006. The java advanded imaging website. <http://java.sun.com/products/java-media/jai>.
4. 2006. Java imaging and graphics library. <http://rivot.cs.byu.edu/jigl>.
5. 2006. The java vision toolkit website. <http://marathon.csee.usf.edu/~mpowell/jvt>.
6. 2006. Javavis web site. <http://javavis.sourceforge.net>.
7. 2006. Khoral. <http://www.khoral.com>.
8. 2006. Open source computer vision library. <http://www.intel.com/research/mrl/research/opencv>
9. 2006. The vigma library website. <http://kogs-www.informatik.uni-hamburg.de/~koethe/vigma>.
10. Lyon, D. 1999. Image Processing in Java. Prentice Hall.
11. Duda, R. and Hart, P. "Use of the Hough transformation to detect lines and curves in pictures" (1972) Communications of the ACM, 15 (1), pp. 11-15
12. Canny, J. "A computational approach to edge detection". (1986) IEEE Trans. on Pattern Analysis and Machine Intelligence, 6 (8)

El reto de los servicios Web para el software libre

J. J. Domínguez Jiménez, A. Estero Botaro, I. Medina Bulo,
M. Palomo Duarte y F. Palomo Lozano

Depto. de Lenguajes y Sistemas Informáticos. Univ. de Cádiz.
E.S. de Ingeniería de Cádiz. C/ Chile, s/n. 11003 Cádiz. España.
{juanjose.dominguez, antonia.estero, immaculada.medina}@uca.es
{manuel.palomo, francisco.palomo}@uca.es

Resumen Este trabajo discute la situación actual en la que se encuentra el desarrollo de los servicios Web (WS) en la comunidad del software libre. Por un lado, se analizan las tecnologías más importantes implicadas, algunas en fase de estudio y otras ya estandarizadas por organismos internacionales. Por otro, se ofrece una comparación entre distintas herramientas de desarrollo de WS, libres y privativas, atendiendo a los distintos niveles de abstracción existentes en estos servicios y a su adhesión a recomendaciones y estándares internacionales. Esto nos permite analizar qué papel juega hoy día el software libre en el sector de los WS y establecer conclusiones acerca de dónde se deberían centrar los esfuerzos de la comunidad de desarrollo de software libre en este campo. Finalmente, se realiza una propuesta realista para la comunidad acerca de qué debería hacerse en los próximos años para lograr colocar al software libre a la altura del software privativo en el desarrollo de WS.

1. Introducción

El World Wide Web Consortium (W3C) define un servicio Web (WS) como una aplicación software identificada por un URI cuyas interfaces se pueden definir, describir y descubrir mediante documentos XML. Los WS permiten la interoperación de sistemas distribuidos heterogéneos con independencia de las plataformas hardware y software empleadas [7].

Puede pensarse en ellos como en una arquitectura, conceptual y tecnológica, que hace posible que distintos servicios se describan, publiquen, descubran y utilicen a través de sistemas distribuidos, empleando la infraestructura proporcionada por Internet. Este enfoque está avalado por la aparición de un gran número de recomendaciones y estándares, que han sido impulsados por los grandes actores empresariales del sector de las TIC, en los que XML forma la espina dorsal de estas tecnologías.¹

¹ No trataremos en este trabajo las tecnologías basadas en REST ni en ebXML, que representan enfoques más específicos y no necesariamente ligados al desarrollo de WS en el sentido estricto de la definición que proporciona el W3C.

En realidad, los WS constituyen un caso particular de arquitectura orientada a servicios (SOA). A pesar de todo lo que se puede leer hoy día sobre la revolución de las SOA, en el fondo no son más que arquitecturas distribuidas —al igual que las que pueden crearse empleando CORBA, DCOM, EJB y otras tecnologías— que tienen una serie de características que las hacen de especial interés:

1. Diversifican las oportunidades de negocio al facilitar que aparezcan escenarios de libre intercambio de servicios normalizados.
2. Son arquitecturas acopladas de manera muy débil, lo que facilita la operación entre plataformas con distinto hardware, sistema operativo o que empleen distintos lenguajes de programación en sus aplicaciones.
3. Los servicios son fáciles de reutilizar y reemplazar, lo que produce una reducción de costes, especialmente del de mantenimiento.
4. Es posible lograr un mayor control sobre el funcionamiento de los procesos remotos cuando se emplean las tecnologías adecuadas.
5. Pueden funcionar síncrona o asíncronamente, según las necesidades.

No obstante, también presentan algunas desventajas como que su rendimiento es, por lo general, bastante inferior al de otras arquitecturas distribuidas clásicas, y que necesitan una serie de tecnologías y especificaciones² adicionales para poder operar de manera adecuada (con seguridad, fiabilidad, etc.).

Aunque el hecho de que el desarrollo de estas tecnologías vaya ligado a procesos de estandarización es importante, no basta sólo con disponer de estándares, es necesario disponer de herramientas que los implementen. En este sentido, creemos que la comunidad del software libre no está a la cabeza en la carrera tecnológica de los WS, liderada actualmente por herramientas privativas de grandes empresas del sector de las TIC.

Para analizar la situación actual ofrecemos una comparación entre diversas herramientas de software disponibles, tanto libres como privativas, organizada en torno a los distintos niveles o capas de abstracción existentes en el desarrollo de estos servicios, estudiando en qué medida respetan las recomendaciones y estándares internacionales. También se estudian las posibilidades que presentan, dónde se deberían centrar los esfuerzos de desarrollo de software libre en este campo y, en definitiva, qué papel juega actualmente el software libre en el campo de los WS y qué papel puede jugar en el futuro si se toman las medidas adecuadas.

El objetivo final es realizar una propuesta realista para la comunidad del software libre acerca de qué podría hacerse en los próximos años para lograr colocar al software libre a la cabeza en el desarrollo de WS.

La estructura del trabajo es la siguiente. En primer lugar, la sección 2 presenta alguna de las tecnologías más importantes relacionadas con los WS. A continuación, las secciones 3 y 4 describen ejemplos representativos de herramientas disponibles como software privativo y software libre, respectivamente. Seguidamente, en la sección 5, se comparan y se realiza una propuesta acerca de qué podría hacer la comunidad del software libre para mejorar su situación en este campo. Finalmente, en la sección 6 se ofrecen algunas conclusiones.

² Comúnmente, a estas especificaciones se las denomina colectivamente WS-* y forman lo que se ha dado en llamar «la pila de los WS» (*the WS stack*).

2. Principales tecnologías involucradas en los WS

Los WS engloban una serie de tecnologías XML que se encargan de solucionar problemas concretos de interoperación.³ La mayor parte de ellas nacieron de empresas privadas (normalmente un consorcio de varias empresas a las que se iban uniendo otras a medida que la necesidad de la tecnología era más patente).

Muchas de estas tecnologías han sido remitidas a organismos de estandarización, principalmente W3C y OASIS, que o bien las han rechazado, o bien han creado un grupo de trabajo para convertirlas en recomendaciones o estándares.

2.1. Procesos de estandarización en W3C y OASIS

Para que una propuesta se convierta en una recomendación W3C se parte de un informe técnico que se asigna a un grupo de trabajo. Estos informes técnicos siguen el proceso técnico [32] que se resume a continuación.

Borrador de trabajo. Un informe técnico se convierte en un borrador de trabajo (*working draft*) que se publica para su revisión por parte de la comunidad.

Borrador de trabajo en última convocatoria. Cuando un borrador cumple ciertos requisitos, queda a la espera (*working draft in last call*) de que los grupos que trabajan en materias afines confirmen que no presenta problemas.

Recomendación candidata. Cuando un borrador de trabajo supera su última convocatoria puede convertirse, si bien este paso no es imprescindible, en recomendación candidata (*candidate recommendation*) con el objetivo de recopilar información sobre su posible implementación.

Propuesta de recomendación. Una vez que un documento ha sido enviado al W3C Advisory Committee para que lo eleve a recomendación, se convierte en una propuesta de recomendación (*proposed recommendation*).

Recomendación. Una recomendación (*recommendation*) equivale en el ámbito del W3C a un estándar de otros organismos.

En OASIS, las propuestas se asignan a un comité y siguen un proceso técnico [14] distinto al anterior. Comentamos brevemente las etapas que atraviesa una propuesta hasta convertirse en estándar.

Borrador de comité. Una vez se considera que una propuesta es de interés público se publica como borrador de comité (*committee draft*). Estos borradores pueden ser revisados y publicados tantas veces como se considere necesario.

Borrador para revisión pública. Cuando se considera que un borrador de comité ha alcanzado su madurez se publica como un borrador para revisión pública (*public review draft*) sobre el que cualquier interesado puede opinar.

Especificación de comité. Vistas las propuestas de revisión y realizadas las modificaciones pertinentes, un borrador para revisión pública se promociona a especificación de comité (*committee specification*).

Estándar OASIS. Cuando los miembros de OASIS aprueban una especificación de comité ésta pasa a convertirse en estándar OASIS (*OASIS standard*).

³ Supondremos al lector familiarizado con las tecnologías XML fundamentales como XML Schema, XSLT, etc. Tampoco definiremos conceptos como DTD o XSD.

2.2. Especificaciones WS-*

A continuación se describen las principales especificaciones relacionadas con los WS disponibles en la actualidad. W3C y OASIS mantienen una relación exhaustiva de informes técnicos, borradores, recomendaciones y estándares relacionados con estas tecnologías [31,15]. Puede encontrarse información general sobre estas especificaciones y sus aplicaciones en [6,13].

SOAP Recomendación W3C [36]. Especifica la estructura de los mensajes que los WS intercambian. Es independiente de la plataforma, flexible y fácilmente extensible.

WSDL Recomendación candidata del W3C [34]. Permite la descripción de WS: estructura de los mensajes SOAP que intercambiará con otros WS, servicios que ofrece, negociación de los parámetros de seguridad en las comunicaciones entre WS, etc.

UDDI Estándar OASIS [16]. Permite mantener repositorios de especificaciones WSDL simplificando el descubrimiento de WS y el acceso a sus especificaciones. Hace posible que una aplicación busque dinámicamente servicios que ofrezcan una serie de características, seleccione el más adecuado (por coste, calidad, etc.) e incluso localice servicios alternativos si uno falla.

WS-Addressing Recomendación W3C [33]. Permite incluir en un mensaje SOAP información sobre el emisor, los destinatarios, a quién se debe responder, a quién se debe informar en caso de error, etc. Con estos elementos se consigue un direccionamiento independiente de capas de transporte inferiores como HTTP.

WS-AtomicTransaction Borrador para revisión pública de OASIS [25]. Suele emplearse en la coordinación de WS que realicen actividades de corta duración en entornos de confianza. Las acciones a realizar por cada servicio implicado se agrupan en una transacción atómica. El coordinador decide cuándo realizarla y puede abortar una transacción en curso devolviendo a los WS implicados a su estado previo.

WS-BPEL Especificación de comité OASIS [17]. Define un lenguaje que facilita la composición de WS. Permite especificar la lógica de la composición de los servicios (envío de mensajes, sincronización, iteración, tratamiento de transacciones erróneas, etc.) independientemente de su implementación.

WS-BusinessActivity Borrador para revisión pública de OASIS [25]. Permite la coordinación de WS compuestos (normalmente con WS-BPEL) a partir de actividades independientes que no se pueden modelar como transacciones atómicas por su duración, por requerir intervención humana o por ser incapaces de bloquear recursos.

WS-Coordination Borrador para revisión pública de OASIS [25]. Permite crear contextos de coordinación para la sincronización de WS. Requiere protocolos complementarios como WS-AtomicTransaction o WS-BusinessActivity.

WS-DistributedManagement Estándar OASIS [18]. Permite gestionar recursos distribuidos de todo tipo (PDA, televisores, dispositivos de conexión de redes, etc.) mediante WS.

WS-Notification Estándar OASIS [19]. Permite que un WS reciba información puntual sobre determinados acontecimientos y está formado por tres especificaciones: WS-BaseNotification, WS-BrokeredNotification y WS-Topics.

WS-Policy Borrador de trabajo del W3C [35]. Proporciona un medio de especificar las características que presentan y exigen los WS durante su operación. Por ejemplo, un determinado servicio puede exigir para operar que los mensajes se firmen o cifren con determinados algoritmos o que la coordinación se realice mediante un protocolo dado. Con esto se dota a los WS de la capacidad de negociar entre ellos las condiciones de interacción.

WS-Reliability Estándar OASIS [21]. Es un protocolo que permite numerar los mensajes SOAP y obtener confirmación de su recepción en destino. Con esto se puede garantizar el orden de recepción de los mensajes, evitar duplicados, comprobar la entrega, etc.

WS-ReliableMessaging Borrador para revisión pública de OASIS [20]. Esta especificación define un protocolo que permite el intercambio de mensajes de manera fiable en presencia de fallos en el software, la red, etc.

WS-ReliableMessagingPolicyAssertion Borrador para revisión pública de OASIS [20]. Con este protocolo se puede manejar políticas WS-Policy que expresen los requisitos de los emisores y receptores de mensajes que usen WS-ReliableMessaging.

WS-ResourceFramework Estándar OASIS [22]. Mediante este protocolo se pueden modelar y utilizar servicios con estado interno. Permite superar las limitaciones que poseen las tecnologías WS-* respecto a REST, que sí permite contemplar servicios con estado interno. Para ello se definen varias especificaciones: WS-Resource, WS-ResourceProperties, WS-ResourceLifetime, WS-ServiceGroup y WS-BaseFaults.

WS-SecureConversation Borrador para revisión pública de OASIS [23]. Define extensiones de WS-Security que proporcionan un marco de trabajo en el que se pueden solicitar y emitir *tokens* de seguridad, de manera que los agentes implicados puedan establecer relaciones de confianza mutua.

WS-Security Estándar OASIS [24]. Proporciona integridad, confidencialidad y autenticación en las comunicaciones entre WS. Incluye varios protocolos de seguridad, como X.509 y Kerberos, de manera que los WS puedan utilizar distintas políticas de seguridad.

WS-Trust Borrador para revisión pública de OASIS [23]. Su objetivo es facilitar el intercambio de series de mensajes seguros mediante la emisión, renovación y validación de *tokens* de diversos protocolos.

3. Software privativo en WS

Actualmente existe una amplia gama de productos privativos relacionados con los WS, por lo que no ha sido fácil llevar a cabo una selección de herramientas, de entre las múltiples opciones disponibles en el mercado, que resultara suficientemente representativa para la realización de un estudio comparativo. Los criterios seguidos para esta selección son los siguientes:

- En primer lugar, se han seleccionado herramientas pertenecientes a las principales compañías desarrolladoras de software. En concreto, se han elegido herramientas de las siguientes compañías: Microsoft, Sun Microsystems, Oracle e IBM, ya que se puede considerar que son los grandes actores en el mercado de los WS.
- Dado que todas estas compañías producen un vasto abanico de productos relacionados con los WS, se ha optado por escoger, para cada una de ellas, la herramienta integrada que proporcione mayores posibilidades a la hora de trabajar con WS.

A continuación se exponen las principales características de las herramientas de software privativo incluidas en el estudio.

3.1. Microsoft Windows Communication Foundation

Windows Communication Foundation (WCF) [27] es la plataforma de Microsoft para el desarrollo y ejecución de WS. WCF integra un conjunto de tecnologías proporcionadas por .NET Framework, tales como ASMX (denominada también ASP.NET Web Services), .NET Remoting, Enterprise Services y Web Services Enhancements, que implementan una serie de especificaciones WS-* y Microsoft Message Queuing.

WCF incluye un número significativo de tecnologías relacionadas con los WS entre las que destacan SOAP, WSDL, WS-Addressing, WS-AtomicTransaction, WS-Coordination, WS-Policy, WS-ReliableMessaging, WS-SecureConversation, WS-Security, WS-SecurityPolicy y WS-Trust.

3.2. Sun Microsystems Java 2 Enterprise Edition

La plataforma Java 2 Enterprise Edition 5 (J2EE 5) [28] de Sun Microsystems proporciona las herramientas necesarias para diseñar, desarrollar, probar y desplegar WS de forma rápida.

En relación a los WS, esta plataforma soporta SOAP 1.1/2.0, WSDL 1.1, UDDI 1.0, WS-Addressing y WS-Security.

3.3. Oracle JDeveloper

Oracle JDeveloper [26] es un entorno de desarrollo integrado que permite la construcción de aplicaciones orientadas a servicios. Esta herramienta permite desarrollar el ciclo de vida completo de este tipo de aplicaciones: modelado, codificación, depuración, prueba, perfilado, ajuste y despliegue.

Oracle JDeveloper permite desarrollar WS con SOAP, WSDL y UDDI. El soporte para UDDI incluye el despliegue de WS en repositorios, un navegador UDDI y la posibilidad de generar esqueletos de código para la activación de WS.

Este producto también integra otras tecnologías WS, como WS-Reliability y WS-Security. Por último, proporciona un entorno gráfico que permite diseñar procesos BPEL.

3.4. IBM Websphere Application Server

La compañía IBM dispone de un conjunto muy amplio de productos que constituyen la plataforma Websphere. Entre todos estos, se ha elegido IBM Websphere Application Server [8] por ser la base de la plataforma Websphere.

IBM Websphere Application Server 6.1 está disponible en múltiples configuraciones: Websphere Application Server Community Edition, Websphere Application Server Express, Websphere Application Server y Websphere Application Server Network Deployment. De todas ellas, destacamos Websphere Application Server Network Deployment, que es la más completa y a la que nos referiremos a continuación, y Websphere Application Server Community Edition, que se discute en el apartado 4, por tratarse de software libre.

La plataforma proporciona un entorno integrado de desarrollo y despliegue de WS. Soporta J2EE 5, así como las siguientes tecnologías relacionadas con los WS: SOAP, WSDL, UDDI 3.0, WS-Addressing, WS-BusinessActivity, WS-Notification y WS-Security.

4. Software libre en WS

Al igual que con las herramientas de software privativo, ha sido necesaria una selección de las herramientas de software libre a incluir en este estudio. Esta selección se ha basado en la madurez de las herramientas y en su representatividad dentro del mundo de los WS. Las licencias bajo las que se distribuyen estas herramientas son muy variadas e incluso hay casos en los que coexisten licencias libres con otras privativas, por lo que las mencionaremos explícitamente.

4.1. gSOAP Web Services Toolkit

El conjunto de herramientas gSOAP Web Services [30] permite desarrollar WS con C y C++. Se utiliza principalmente para desarrollar WS a partir de su descripción en WSDL. Entre las herramientas disponibles se incluye un generador de código fuente que realiza parte del trabajo de codificación e incorpora un analizador de WSDL y de esquemas XML capaz de asociar automáticamente los tipos que aparecen en los esquemas con tipos de datos de C y C++.

En la actualidad permite trabajar con SOAP 1.1/1.2, WSDL 1.1 y UDDI 2.0, así como con otras tecnologías como WS-Addressing y WS-Security. Entre los planes de desarrollo futuro está el incorporar nuevas especificaciones.

Las herramientas gSOAP se distribuyen con tres tipos de licencia: GNU GPL, código abierto público gSOAP y «comercial». Algunas partes no son libres, como el código fuente del analizador, que sólo se distribuye bajo la licencia comercial.

4.2. JBossWS

JBoss Application Server [10] es una plataforma J2EE certificada que permite desarrollar y desplegar aplicaciones Java empresariales, además de aplicaciones y portales Web. A partir de su versión 4.0.4 incorpora el submódulo JBossWS, que es el encargado de proporcionar el soporte para los WS.

JBossWS es capaz de trabajar con las siguientes tecnologías: SOAP 1.1/1.2, WSDL 1.1, UDDI, WS-Addressing, WS-BPEL, WS-Coordination, WS-Policy y WS-Security. JBoss Application Server y JBossWS se distribuyen con licencia GNU LGPL.

4.3. GlassFish

Este proyecto representa una alternativa libre al entorno J2EE, descrito en el apartado 3.2. Su desarrollo se debe a la liberación por parte de Sun Microsystems de Java System Application Server 9.0 PE. Esto permite a Sun desarrollar y distribuir sus tecnologías WS, mientras se garantiza a los desarrolladores del proyecto el acceso a las últimas versiones de las tecnologías XML y WS.

En este sentido, GlassFish, a partir de la liberación de la versión 2.0 del Java Web Services (Java WS) Developer Pack, soporta SOAP 1.1/2.0, WSDL 1.1, UDDI 1.0, WS-Addressing y WS-Security.

En cuanto a las licencias, la mayor parte del código del proyecto GlassFish está disponible bajo licencia CDDL. Sin embargo, ciertos componentes sólo están liberados en forma binaria bajo una licencia específica de Sun [29].

4.4. NetBeans IDE

NetBeans IDE [12] es un entorno de desarrollo integrado (IDE) de código abierto. Proporciona a los desarrolladores múltiples herramientas que permiten crear aplicaciones empresariales, tanto para la Web como para dispositivos portátiles. NetBeans IDE 5.0 ofrece soporte completo para J2EE 5.

Esta herramienta nos permite trabajar con SOAP 1.1/1.2 y WSDL 1.1/2.0. En su contra, podemos indicar que no se ha considerado UDDI entre los objetivos de desarrollo, aunque es posible su integración como elemento independiente.

El IDE se puede complementar con varios paquetes que le proporcionan funciones adicionales. Uno de estos paquetes es Enterprise Pack [11] que añade a NetBeans las herramientas necesarias para escribir, depurar y probar aplicaciones SOA utilizando XML, BPEL y Java WS. Este paquete instala herramientas gráficas para la creación de esquemas XML y el diseño de orquestaciones de WS con BPEL. NetBeans Enterprise Pack 5.5 es la última versión disponible hasta la fecha y permite trabajar con WS-BPEL 2.0, aunque no de forma completa.

Tanto NetBeans IDE como NetBeans Enterprise Pack se desarrollan y distribuyen bajo licencia CDDL.

4.5. Apache Axis2

Apache Axis2 [1] es una evolución de su predecesor, Apache Axis. Su diseño presenta una arquitectura modular que facilita la introducción de funciones adicionales y la integración de nuevas tecnologías de la pila WS. Esta plataforma integra SOAP 1.1/1.2, WSDL 1.1/2.0, WS-Addressing, WS-ReliableMessaging, WS-Security y WS-SecureConversation.

Además, como parte del proyecto, se han desarrollado varios módulos que permiten integrar otras tecnologías. Los principales son: Kandula [2], que proporciona WS-AtomicTransaction, WS-BusinessActivity y WS-Coordination, Sandesha [3], que implementa WS-ReliableMessaging, y Rampart [5], que incorpora WS-SecurityPolicy. Actualmente, se está desarrollando un nuevo módulo para soportar WS-Policy. Aunque la plataforma no permite trabajar directamente con UDDI, es posible integrar jUDDI [4] con Axis2.

En cuanto a la licencia, tanto Axis2 como los módulos descritos —Kandula, Sandesha y Rampart— se distribuyen bajo licencia Apache 2.0.

4.6. IBM Websphere Application Server Community Edition

IBM Websphere Application Server Community Edition [9] es un servidor de aplicaciones J2EE con tecnología Apache Geronimo que incorpora las últimas innovaciones en software libre para proporcionar una plataforma integrada y flexible con la que desarrollar y desplegar aplicaciones Java.

En relación a los WS, soporta las tecnologías SOAP, WSDL, UDDI, WS-BusinessActivity, WS-Notification y WS-Security. En comparación con otros productos de la familia no permite trabajar con UDDI 3.0.

Está construido sobre Apache Tomcat y otras aplicaciones de código abierto líderes como OpenEJB, Apache Axis e IBM Cloudscape, a su vez basada en Apache Cloudscape. Se distribuye bajo licencia Apache 2.0.

5. Comparación y propuesta de desarrollo

Después del estudio realizado en las secciones 3 y 4, presentamos en el cuadro 1 una comparación entre las distintas herramientas de acuerdo con las especificaciones con las que son capaces de trabajar.

Se observa que todas las herramientas permiten trabajar con SOAP y WSDL. La mayoría intentan incorporar UDDI y algún tipo de orquestación de servicios, ya sea mediante WS-Coordination o a través de WS-BPEL. La seguridad también aparece como un punto esencial en casi todas ellas.

Tras el estudio desarrollado en las secciones precedentes, estamos en disposición de realizar una propuesta realista para la comunidad del software libre acerca de qué debería hacerse en los próximos años para lograr colocar al software libre a la altura del software privativo en el desarrollo de WS. Dividiremos la propuesta a lo largo de varias áreas que afectan a esta tarea.

5.1. Bibliotecas fundamentales

Hasta ahora se ha conseguido un notable éxito en el desarrollo de pequeños componentes que, siendo importantes por sí mismos, son fundamentales para el desarrollo de los WS. Dado el carácter fragmentario de los equipos de desarrollo y de los distintos proyectos, un paradigma de desarrollo plausible consiste en crear primero bibliotecas completas para los principales lenguajes de programación, proporcionar aplicaciones de consola sobre estas bibliotecas y, por último, interfaces gráficas sobre dichas aplicaciones. Este enfoque no es nuevo y, de hecho, se está siguiendo con éxito en el desarrollo de aplicaciones multimedia libres.

Por lo general, un equipo se encarga de desarrollar una biblioteca y sus aplicaciones de consola, ya que, normalmente, estas aplicaciones son útiles para probar la biblioteca. Otros equipos, a través de distintos proyectos, pueden desarrollar distintas interfaces gráficas o incluso diferentes integraciones de la biblioteca o de sus aplicaciones de consola en entornos de desarrollo ya existentes.

Un reto es la integración de nuevos estándares en las herramientas disponibles con la mínima duplicidad de esfuerzos. Por ejemplo, es previsible que se desee disponer de bibliotecas de desarrollo para múltiples lenguajes. Esto suele dar lugar a una multiplicidad estéril de bibliotecas independientes con funciones solapadas. En vez de desarrollar una biblioteca para cada lenguaje, es posible reducir el esfuerzo de desarrollo si en su lugar se realizan, siempre que sea posible, envoltorios a bibliotecas desarrolladas en lenguajes de más bajo nivel. A este respecto, es importante que proyectos con intereses comunes se coordinen para evitar duplicar esfuerzos. Si la industria del software es capaz de hacerlo cuando se lo propone, no vemos por qué la comunidad del software libre no.

Un caso concreto lo representan las bibliotecas que se necesitan para implementar los protocolos de seguridad en los WS. Actualmente, XML Security Library representa un buen punto de partida, pero sólo está disponible en C. No sería difícil desarrollar una versión C++ como envoltorio de la biblioteca existente, ni una versión Java mediante JNI que podría integrarse en servidores de aplicaciones Java sin pérdida significativa de eficiencia.

	SOAP	WSDL	UDDI	WS-Addr.	WS-BPEL	WS-Bus.Act.	Ws-Coord.	WS-Not.	WS-Pol.	WS-Rel.	WS-Rel.Mess.	WS-Sec.
Microsoft WCF	x	x		x			x		x		x	x
Sun J2EE 5	x	x	x	x								x
Oracle JDeveloper	x	x	x		x					x		x
IBM Websphere AS ND	x	x	x	x		x		x				x
gSOAP WS Toolkit	x	x	x	x								x
JBoss WS	x	x	x	x	x		x		x			x
GlassFish	x	x	x	x								x
NetBeans IDE	x	x			x							x
Apache Axis2	x	x		x			x				x	x
IBM Websphere AS CE	x	x	x			x		x				x

Cuadro 1. Comparación entre software privativo y libre respecto al estado actual de las principales tecnologías WS.

5.2. Tecnologías XML

Como se puede comprobar, las tecnologías XML se encuentran en la base de gran parte de los elementos que constituyen la pila WS y son, por tanto, fundamentales para su desarrollo. Hoy día disponemos de numerosas bibliotecas libres para crear, analizar y manipular documentos XML en los lenguajes de programación más utilizados en la actualidad. También están disponibles multitud de editores, navegadores estructurales, herramientas de validación y transformación, etc. No obstante, existen algunos puntos débiles.

Creemos que es cuestión de tiempo que XML Schema se imponga a DTD como estándar para la descripción de documentos XML. Aparte del atractivo que presentan los esquemas como forma de describir XML en XML, disponen de ventajas técnicas evidentes. En este sentido, y puesto que también disponemos de herramientas libres de transformación de DTD a XSD, los esfuerzos de la comunidad deben concentrarse en proporcionar herramientas de edición y validación lo más completas posibles para XSD que a su vez sean fáciles de integrar en editores y entornos de desarrollo que manipulen documentos XML.

Debe prestarse especial atención a las cuestiones relativas a la internacionalización. De hecho, el formato nativo de codificación de textos para XML es UTF-8. Esto es fundamental para el desarrollo de WS, ya que por su naturaleza distribuida y de oferta global, muchos de estos servicios deben ser capaces de operar con clientes de distintas lenguas. Éste es un proceso complejo que implica desde el diseño de tipos de letra que cubran, al menos, el plano multilingüe básico (BMP) de Unicode, hasta el desarrollo de métodos de entrada para distintos lenguajes. Afortunadamente, la comunidad está realizando esfuerzos importantes en este sentido. Ya disponemos de consolas y de editores, como Emacs, que trabajan a un nivel aceptable con el BMP, así como de aplicaciones que gestionan diversos métodos de entrada como SCIM.

Por último, no son pocas las ocasiones en que la interacción del usuario final con un WS se realiza a través de un navegador Web. En el futuro, es previsible que estos WS produzcan documentos XML puros —en lugar de, por ejemplo, código XHTML— con la intención de que el navegador los valide y transforme para su presentación al usuario. Disponemos del excelente navegador Firefox, pero es importante que en un futuro sea capaz de validar documentos XML a partir de su XSD y que su motor de transformación XSLT permanezca actualizado a las últimas versiones del estándar.

5.3. Estándares

Estamos convencidos de que la comunidad del software libre debería concentrar sus esfuerzos en conseguir que sus instituciones y miembros más cualificados del sector de los WS se involucren en el proceso de creación y revisión de los estándares, así como en promover la adopción de éstos en sus productos.

Una peculiaridad del proceso de estandarización seguido por muchas de las tecnologías relacionadas con los WS es que está siendo impulsado de manera singular por un número reducido de grandes empresas. Esto es muy diferente

de lo que ocurre, por ejemplo, con los lenguajes de programación o con otras tecnologías Web donde históricamente los estándares han llegado cuando ha existido un apoyo amplio por parte de varias comunidades de programadores que han comenzado a divergir en aspectos importantes. Por un lado, esta situación con los WS supone una ventaja, ya que pueden hacerse las cosas bien «desde el principio», definiendo primero los estándares y creando luego las herramientas de manera que éstas cumplan los estándares. Por otro, existen algunos riesgos:

1. La presión de un grupo reducido de empresas no siempre de acuerdo en sus objetivos puede dar lugar a que en algunas áreas aparezca una multiplicidad de estándares solapados en sus funciones.
2. Los estándares van por delante de la tecnología o, al menos, de las herramientas disponibles. Algunas empresas partirán con ventaja en esta carrera al promover estándares que reflejen sus resultados más recientes en I+D.
3. La mayor rapidez en la creación de estándares, puede provocar la necesidad de introducir cambios significativos a través de sucesivas versiones, probablemente de manera precipitada. En el peor de los casos, algunos estándares podrían quedar obsoletos antes de que su adopción por parte de la comunidad internacional se consumara.

Un caso representativo de cómo la comunidad puede perder la carrera de los estándares en este sector es el proyecto DotGNU. El proyecto GNU está intentando convertirse en líder industrial y proveedor de software libre para los WS, principalmente como reacción a la política de Microsoft respecto a su plataforma .NET, a la que acusan de monopolista. DotGNU tiene entre sus objetivos desarrollar herramientas que permitan compilar y ejecutar aplicaciones .NET. En realidad, más que una implementación de .NET, DotGNU aspira ser una alternativa completa a .NET con un nivel razonable de compatibilidad.

El entorno de ejecución de DotGNU (DGEE) proporciona el componente fundamental que permite trabajar con WS. Sin embargo, únicamente es capaz de hacerlo a través de XML-RPC, un estándar que ya hace tiempo que ha sido superado por SOAP. Sin la incorporación de, al menos, SOAP y WSDL es difícil que DotGNU se convierta en una verdadera alternativa a .NET en cuanto a WS se refiere. SOAP y WSDL están lo suficientemente aceptados y extendidos como para que el proyecto DotGNU los incorpore sin temor a caer en un trampa.

Lo mismo podría decirse de WS-Security: sería difícil aceptar que DotGNU desarrollara su propia solución al problema de la seguridad de espaldas a otras ya existentes que, en nuestra opinión, son compatibles con la filosofía del proyecto.

5.4. Documentación

Otro aspecto importante es la documentación. Actualmente se pone más énfasis en el desarrollo que en la documentación. Muchos proyectos no están documentados en absoluto y en otros la documentación se reduce a la que se puede extraer automáticamente de los, a veces escasos, comentarios del código. Un paseo por SourceForge.net nos permite comprobar como, en algunos casos,

los desarrolladores ni siquiera se han molestado en rellenar el apartado de descripción del proyecto. Esta tendencia debe cambiar. Entender la documentación como una pérdida de tiempo o como una actividad *a posteriori* es un error.

En un contexto en el que los recursos humanos son siempre escasos, los proyectos deben primar la documentación frente a la traducción. Para bien o para mal, el inglés se ha convertido en una *lingua franca* en el desarrollo de software. Es preferible una buena documentación en inglés que cuatro malas traducciones.

6. Conclusiones

Hemos presentado una comparación entre diversos sistemas de software, tanto libres como privativos, disponibles actualmente para trabajar con WS. Esta comparación atiende a las distintas capas de abstracción existentes en los WS y al cumplimiento por parte del software de las recomendaciones y estándares internacionales más importantes en este sector.

También se han analizado qué capacidades tienen unos y otros, qué falta en los sistemas de software libre que tengan los sistemas privativos, dónde se deberían centrar los esfuerzos de desarrollo de software libre en este campo y, en definitiva, qué papel juega el software libre en el entorno de los WS.

Aunque la situación de partida no es mala, el mundo de los WS es muy cambiante y, si bien existen proyectos libres que se encuentran avanzando en la dirección correcta, el ritmo al que aparecen propuestas, borradores de trabajo, recomendaciones y estándares hace necesario un especial cuidado a la hora de decidir qué tecnologías implementar.

Se ha realizado una propuesta realista para la comunidad del software libre que contiene acciones concretas que deberían llevarse a cabo para colocar al software libre a la cabeza en el desarrollo de WS. Una conclusión final importante es que seleccionar los estándares adecuados e incorporarlos a las herramientas disponibles con la mínima duplicación de esfuerzos posible es quizás el reto más difícil al que se enfrenta la comunidad. Para lograr esto creemos que no sólo es necesaria una mayor coordinación entre proyectos distintos, sino que se debe trabajar en promover una mayor implicación de la comunidad del software libre en los procesos de estandarización.

Referencias

1. Apache Software Foundation. *Apache Axis2/Java*. 2007.
<http://ws.apache.org/axis2>
2. Apache Software Foundation. *Apache Kandula*. 2007.
<http://ws.apache.org/kandula>
3. Apache Software Foundation. *Apache Sandesha*. 2007.
<http://ws.apache.org/sandesha>
4. Apache Software Foundation. *jUDDI*. 2007.
<http://ws.apache.org/juddi>
5. Apache Software Foundation. *Securing SOAP messages with Rampart*. 2007.
http://ws.apache.org/axis2/modules/rampart/1_1/security%-module.html

6. Erl, T. *Service-Oriented Architecture Concepts, Technology, and Design*. Prentice Hall. 2005. ISBN 978-0131858589.
7. Fensel, D. y Bussler, C. *The Web service modeling framework WSMF*. En *Electronic Commerce Research and Applications*, páginas 113–137. 2002.
8. IBM. *Websphere application server*. 2007.
<http://www-306.ibm.com/software/webservers/appserv/was>
9. IBM. *Websphere application server community edition*. 2007.
<http://www-306.ibm.com/software/webservers/appserv/comm%unity>
10. JBoss. *Application server*. 2007.
<http://labs.jboss.com/portal/jbossws>
11. NetBeans. *NetBeans enterprise pack*. 2007.
<http://www.netbeans.org/products/enterprise>
12. NetBeans. *NetBeans IDE*. 2007.
<http://www.netbeans.org/products/ide>
13. Newcomer, E. *Understanding Web Services. XML, WSDL, SOAP, and UDDI*. Addison-Wesley. 2002. ISBN 978-0201750812.
14. Organization for the Advancement of Structured Information Standards. *Governing the OASIS Technical Committee process*. 2007.
<http://www.oasis-open.org/committees/process.pdf>
15. Organization for the Advancement of Structured Information Standards. *Standards and other approved work*. 2007.
<http://www.oasis-open.org/specs>
16. Organization for the Advancement of Structured Information Standards. *Universal description, discovery, and interoperability specification technical committee*. 2007.
http://www.oasis-open.org/committees/tc_home.php?wg_abb%rev=uddi-spec
17. Organization for the Advancement of Structured Information Standards. *Web services business process execution language technical committee*. 2007.
http://www.oasis-open.org/committees/tc_home.php?wg_abb%rev=wsbpel
18. Organization for the Advancement of Structured Information Standards. *Web services distributed management technical committee*. 2007.
http://www.oasis-open.org/committees/tc_home.php?wg_abb%rev=wsdm
19. Organization for the Advancement of Structured Information Standards. *Web services notification technical committee*. 2007.
http://www.oasis-open.org/committees/tc_home.php?wg_abb%rev=wsn
20. Organization for the Advancement of Structured Information Standards. *Web services reliable exchange technical committee*. 2007.
http://www.oasis-open.org/committees/tc_home.php?wg_abb%rev=ws-rx
21. Organization for the Advancement of Structured Information Standards. *Web services reliable messaging technical committee*. 2007.
http://www.oasis-open.org/committees/tc_home.php?wg_abb%rev=wsrc
22. Organization for the Advancement of Structured Information Standards. *Web services resource framework technical committee*. 2007.
http://www.oasis-open.org/committees/tc_home.php?wg_abb%rev=wsrf
23. Organization for the Advancement of Structured Information Standards. *Web services secure exchange technical committee*. 2007.
http://www.oasis-open.org/committees/tc_home.php?wg_abb%rev=ws-sx
24. Organization for the Advancement of Structured Information Standards. *Web services security technical committee*. 2007.
http://www.oasis-open.org/committees/tc_home.php?wg_abb%rev=wss

25. Organization for the Advancement of Structured Information Standards. *Web services transaction technical committee*. 2007.
http://www.oasis-open.org/committees/tc_home.php?wg_abb%rev=ws-tx
26. Shmeltzer, S. *Oracle JDeveloper overview*. 2007.
<http://www.oracle.com/technology/products/jdev>
27. Shodjai, P. *Web services and the Microsoft platform*. 2006.
<http://msdn2.microsoft.com/en-us/webservices>
28. Sun Microsystems, Inc. *Java platform enterprise edition 5 specification*. 2004.
<http://jcp.org/aboutJava/communityprocess/final/jsr244>
29. Sun Microsystems, Inc. *Licencia de GlassFish*. 2007.
<http://wiki.java.net/bin/view/Projects/GlassFishCodeDep%endencies>
30. van Engelen, R. y Gallivan, K. *The gSOAP toolkit for web services and peer-to-peer computing networks*. En *2nd IEEE International Symposium on Cluster Computing and the Grid (CCGrid 2002)*, páginas 128–135. 2002.
31. World Wide Web Consortium. *Technical reports and publications*. 2007.
<http://www.w3.org/TR>
32. World Wide Web Consortium. *W3C process document*. 2007.
<http://www.w3.org/Consortium/Process>
33. World Wide Web Consortium. *Web services addressing working group*. 2007.
<http://www.w3.org/2002/ws/addr>
34. World Wide Web Consortium. *Web services description working group*. 2007.
<http://www.w3.org/2002/ws/desc>
35. World Wide Web Consortium. *Web services policy working group*. 2007.
<http://www.w3.org/2002/ws/policy>
36. World Wide Web Consortium. *XML protocol working group*. 2007.
<http://www.w3.org/2000/xp/Group>

An Open Source Framework Oriented to Modular Web-Solution Development based in Software Product Lines

Ildefonso Montero Pérez

Dpto. de Lenguajes y Sistemas Informáticos, Universidad de Sevilla
Av. Reina Mercedes s/n 41012 Seville (Spain)
monteroperez@us.es

Resumen In this paper we present the GPL license open source project *Portalframework*, oriented to design and to implement a framework that would be able to give us an agil development enviroment of web-solutions based on software product lines and software factory approaches. Current software applications are day by day more complex. A software factory approach is devoted to overcome complexity providing all the techniques and tools needed for enabling the mass production of software in a limited application domain. The main goal of this project is to reach a notable increment of productivity by means of identification and parametrization of common web modules in a standard web application or web portal, and the automatization of the implementation process of this products, oriented to mass production with a factor of variability.

1. Introduction

Software Product lines, SPL [2], give us a way to obtain a software product family based on reutilization and extension of a common product obtained by means of a previous domain engineering step, and a set of products with a determined variability factor that are the results of the needings to satisfy for each one in application engineering step. A Modular Web-Solution consists of a software product oriented to the web that give us a specific functionality, are called components or modules in frameworks of content management oriented to the web, Web Content Management Systems: CMS, or oriented to develop web applications based on implementations of patterns like Model-View-Controller, MVC [25]. *Portalframework* presents an approach of framework based on SPL theory that cover the life-cycle development of modular web-solutions. This implies that it provides an integrated development enviroment based on a MVC pattern implementation taken as input the information of the desired product represented in XML metadata format as a feature model, being the output the software product desired able to be deployed in a application web server.

2. Related work and motivation

There exist several approaches to develop web-solutions by means of an implementation of MVC pattern as Struts [17] or Spring [18], but there not exists tools support to the software product lines theories applied to this scope or context. We are going to introduce FAMA [5] [4] as a SPL tool that is able to give us an automated analysis of features models integrating some of the most commonly used logic representations and solvers. The reason of quote this software tool is because it can be integrated with *Portalframework* in an Eclipse [20] environment to have a better productivity and make it easier to use. Thus, the main motivation of this work is two fold: (i) there not exist a tool support to develop a software product line of modular web-solutions to the best of our knowledge, (ii) we can integrate this product with others to obtain a well-formed integrated development environment based on SPL and (iii) we define this product as an open source project with the objective to the community could participate in it and making it grow faster.

3. *Portalframework* architecture

Portalframework development has done in the following technological environment where we only used open source tools: Eclipse 3.0 with PHPEclipse-Plugin [21] and XAMPP [22] , Apache Server with MySQL database, PHP and phpm-admin. Core and modules of *Portalframework* has been written in PHP5 [23]. Now, we are going to present the architecture of this tool.

The first step is the elicitation user requirements. This requirements give us all the user needings that the final product has to satisfy [7] [8] [9]. The variability factor of a requirement is too high and is a very important concept in a SPL because determines the different models of the same product family. The proposed workflow lies in obtaining the requirements and translating it to a features model tree. A feature model [10] [11] [12] [13] [14] [15] [16] [3] is a hierarchical model that can be used to represent system goals/features. A feature is anything users or client programs might want to control about a concept. Thus, during feature modeling, we document not only functional features but also implementation features, various optimizations, alternative implementation techniques, and so on. In Figure 1 we present the feature model of the case study used in this paper, a collaborative web application environment for OSLUCA [24], acronym of Oficina de Software Libre de la Universidad de Cádiz, Free Software Office of University of Cádiz, members.

We can see three parental relationship between each feature that can be:

Mandatory: If a child feature node is defined mandatory, it must be included in every father feature products. For example, every collaborative environment has to give us a *User Management*. This is represented in Figure 1 feature model as a mandatory relationship between OSLUCA and *User Management* node.

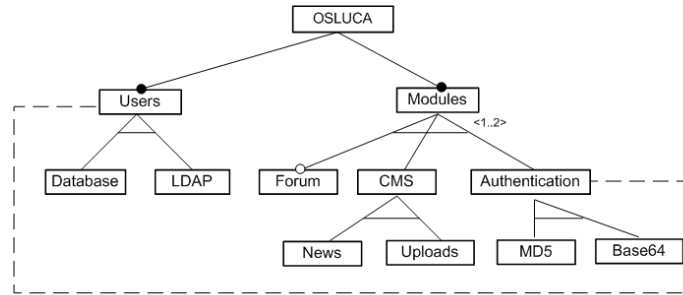


Figura 1. OSLUCA feature model

Optional: If a child feature node is defined optional, it could be included in every father feature products. For example, we can define as optional *Forum* node in *Modules* feature.

Alternative: If the relationship between a set of children nodes and their father is defined alternative, only one of the children features could be included in every father feature products. For example, every *Authentication* module can give us or *Base64* or *MD5* authentication method.

These models can be generated by means of self-automatic tools that process the requirements in a metadata format, expressed for example in Alan Cockburn [6] [8] [7] [9] templates translated to XML notation. There exists software tools like FAMA that can provide us a environment to validate the feature model created by different policies like constraint satisfaction problems, CSP, SAT o JBD. In Figure 1 example we identify basical features of a web collaboration environment. We are going to develop a factory to mass production of authentication modules and user management module oriented to find users in the database of the environment. This two products will be a web-solution product that can be inserted in a web application or webpage to be deployed in a application server.

The structure of the web-solution, as we can see in Figure 2, will be the following:

init.php: writable script that the *framework* will use to build and deploy the web-solution into a web server.

Packages: set of interfaces and classes needed to the implementation of the functionalities that the user desire with the alternative of redefine it.

Data: set of data needed to store that the framework will generate automatically.

4. Features of our approach

The two mainly features that makes *Portalframework* an attractive product are the following:

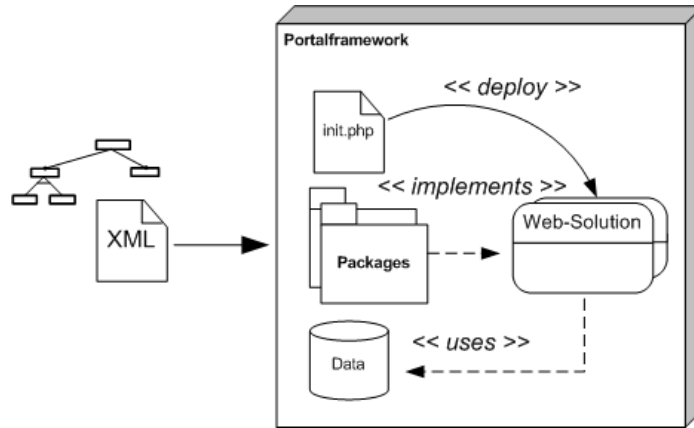


Figura 2. Portalframework architecture

Architecture and design based in MVC pattern, as other products like Struts or Spring with the difference that actions methods are now implemented in the framework and can be used and redefined by the user. It means that *Portalframework* can be considered as a module container that give us a complete solution to a user needing, we have only to call the required module in the init script to be deployed in the user web application.

Each module can be extended as free as we want with zero cost and without impact. We are looking for an optimal **maximum cohesion and minimal coupling** as possible because as we quoted before there exists a variability factor that can crash our web-solutions.

5. Case Study: OSLUCA web collaboration enviroment

We present the case study development process. First we have to create a new project into Portalframework related with the desired web-solution called *osluca*. It will be deployed in Eclipse, but it is not a necessary condition, but justified if we use FAMA plugin to develop the initial feature model.

The project package and folder structure is the following:

- /class: it store all the classes needed to establish the «implements» relationship between framework packages and the web-solution. It contains:
- /class/modules: it store the set of classes that define the modules needed to aim the functionalities to satisfy by the web-solution represented in the feature model.
- /class/request: it store the set of classes that implements the request actions that provides the web-client side to the web-server side.
- /class/src: it store the set of clases that implements the business model. These classes are auto-generated by the *datagen* tool utility provided in *Portal-*

framework that establishes a mapping between stored data and instanciable bean entities.

`/class/views`: it store the set of classes that represents the views to apply in the web-solution.

`/class/oslucaportal.class.php`: it contains the definition of the class that represents the container portal of the modules that we need to satisfy the user needings.

`/css`: it store the set of cascade stylesheets files to apply to web-solutions.

`/data`: it store the classes to access to a database:

`/data/core`: it store the classes to represent a single point to access to the project database designed by a Facade GoF pattern.

`/data/dao`: it store the classes to provide adapters to Oracle and MySQL architectures using the Access Data Object pattern.

`/http`: it store the HTML files that provides the deployed web-solutions.

`/src`: it store the init script: `init.php`.

This structure must be generated automatically and can be editable always that we require it. Now we present the deploy file `init.php`, where we can see the following areas:

Definition files: This area contains the set files calls needed to the valid deploy of our solutions. In this area we can see the set of files included in the project with their dependencies. There are the following blocks areas:

- Framework Configuration files
- Web-Portal-solution definition files
- Web-Modules-solution definition files with their dependencies
- Views definition files
- Entities/Beans definition files

Portal Forms: This area contains the block to load the forms into the portal. Each form that be processed in web-client side and contains the implementation of one functionality needed will be stored as a session variable into the web-browser.

Portal and Modules Creation: This area contains all the commands needed to introduce a module into the portal.

To develop our example first we create an object that represents the portal in init deploy file in portal and modules creation area, the source code of this has the following aspect:

```
// Creating a new portal ...
$portal = OSLUCAPortal::getPortal();
$portal->setAuthor("Ildefonso Montero Pérez");
$portal->setTitle("OSLUCA portal");
$portal->setVersion("0.1");
$portal->setDateCreation(date("Ymd"));
```

Now we create the modules that can give us an implementation of the features needed by the user, in this example we need an authentication module and a list module:

```
// Creating an authentication module ...
$auth = new AuthOSLUCAModule(new User(),
new UserDAO(),
new OSLUCADataAccess());
```

This example contains a creation of a new module using two auto-generated modules by `datagen` called `User` and `UserDAO`, that contains a representation of a collaboration environment user and the datapoint of the table in the database that store his data. To access to this database we use an instance of `OSLUCADataAccess` facade, an extension of generic data access object provided by the framework.

Now we can define the authentication type, it is a variation point into the variability factor of the software product line. It means that our factory can develop web-solutions to authenticate by different policies, in that way for each policy we could have one product to mass production in the factory. In this example we define Base64 authentication method as variation point:

```
$auth->getModule()->setType(BASE64AUTH);
```

Now, we create the List Module as we have created Authentication module:

```
// Creating a list module ...
$list = new ListOSLUCAModule(new User(),
new UserDAO(),
new OSLUCADataAccess());
```

The meaning of the three parameters of the creation of a module is the following: the first parameter is the object to impact, if we are talking about authentication the functionality has to work with the entity `User` to validate it. The second parameter is the ADO object to access to data information of the entity quoted, always with the rule *Entity* and *EntityDAO*, and the last one is the facade to the database architecture.

To finish we have to add this modules into the portal:

```
// Adding modules into the portal ...
$portal->addModule($auth);
$portal->addModule($list);
```

Figure 4 presents the final result when we deploy this script into the web application server. It shows four modules, two are the quoted in this paper and the other two are a `RichTextModule` and a `TextModule` to write the messages.

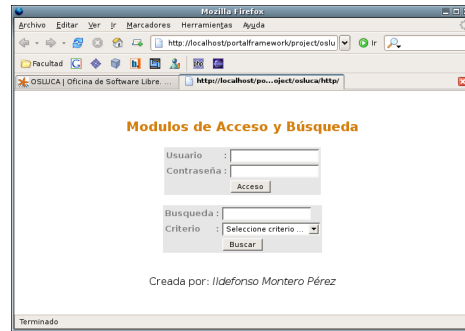


Figura 3. Web-Solution auto-generated by *Portalframework*

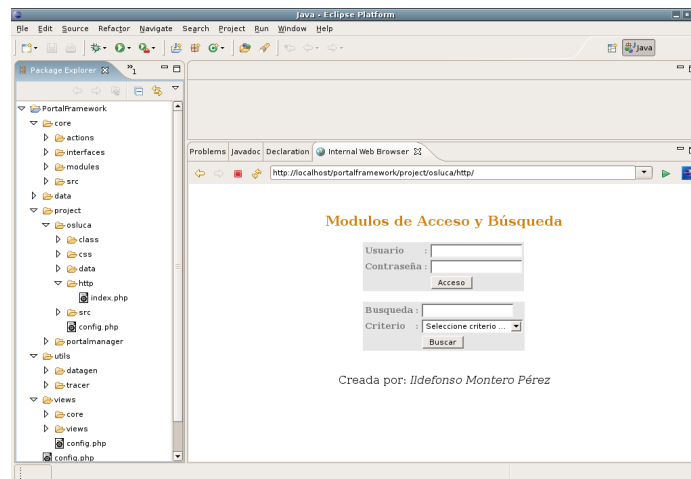


Figura 4. *Portalframework* integrated into Eclipse

6. Conclusions and Future Work

As a result of the framework presented, we draw two main conclusions:

The need of open source tools focused to develop web-solutions based in SPL: in that way we develop *Portalframework* as a first approach for this need.

Research areas like:

- Reutilization pattern identification and analysis in web development.
- Automatization in life cycle development of software products based in features model, reasoners, ontologies, etc. as for example Apache Maven [19].
- Software factories and Business Driven Development [1]

Referencias

1. Jack Greenfield, Keith Short, Steve Cook, Stuart Kent, John Crupi, “Software Factories: Assembling Applications with Patterns, Models, Frameworks, and Tools”
2. Klaus Pohl, Günter Böckle and Frank J. van der Linden, “Software Product Line Engineering: Foundations, Principles and Techniques”
3. Ildefonso Montero, Joaquin Peña, Antonio Ruiz-Cortés, Amador Duran, “Sended to the 10th Workshop Engineering Requirements (WER 2007) Survey on Approaches and Tools for Documenting Variability in Requirements of Software Factories” 2007
4. D. Benavides, S. Segura, P. Trinidad and A. Ruiz-Cortés, “Managing Variability for Software Product Lines: Working With Variability Mechanisms. Managing Variability for Software Product Lines: Working With Variability Mechanisms” (2006)
5. D. Benavides, S. Segura, P. Trinidad and A. Ruiz-Cortés, “Proceeding of the First International Workshop on Variability Modelling of Software-intensive Systems (VAMOS). FAMA: Tooling a Framework for the Automated Analysis of Feature Models” (2007)
6. A. Cockburn, “Structuring Use Cases with Goals. Journal of Object-Oriented Programming” (1997)
7. Amador Durán Toro, B. Bernárdez Jiménez, Antonio Ruiz Cortés and M. Toro Bonilla, “In proceedings of the Workshop Engineering Requirements (WER 1999). A Requirements Elicitation Approach Based in Templates and Patterns.” (1999)
8. Amador Durán Toro, “PHD Thesis: A Methodological Framework for Requirements Engineering of Information Systems.” (2000)
9. A. Durán y B. Bernárdez, “Metodología para la Elicitación de Requisitos de Sistemas Software (versión 2.3). Informe Técnico LSI-2000-10” (2000)
10. K. Kang, S. Cohen and S. Peterson, “Technical report Carnegie Mellon University, Feature-oriented domain analysis (FODA)” (1990)
11. K. Kang and P. Donohoe, “Technical report IEEE Software, Feature-oriented product line engineering” (2002)
12. M. Griss and M. d’Alessandro, “In proceedings of the Fifth International Workshop on Product Family. Integrating feature modelling with the RSEB” (1998)
13. D.S. Mathias Riebish, Kai Böllert and I. Philippow, “In proceedings of the Sixth Conference on Integrate Desing and Process Technology. Extending feature diagrams with UML multiplicities” (2002)
14. T. Pierre-Yves Schobbens, Patrick Heymans and Y. Bontemps, “Generic semantics of Feature Diagrams” (2006)
15. T. Pierre-Yves Schobbens, Patrick Heymans and J-C. Trigaux, “Feature diagrams: A survey and a formal semantics.” (2006)
16. Jean-Christophe Trigaux, Patrick Heymans, “Modelling variability requirements in Software Product Lines: a comparative survey.” (2003)
17. The Struts framework website. <http://struts.apache.org>
18. The Spring framework website. <http://www.springframework.org>
19. The Apache Maven website <http://maven.apache.org>
20. The Eclipse website <http://www.eclipse.org>
21. The PHPEclipse plugin website <http://sourceforge.net/projects/phpeclipse>
22. The XAMPP website <http://www.apachefriends.org/en/xampp>
23. The PHP website <http://www.php.net>
24. The OSLUCA website <http://osl.uca.es>
25. Wikipedia entry for MVC pattern <http://en.wikipedia.org/wiki/Model-view-controller>

SIGA

Sistema Integrado de Gestión de Aulas

Francisco Almeida Rodríguez
Alberto Morales Díaz
Jonás Regueira Rodríguez

Secretariado del Software Libre, Universidad de La Laguna
Vicerrectorado de Tecnologías de la Información
y las Comunicaciones
info@ssl.ull.es
<http://ssl.ull.es>

Resumen Los alumnos de la Universidad de la Laguna (ULL) disponen de un conjunto de aulas de informática de uso genérico, y laboratorios de sus correspondientes centros en los que pueden trabajar y realizar sus actividades. Estas aulas, aparte de estar físicamente dispersas en los campus, están gestionadas de acuerdo a criterios fijados por cada facultad o escuela y lo más habitual es encontrar entornos operativos heterogeneos y basados en tecnologías propietarias.

Presentamos en este trabajo el estado actual del proyecto SIGA (Sistema Integrado de Gestión de Aulas) con el que pretendemos centralizar, unificar y migrar las aulas de informática de la ULL a Software Libre. En este artículo se describirán los problemas encontrados en esta migración, la justificación y descripción de las soluciones adoptadas.

Palabras clave: administración aulas implantación bardinix openldap squid nfs samba cfengine cacti nagios syncrpl

1. Introducción

Como parte de las tareas de promoción que viene llevando a cabo el Secretariado del Software Libre (SSL) de la Universidad de La Laguna (ULL), hemos fijado el objetivo de intentar garantizar el uso de tecnologías basadas en Software Libre al alumnado de la ULL que así lo desee.

Generalmente, el modo de acceso del alumnado de la ULL es a través de las aulas de informática y laboratorios de su centro o bien a través de aulas de uso genérico que se ubican a lo largo de un campus universitario enormemente disperso. Actualmente, estas aulas son gestionadas de acuerdo a criterios fijados por cada facultad o escuela y lo más habitual es encontrar entornos operativos basados en tecnologías propietarias.

Presentamos en este trabajo el por qué y la necesidad del proyecto SIGA (Sistema Integrado de Gestión de Aulas), así como el estado actual y las dificultades

encontradas. Más específicamente, nos centraremos en las acciones tomadas para centralizar, unificar y migrar las aulas de informática de la ULL a Software Libre.

2. Estado anterior y necesidad del proyecto SIGA

Los centros de la Universidad de La Laguna poseen aulas de informática de uso genérico que pueden estar dedicadas exclusivamente a la docencia o ser de uso libre. Estas últimas no están dedicadas completamente a la formación de los alumnos por lo que pueden ser usadas por estos para realizar sus trabajos, documentarse, etc.

Estas son atendidas y mantenidas únicamente por becarios, alumnos de la ULL, sin el necesario personal de administración, y supervisados por el Vicedecano responsable de acuerdo a criterios fijados por cada facultad o escuela. Esto repercutía negativamente en una gran heterogeneidad.

El proyecto SIGA está homogeneizando las aulas de informática. Se pretende que entre aulas sólo haya diferencias de software específico a cada facultad. De este modo, se facilita el proceso de adaptación de los nuevos becarios o el cambio de los mismos de un aula a otra. Aquellos recién incorporados pueden ser ayudados por otros becarios ya que la configuración del aula es similar, existiendo además la posibilidad de recurrir al propio personal del SSL. Los que han realizado un cambio de puesto de trabajo no notarán grandes diferencias porque sólo existen pequeñas modificaciones en las aplicaciones instaladas. Por otro lado también se beneficiarán los propios usuarios, esto es los alumnos de la ULL. En este nuevo esquema pueden iniciar sesión en casi cualquier aula de la ULL y además obtienen servicios básicos de los que carecían como es una cuenta personal con su propio espacio de disco.

Este proyecto y trabajo son fruto, no sólo de los que constamos como autores, sino del SSL cuyos integrantes son:

- Francisco Almeida Rodríguez (Director)
- Carlos de la Cruz Pinto
- René Martín Rodríguez
- Alberto Morales Díaz
- Jonás Regueira Rodríguez
- Moisés Rodríguez Barrera
- Esaú Rodríguez Sicilia
- Adrián Santos Marrero

3. Arquitectura proyecto SIGA

Las facultades poseen una o más aulas de informática. A cada facultad el proyecto SIGA le presta un servidor y un Sistema de Alimentación Ininterrumpida (SAI) a la que va conectada el servidor.

El nuevo esquema de aulas necesita que los clientes no estén conectados directamente a la red universitaria. Por ello, si el aula no tiene infraestructura de comunicaciones, el proyecto SIGA cede un switch que les permitirá conformar una red local.

El servidor que se presta a cada facultad esta totalmente configurado, comprobado y listo para ponerse en producción. Tiene instalado el sistema operativo Debian y ofrece las siguientes funcionalidades:

Firewall: Con Iptables[1] se implementa un firewall y un enrutador cuando sea necesario.

Proxy: El servidor proporcionado actúa como proxy transparente. Esto se consigue con Squid[2]. Así los usuarios tienen acceso a internet desde los clientes de las aulas pues, como se nombró antes, los clientes en este nuevo esquema no están conectados directamente a la red de la universidad.

Autenticación: Los servidores entregados a cada facultad se sincronizan periódicamente con el LDAP maestro que se encuentra en las dependencias del Centro de Comunicaciones y Tecnologías de la Información (CCTI) mediante «syncrepl». Los servidores de las facultades poseen un LDAP réplica que acelera las consultas y permite que la facultad sea tolerante a caídas de los enlaces al CCTI.

Homes: Cada servidor cedido a cada facultad dispone de dos discos duros de 80GB. De ellos se utilizan aproximadamente 60 GB para el home configurados en RAID-1. Los homes están distribuidos, un servidor almacena y hace accesible los homes de los alumnos que están matriculados en la facultad en la que fue prestado. Por tanto, un alumno que hace login desde un aula de otra facultad recibe su espacio personal desde el servidor de la facultad en la que esta matriculado.

Controlador de dominio o PDC: Los clientes Windows se dan de alta en el dominio creado con Samba en cada servidor de aula.

Con respecto a los clientes del aula, tienen arranque dual, sistemas operativos Linux (Bardinux, que es la distribución oficial de Linux de la ULL) y Windows.

La autenticación es institucional basada en OpenLDAP, tanto en Linux como en los sistemas operativos de Windows. Los alumnos de la ULL cuando se matriculan reciben en sus secretarías un NIU (Número de Identificación Universitario) y una clave de servicio, usadas tradicionalmente para acceder a servicios en puntos de información, y también para el *Correo del Alumnado*[3]. Esta información es la que utilizará el alumno para iniciar sesión en los clientes. Es decir, tanto el LDAP maestro como sus réplicas contienen los NIU y las contraseñas de aproximadamente 40.000 alumnos.

También en el CCTI, el SSL dispone de un servidor desde el cual se administran y monitorizan de forma centralizada los servidores repartidos. Para ello se utilizan las siguientes herramientas software: Cacti[4], Nagios[5] y Cfengine[6].

En resumen, el proyecto SIGA lo conforman un conjunto de aulas con idéntica infraestructura, autenticación de usuarios centralizada, homes compartidos desde el servidor en el que se encuentra matriculado el alumno, y administración y monitorización de todos los servidores centralizada desde el CCTI.

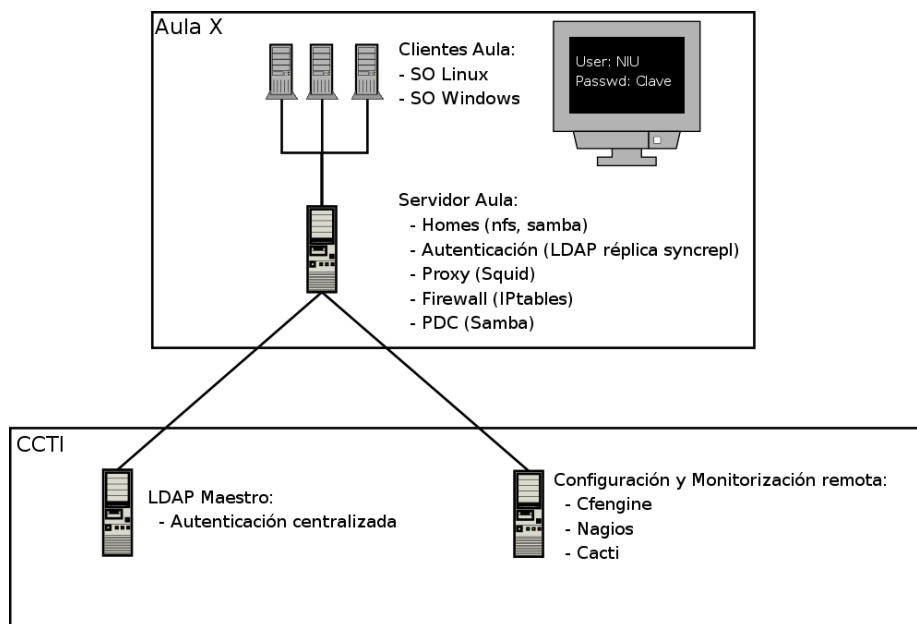


Figura 1. Arquitectura SIGA

4. Sistema de autenticación centralizado OpenLDAP

Ya hemos comentado que los alumnos entran al sistema usando su NIU y su clave de servicio. Para almacenar esta información de autenticación, así como otros datos, como puede ser nombre, apellidos y facultad, se ha decidido utilizar una red de servidores OpenLDAP. En el CCTI se aloja el servidor maestro, y en cada aula un servidor réplica, que sincroniza su información con el primero. En este capítulo explicamos la configuración para hacer esta replicación.

4.1. Configuración del maestro

El LDAP maestro es la pieza más importante del sistema porque es desde la que se harán todas las réplicas. Esto tiene la ventaja de que cualquier cambio que haya que hacer en las cuentas, con hacerlo en el maestro es suficiente. En él damos de alta (y de baja) las cuentas que nos da la secretaría con las nuevas matrículas. También tenemos una interfaz web en Horde[3], de la que hablaremos más adelante, donde los alumnos deben elegir en que *Facultad* están y se guarda en este LDAP. Esta información es un dato necesario para los homes. Los alumnos podrían cambiar también su contraseña, pero no lo permitimos para no

generar discrepancias, ya que hay otros servicios externos a nosotros que usan el NIU y clave de servicio.

La herramienta usada es *syncrepl* en los clientes. Syncrepl es un proceso que abre periódicamente el LDAP de los clientes. Este proceso se conecta al LDAP maestro, verifica que nada haya cambiado, y si cambia, actualiza el LDAP local para dejarlo igual que el maestro. De esta manera, cada uno de los servidores se sincronizan periódicamente con lo que conseguimos clonar el servidor maestro sin esfuerzo.

La mayoría de la configuración de syncrepl va en los LDAP clientes. En el maestro solamente tenemos que crear un usuario *replicador* con permisos de lectura en todo el árbol. En nuestro caso ese usuario es *cn=replicador,dc=ull,dc=es*, y los permisos de lectura deben incluir *userPassword*, ya que las contraseñas hay que replicarlas también. Esto no presenta demasiado problema de seguridad pues se almacenan los hashes de las contraseñas reales.

De la configuración se pueden destacar las siguientes líneas, que es donde se autoriza al usuario replicador a que haga consultas sin el límite por defecto de tamaño ni tiempo, manteniendo los límites para el resto de las consultas de otros usuarios. No hay que olvidar esta peculiaridad, porque el tamaño del árbol es de 40.000 alumnos, y el número de resultados por defecto es de 500.

Extracto de /etc/ldap/slapd.conf del maestro

```
% cambiar por \begin{verbatim} si hay problemas

limits dn.exact="cn=replicador,dc=ull,dc=es"
size=-1
time=-1
```

4.2. Configuración de las réplicas

Aparte de la configuración particular de schemas que tengamos, que debe incluir el de samba, y de los accesos (*access to*), existe en cada réplica una configuración llamada *syncrepl* que es la que nos permitirá conectar a nuestro servidor maestro para recoger los cambios.

Extracto de /etc/ldap/slapd.conf de la réplica

```
% cambiar por \begin{verbatim} si hay problemas

syncrepl rid=1
provider=ldap://ldap2.ccti.ull.es
type=refreshAndPersist
interval=00:00:10:00
searchbase="dc=alumnado,dc=ull,dc=es"
filter="(objectClass=*)"
```

```

scope=sub
attrs="*"
schemachecking=off
updatedn="cn=admin,dc=ull,dc=es"
bindmethod=simple
binddn="cn=replicador,dc=ull,dc=es"
credentials=aqui-va-una-contraseña-real

```

En este caso, el LDAP réplica se conecta al LDAP maestro como *cn=replicador,dc=ull,dc=es* (*binddn*), con la contraseña que se ponga en *credentials*, y realiza cambios en sí mismo como usuario *cn=admin,dc=ull,dc=es* (*updatedn*). Se conecta cada 10 minutos (*interval*) a la máquina *ldap2.ccti.ull.es* (*provider*).

4.3. Ramas locales

Puede darse el caso de que en cada facultad personas no matriculadas en la universidad necesiten usar los ordenadores, o visitantes de postgrados, o alumnos recién matriculados antes de que la secretaría nos facilite las altas y bajas. En estos casos, se contempla la necesidad de que los becarios de aula puedan crear y administrar cuentas temporales en sus facultades. Dada la temporalidad de estas cuentas, no tiene sentido crearlas en el LDAP maestro, por tanto se habilitan ramas locales en el LDAP de cada facultad para que sean administradas por los becarios. Estas ramas locales sólo son visibles en la propia facultad, no desde el resto.

Dentro de la formación que se da a los becarios se incluyen instrucciones para la creación de estas cuentas así como los homes correspondientes, aunque se están preparando interfaces web para simplificar esta gestión.

4.4. Clientes Linux

En los clientes Linux tenemos que modificar el comportamiento de PAM para que consulte en el LDAP. Se ha seguido la documentación oficial de PAM, que la podemos encontrar en la guía para administradores[7]. Los pormenores de nuestra configuración para las aulas, se pueden encontrar en un HOWTO que hemos escrito[8].

En resumidas cuentas, basta editar los archivos de configuración en */etc/pam.d/* para que autentifique por *pam_ldap* antes que por *pam_unix*, y luego editar los datos de acceso al servidor LDAP (IP, usuario, contraseña, etc.) en */etc/pam_ldap.conf*.

Con la configuración de PAM funcionando, configuraremos NSS, que es un servicio que se encarga de hacer resolución de usuarios y grupos, no con los ficheros */etc/passwd* y */etc/group* como un Unix habitual, sino por LDAP. Esto nos permitirá que los números UID de los usuarios y los GID de los grupos se traduzcan a nombres usando la información del LDAP.

4.5. Clientes Windows

Los clientes Windows tenemos que unirlos al dominio para que consulte el LDAP. El samba configurado en el servidor entregado a la facultad hace de controlador de dominio.

Con Samba y OpenLDAP como backend, usamos tecnología libre para interactuar con los clientes Windows y evitamos utilizar un Active Directory.

5. Sistema de almacenamiento distribuido

Los homes de los usuarios se encuentran distribuidos. El home de un alumno se encuentra disponible en todas las aulas de la Universidad de La Laguna en la que se ha implantado el proyecto SIGA pero físicamente sólo se encuentra en los discos duros del servidor de la facultad en la que el alumno está matriculado.

Los homes en el sistema operativo Linux se exportan por NFSv3. En el sistema operativo Windows están disponibles gracias a Samba. La figura 2 muestra como es el proceso. El LDAP, además del NIU y la clave de servicio, contiene un atributo que indica en qué facultad está el home del alumno. Como se observa en el gráfico, pueden darse cuatro casos:

- Alumno *A* matriculado en la facultad *X* que inicia sesión en Linux en un aula de la facultad *X*: En este caso, el alumno tiene acceso a su home por NFS desde el propio servidor del aula en el que se encuentra.
- Alumno *B* matriculado en la facultad *X* que inicia sesión en Windows en un aula de la facultad *X*: En este caso, el alumno tiene acceso a su home por Samba desde el propio servidor del aula en el que se encuentra.
- Alumno *C* matriculado en la facultad *X* que inicia sesión en Linux en un aula de la facultad *Y*: En este caso, el alumno tiene acceso a su home por NFS desde el servidor de la facultad en la que esta matriculado (*X*, no *Y*).
- Alumno *D* matriculado en la facultad *X* que inicia sesión en Windows en un aula de la facultad *Y*: En este caso, primero se exporta por NFS el home desde el servidor de la facultad *X* al servidor de la facultad *Y*. En segundo lugar, el servidor *Y* hace que el home del alumno *D* esté accesible en Windows con Samba.

6. Servidores en las aulas

Se ha hecho la inversión de adquirir para el proyecto 22 servidores de aula con las siguientes características:

- Procesador: Intel® Pentium® 4 521, 2.8GHz, 1MB L2 cache, 800 MHz FSB
- Memoria: 1GB DDR2 SDRAM (2x512MB Single rank DIMMS) 533 MHZ
- 2 Discos duros: Disco duro 80GB SATA a 7200 rpm
- Tarjeta de red: Broadcom NetXtreme 10/100/1000 PCI Express

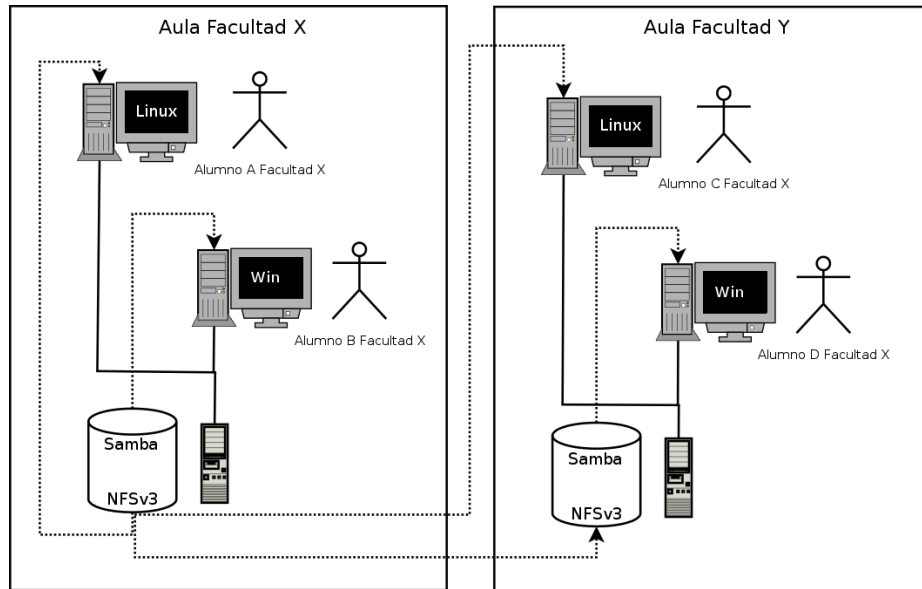


Figura 2. Esquema almacenamiento distribuido

- Dispositivos ópticos fijos: 48x CD-ROM, Interna

A cada servidor se asocia también un SAI de modelo *APC Smart-UPS 750i 480W*, para dar cierta tolerancia al servicio ante caídas del suministro eléctrico. También se adjunta un Switch Gestionable *PowerConnect Switch: PC 3424 Managed Stackable Switch* (de 24 Fast Ethernet switching ports + 2 Gigabit Ports + 2 SFP Ports), dedicado a interconectar los clientes del aula, en caso de que el aula no disponga de electrónica de red actualizada.

En los servidores se ha instalado Debian Etch, la próxima estable, ya que la actual estable (Sarge) no soportaba adecuadamente el hardware. Se ha hecho una instalación base usando el CD de netinst, con los repositorios oficiales de Debian. Seguidamente se instalan los paquetes relacionados con los servicios de la máquina:

Instalación paquetes relacionados con los servicios

% cambiar por `\begin{verbatim}` si hay problemas

```
apt-get install slapd samba samba-doc smbldap-tools libnss-ldap nscd \
db4.2-util nfs-common nfs-kernel-server ldap-utils ntpdate smbclient \
metamail vim gpm lynx
```

Después se configuran los servicios que ya hemos mencionado en el aula:

- Firewall

- Proxy
- Autenticación
- Homes
- Controlador de dominio o PDC

La administración y la monitorización de estos servidores es centralizada y realizada de forma remota por los becarios del SSL desde las dependencias del CCTI. La administración se hace con Cfengine[6]. La monitorización se realiza con Cacti[4] y Nagios[5].

7. Clientes en las aulas

En esta fase del proyecto se ha decidido aportar únicamente el hardware comentado anteriormente. Es por ello que se mantienen los clientes que previamente se encontraban en las aulas, en su mayoría equipos DELL Pentium IV a 1'8 GHZ con 256 megas de RAM y 40 GB de disco duro.

Se ha realizado un análisis de las demandas de las diferentes titulaciones encontrando un núcleo importante de aplicaciones propietarias en algunos dominios específicos que nos impiden realizar el cambio completo a Software Libre.

Los clientes del proyecto, poseen arranque dual, sistemas operativos Linux y Windows (versiones 98, 2000 y XP). En el caso de Windows se ha promocionado y recomendado el uso de Software Libre como por ejemplo el uso de Firefox como navegador y Thunderbird como cliente de correo.

La distribución Linux instalada es Bardinix. Bardinix es la distribución GNU/Linux oficial de la ULL. Está desarrollada por el SSL que se encarga de darle soporte y mantenerla actualizada. Esta basada en una Kubuntu Dapper, por lo que cualquier duda o problema con esta distribución puede ser resuelta consultando la documentación y foros de Ubuntu[9]. No obstante, gracias a un convenio de la ULL con GULIC[10] (Grupo de Usuarios de Linux de Canarias), Bardinix dispone de su propio foro[11] donde los usuarios pueden obtener respuesta a sus dudas. En la distribución se han tenido en cuenta las necesidades de la mayoría de los universitarios por lo que las aplicaciones disponibles son muy variadas:

- Suite ofimática: OpenOffice.
- Navegadores: Mozilla Firefox.
- Clientes de correo: Mozilla Thunderbird.
- Diseño gráfico: Inkscape, Scribus, Blender, Yafaray.
- Reproductores de audio: Amarok.
- Retoque fotográfico: Gimp.
- Y un largo etc.

Los responsables de instalar y administrar los clientes son exclusivamente los becarios de las aulas. El SSL ha creado metapaquetes con el software libre que necesita una determinada facultad (bardinix-física, bardinix-informática, bardinix-matemáticas,...), una lista de correo, documentación, así como un wiki

en el que los becarios tienen acceso de lectura y escritura para que ellos mismos generen documentación y puedan resolver sus propias dudas consultando la información existente.

8. Conclusiones

Este proyecto se acaba de implantar en algunas aulas, que citamos a continuación, y está proveyendo a los becarios y a los alumnos que las usan las ventajas de una gestión centralizada.

Aunque actualmente está en fase de desarrollo creemos que nuestra propuesta podría ser adaptada en universidades en las que se siguen modelos de gestión similares en las aulas de informática.

8.1. Estado actual del despliegue

Actualmente hemos conseguido implantar SIGA en las siguientes aulas:

- Aula del DSIC situada en el edificio de la ETSII (Escuela Técnica Superior Ingeniería Informática) utilizada para las prácticas del departamento.
- Aula de la Facultad de Filología.
- Aulas de la Facultad de Física.
- Aula de CajaCanarias en Guajara. Aula de 100 ordenadores en la que exclusivamente está instalado el sistema operativo Linux. Esto supuso un importante ahorro en licencias y demuestra que hay motivación y se está apostando por el uso de Software Libre.
- Aula de la Facultad de Química.

8.2. Dificultades encontradas

Enumeramos las dificultades que hasta el momento hemos encontrado en el despliegue:

- Enlace RDSI Náutica. La Facultad de Náutica está comunicada por una línea RDSI que imposibilita realizar la sincronización del LDAP réplica del servidor de las aulas con el maestro situado en las dependencias del CCTI.
- Aulas de una misma facultad separadas geográficamente en distintos edificios. La solución es crear VLANs que las conectan con su servidor de aula.
- Para que los clientes Windows autentificasen correctamente en el controlador de dominio, tuvimos que desarrollar una DLL[12] con rutinas para generar, a partir de la clave de servicio original, los hashes utilizados por las diferentes versiones de Windows para autenticarse.
- El LDAP maestro no posee información sobre en que facultad está matriculado el alumno. Se ha solucionado este problema mediante un módulo para Horde en el que el alumno selecciona su facultad. Este módulo se ha integrado en Horde ya que los alumnos también disponen de una cuenta de correo institucional al que pueden acceder mediante el *Webmail*[3] del mismo proyecto. Este servicio también hace uso del nombre de usuario (NIU) y contraseña (clave de servicio) ya mencionadas anteriormente.

- Aulas de informática gestionadas por administradores. Este sólo es el caso del centro de cálculo de la facultad de informática.
- Aulas que ya poseen cuentas de usuarios y que en principio no ganan tanto como las otras son más reticentes al cambio.

9. Agradecimientos

Agradecemos a la Consejería de Industria, Comercio y Nuevas Tecnologías del Gobierno de Canarias su financiación parcial de este trabajo mediante el proyecto “Aguaviva”.

Referencias

1. Página oficial del proyecto Netfilter/iptables Url: <http://www.netfilter.org/>
2. Web oficial de Squid Web Proxy Cache. Url: <http://www.squid-cache.org/>
3. Correo del Alumnado de la Universidad de La Laguna Url: <http://alumnado.u11.es>
4. Página oficial. Url: <http://cacti.net/>
5. Sitio oficial. Url: <http://www.nagios.org/>; Documentación oficial. Url: <http://nagios.org/docs/>
6. Steve Kemp: Artículo “A simple overview of Cfengine”. Url: <http://www.debian-administration.org/articles/223>; Artículo “A introduction to cfengine rules”. Url: <http://www.debian-administration.org/articles/224>; Artículo “Using cfengine in a client/server setup”. Url: <http://www.debian-administration.org/articles/225>
7. Andrew G. Morgan, Thorsten Kukuk: The Linux-PAM System Administrator’s Guide. Url: http://www.kernel.org/pub/linux/libs/pam/Linux-PAM-html/Linux-PAM_SAG.html
8. HOWTO Instalación y configurar el servidor de aulas. Url: <http://ssl.u11.es/node/41>
9. Página web oficial de Ubuntu. Url: <http://www.ubuntu.com>
10. Página web de GULIC, Grupo de Usuarios de Linux de Canarias. Url: <http://www.gulic.org>
11. Foro de atención al usuario de la distribución GNU/Linux Bardinix. Url: <http://bardinix.u11.es>
12. Hashes de Windows para SQL Server. Url: <http://ssl.u11.es/node/184>

siLeDAP: Implementando XLDAP

Javier Masa y Cándido Rodríguez

RedIRIS

{javier.masa,candido.rodriguez}@rediris.es

<http://www.rediris.es>

Resumen Los servicios de directorios aun siendo, hoy en día, una de las bases más utilizadas en los entornos tecnológicos no proporcionan toolkits o frameworks significativos alrededor de ellos que faciliten su interoperabilidad con las aplicaciones desplegadas en dicho entorno. Además, la problemática aumenta en el momento que se trata de aplicar tecnologías más complejas en el campo del directorio, como COPA. siLeDAP tiene como objetivo desarrollar no sólo un navegador/administrador de directorio avanzado, aprovechando las características de las aplicaciones web 2.0, sino también crear un conjunto de clases y servicios web que permitan que las aplicaciones puedan aprovechar toda la potencia que ofrecen los servicios de directorio. En especial, su desarrollo se enfoca en implementar el protocolo XLDAP que servirá para comunicarse con directorios utilizando mensajes XML.

Palabras clave: servicios de directorio, servicios web, XML, SOAP

1. Introducción

Aunque se extiende cada vez más la implantación de servidores de directorio en los distintos entornos tecnológicos que nos podemos encontrar, el mayor handicap que existe para su crecimiento positivo es la inflexibilidad que ofrecen estos tipos de plataformas.

El objetivo del proyecto siLeDAP [1] es desarrollar una aplicación web y un conjunto de servicios web que faciliten la administración de servidores LDAP, resolviendo la problemática anterior, teniendo claro que debe estar diseñado para facilitar la aplicación de diferentes líneas de investigación en este campo, como puedan ser las vistas virtuales (COPA) [2] sobre el directorio.

En una de sus líneas de actuación, siLeDAP pretende ofrecer un entorno de comunicación y trabajo con el servidor de directorio más transparente y ligero, de manera que pueda integrarse con las aplicaciones distribuidas que conviven a su alrededor. Esto es posible gracias al desarrollo de un conjunto de APIs que pueden ser utilizadas a través de servicios web de tipo REST [3] o SOAP [4].

2. Arquitectura del sistema

El desarrollo de siLeDAP, enfocado a la provisión de operaciones a través de servicios web, permite que las operaciones triviales, como hacer consultas en un directorio o modificar una entrada de ésta, y las operaciones más complejas, como la integración de COPA en el directorio, sean alcanzables desde cualquier aplicación que realice peticiones HTTP al conjunto de servicios web. Además, sea cual sea el lenguaje de programación utilizado para nuestras aplicaciones, pueden aprovechar las capacidades que ofrecen las APIs. Generalmente, cada uno de los servicios web tiene una actuación delimitada, casi atómica, como añadir una entrada o eliminarla, las cuales especifican en su documentación si en la petición HTTP se admite que se pasen parámetros por POST o por GET, así como la especificación que deberán tener. La respuesta que generaran éstas será siempre en XML, facilitando la interacción con cualquier tipo de aplicación y permitiendo que aplicaciones web utilicen tecnología AJAX para comunicarse con el servidor de directorio.

En el siguiente diagrama podemos ver la arquitectura general de siLeDAP:

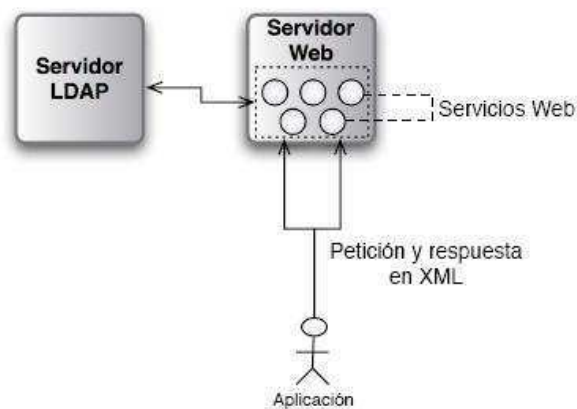


Figura 1. Arquitectura de siLeDAP

Como vemos, los servicios web son instalados en un servidor web, con la única restricción que se le exige es que tenga compatibilidad con PHP 4, puesto que están programados con dicho lenguaje. Tal como hemos comentado, al realizar la aplicación todas las operaciones con el directorio a través de los servicios web, esto permite independizar el lenguaje de programación utilizado.

Para configurar correctamente dichos servicios, necesitaremos especificar en un archivo la siguiente información:

- IP/Nombre del servidor LDAP y su puerto.
- Si es una conexión anónima o el DN y contraseña del usuario.

- DN base al realizar las peticiones.

3. Servicios web proporcionados

Los servicios web en siLeDAP sólo admiten actualmente peticiones bajo el formato REST [3]. Se ha optado inicialmente por este formato ya que es el tipo más simple existente y el que se utiliza en las aplicaciones web que trabajan con tecnología AJAX. De cualquier forma, se espera que en futuro admita peticiones tipo XML-RPC [9] o SOAP [4]. La estructura de una petición REST es básicamente una petición GET con parámetros bajo HTTP. Cada uno de los servicios web especifica qué parámetros admite y los valores válidos. Por ejemplo, para solicitar una entrada en concreto al directorio, utilizaríamos la siguiente URL:

```
http://servidor_web/siledap/API/getEntry.php?dn=
ZGM9cmVkaXJpcyxkYz1lcw==
```

Como vemos, le pasamos un parámetro, dn, con el DN de la entrada codificado en base 64.

Por otro lado, toda petición REST es respondida por una respuesta REST, siendo ésta un mensaje XML que contiene una serie de elementos particulares en cada servicio web. Es decir, cada uno de los servicios especifica en su documentación la estructura del mensaje y su contenido. En el caso de nuestro ejemplo anterior, la respuesta que obtendremos es:

```
<?xml version='1.0' encoding='UTF-8'?>
<Entries>
<Entry dn="uid=persona,dc=rediris,dc=es">
<Attribute name="cn">
<AttributeValue value="Manolito perez"/>
</Attribute>
<Attribute name="description">
<AttributeValue value="Currante"/>
</Attribute>
</Entry>
</Entries>
```

Actualmente se está trabajando en proveer con más tipos de mensajes de respuesta como JSON [10], SOAP o XML-RPC.

La API de siLeDAP proporciona, a día de hoy, los siguientes servicios web:

- *Búsquedas en LDAP*: este servicio web tiene como objetivo realizar las operaciones necesarias para realizar búsquedas, a las cuales podemos especificar el filtro de búsqueda y el alcance de ellas.
- *Actualizaciones en LDAP*: se encarga de realizar las operaciones de escritura en el LDAP como añadir, reemplazar o eliminar entradas en él.
- *Información del esquema*: tiene como objetivo proporcionar la información relativa al esquema almacenado en el directorio como obtener la definición de una clase de objeto (objectClass) o de un atributo.

- *Obtener una entrada*: obtiene de una entrada todos los pares atributos/valores del DN solicitado. Aquellos atributos que sean de tipo binario dará su valor codificado en base 64.
- *Importar/exportar*: este servicio web es capaz de importar o exportar en el formato LDIF.
- *Integración COPA*: se encarga de comunicarse con el servidor LDAP aplicando la tecnología COPA, pudiendo de esta manera navegar utilizando vistas virtuales.

La decisión de soportar inicialmente sólo las peticiones y respuestas de tipo REST es debido a que es la tecnología base que nos permite facilitar la integración de un servidor de directorio en las nuevas aplicaciones que utilizan la tecnología AJAX. Como ejemplo, vamos a programar en nuestra aplicación web que solicite por AJAX la lista de entradas que hay a partir de un DN base determinado. Esta acción se realizará a través de JavaScript con el siguiente método:

```
function updateChildsInfo() {
var url=urlWebServices + "searchldap.php?dnbase=" +
encode64(dnbase) + "&scope=one";
var request = getXMLHttpRequest();
request.open('GET', url, true);
request.onreadystatechange = function() {
if (request.readyState == 4) {
var res="";
var xml=request.responseXML;
var entries=xml.getElementsByTagName("Entry");
for (var i=0; i<entries.length; i++) {
var dnEntry=entries[i].getAttribute("dn");
processEntry(dnEntry);
}
}
}
request.send(null);
}
```

4. XML Enabled Directory

En el mes de Noviembre del 2006, the Internet Engineering Task Force (IETF) ha publicado una serie de Internet-Drafts sobre el XML Enabled Directory (XED) [5], iniciativa para trabajar en XML con los directorios de servicio. El protocolo de comunicaciones que se utilizará será XML Lightweight Directory Access Protocol (XLDAP) [6], el cual en su primera versión es semánticamente equivalente a LDAP versión 3. XED define dos métodos de transporte para los mensajes XLDAP: sobre TCP/IP y sobre SOAP.

En el draft presentado en IETF se especifican los distintos tipos de operaciones que se pueden realizar en mensajes XLDAP. Éstos están codificados

utilizando *Robust XML Encoding Rules* (RXER) [7], el cual es básicamente una definición de Abstract Syntax Notation One (ASN.1) en formato XML. De esta forma, a partir de la definición en ASN.1 de la versión 3 de LDAP y utilizando RXER en el draft de XLDAP se especifican todos los mensajes posibles.

Como ejemplo, podemos ver a continuación un mensaje de búsqueda en XLDAP:

```
<xed:LDAPMessage
xmlns:xed="urn:ietf:params:xml:ns:xed"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xldap="urn:ietf:params:xml:ns:xed-ldap"
xsi:type="xldap:LDAPMessage">
<messageID> 0 </messageID>
<protocolOp>
<searchRequest>
<baseObject>
<!-- DN es dc=example,dc=com -->
<item>
<item><!-- dc=com -->
<type>0.9.2342.19200300.100.1.25</type>
<value>com</value>
</item>
</item>
<item>
<item><!-- dc=example -->
<type>0.9.2342.19200300.100.1.25</type>
<value>example</value>
</item>
</item>
</baseObject>
<scope>wholeSubtree</scope>
<derefAliases>derefInSearching</derefAliases>
<sizeLimit>100</sizeLimit>
<timeLimit>5</timeLimit>
<typesOnly>>false</typesOnly>
<filter>
<and>
<filter> <!-- objectClass = person -->
<equalityMatch>
<attributeDesc>
<type>2.5.4.0</type>
</attributeDesc>
<assertionValue>2.5.6.6</assertionValue>
</equalityMatch>
</filter>
<filter> <!-- surname = Smith -->
```

```

<equalityMatch>
<attributeDesc>
<type>2.5.4.4</type>
</attributeDesc>
<assertionValue>Smith</assertionValue>
</equalityMatch>
</filter>
</and>
</filter>
<attributes><!-- all attributes --></attributes>
</searchRequest>
</protocolOp>
</xed:LDAPMessage>

```

5. Conclusiones

Aun cuando estamos todavía en las etapas iniciales del desarrollo del proyecto siLeDAP, preveemos un futuro prometedor para esta iniciativa, puesto que no sólo proporcionará una herramienta moderna y avanzada para navegar y administrar un servicio de directorio, sino que su conjunto de APIs implementados a través de servicios web va a facilitar la aparición de aplicaciones que se integren con tecnologías tipo COPA, puesto que no va a ser necesario que aquellas tengan que implementarlas.

Además, la realización en colaboración con RedIRIS de la versión 3 de su sencillo navegador LDAP vía web, Navega [11], ha permitido realizar el primer navegador de directorio íntegramente con HTML y JavaScript, lo cual permite demostrar el potencial que ofrecen los servicios web proporcionados con el software.

Con respecto a las características de seguridad de XLDAP, éste al ser una implementación de los mensajes de LDAP versión 3 en XML, conlleva sus mismas capacidades de seguridad. De cualquier forma, la implementación de XLDAP sobre SOAP permite enriquecer dichas capacidades a través de los estándares de seguridad sobre Servicios Web [8].

Referencias

1. siLeDAP. <https://forja.rediris.es/project/siledap>
2. Esquema COPA. <http://www.rediris.es/ldap/esquemas/copa.schema>
3. Fielding, R.: Architectural Styles and the Design of Network-based software Architectures. http://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm
4. Gudgin, M., Hadley, M., Mendelsohn, N., Moreau, J.J., Nielsen., H.F.: Simple Object Access Protocol (SOAP) 1.2. <http://www.w3.org/TR/soap12>
5. Legg, S., Prager, D.: The XML Enabled Directory. draft-legg-xed-roadmap-xx.txt, a work in progress, October 2006.
6. Legg, S., Prager, D.: The XML Enabled Directory: Protocols. draft-legg-xed-protocols-xx.txt, a work in progress, November 2006.

7. Legg, S., Prager, D.: Robust XML Encoding Rules (RXER) for Abstract Syntax Notation One (ASN.1). draft-legg-xed-rxer-xx.txt, a work in progress, October 2006.
8. Nadalin, A., et al.: Web Services Security: SOAP Message Security 1.1 (WS-Security 2004). OASIS Standard. <http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.1.pdf>
9. XML-RPC Specification. <http://www.xmlrpc.com/spec>
10. RFC 4627: The application/json Media Type for JavaScript Object Notation (JSON). <http://tools.ietf.org/html/rfc4627>
11. Navega. <http://www.rediris.es/ldap/software/navega>

Primer Concurso Universitario de Software Libre

P. Neira, A. Rey Botello, F.J. Jordano, and R. Tovar

ETS Ingeniería Informática - Avda. Reina Mercedes, s/n - 41012 SEVILLA
{pneira, arey, arcturus, rafaeltovar}@us.es

Abstract. Hoy día las empresas comienzan a demandar cada vez más profesionales expertos en Software Libre. La administración está adoptando soluciones basadas en Software Libre con éxito, como es el caso de Andalucía y Extremadura, y existe un compromiso firme de mantener esta tendencia en los próximos años, por lo tanto, es previsible que la tendencia ascendente de esta demanda se mantenga. Por otro lado, el Software Libre se presenta como un complemento perfecto en la formación de los estudiantes universitarios, puesto que les da acceso a una tecnología puntera, que puede resultar inalcanzable debido a las restricciones que imponen las soluciones propietarias, y que puede ser útil en la obtención de experiencia en el proceso de desarrollo de software en etapas previas a la inserción en la vida laboral. El principal objetivo del concurso universitario de software libre es estimular a los estudiantes universitarios para que se involucren en la participación y creación de proyectos de Software Libre, generando así beneficios tanto a escala individual, como profesionales, como colectiva, al conjunto de la sociedad. De esta forma, se crearán las condiciones idóneas para generar un tejido tecnológico de futuros profesionales que serán capaces de dar soporte de soluciones basadas en Software Libre a colectivos, empresas y administración.

Key words: Software Libre, Universidad, Concurso

1 Introducción

El movimiento del Software Libre surgió a mediados de la década de los ochenta como respuesta al modelo privatizador de software que comenzó a triunfar a principios de dicha década.

Las bases de este movimiento proclaman la libertad sobre el derecho de uso, estudio, copia, modificación y redistribución del software [2], adquiriendo el Software Libre la categoría de filosofía puesto que defiende una serie de valores éticos (libertad, colaboración, seguridad, privacidad, eficiencia, no discriminación, calidad) desde el nacimiento del fenómeno que sin duda trasciende la barrera de lo meramente técnico.

Si bien fueron unos cientos de expertos informáticos los que dieron vida al movimiento, su expansión ha sido fulgurante tanto en el número de desarrolladores como, y quizás esto sea más importante, en el número de usuarios

y usuarias, pasando a convertirse una alternativa real al software propietario. Además del crecimiento exponencial en el número de personas involucradas con el Software Libre, han nacido otros movimientos íntimamente relacionados con el software libre como por ejemplo el movimiento Open Source [12] que comparte las raíces y que ayuda a incrementar aún más el número de usuarios que comulgan con los ideales del software libre.

Gracias a la naturaleza del Software Libre, su carácter divulgativo, gratuito y educativo, han hecho que se convierta en una herramienta perfecta, al alcance de todo el mundo, para la difusión del conocimiento. Observamos el máximo exponente de lo anteriormente comentado en los centros educativos donde se ha creado una simbiosis perfecta entre las dos entidades. Sirviendo el software libre como estímulo inigualable para el desarrollo personal y profesional de estudiantes y educadores.

Por otra parte, el tejido empresarial al disponer de una masa social importante de desarrolladores y usuarios, y junto con el modelo de negocio alternativo que propone el Software Libre, ha dado el salto y se ha adaptado a la nueva filosofía, constituyendo sin duda uno de los mayores aportadores a la comunidad.

Por último y como paso natural, las administraciones públicas han tomado conciencia de la revolución social que plantea el Software Libre y la están adoptando para mejorar no sólo el ámbito relacionado con los usuarios de software, sino como solución global para el conjunto completo de la sociedad.

2 Situación Actual

La situación actual es prometedora, la administración está dando pasos en pro de la adopción de soluciones de Software Libre (SL), así lo reflejan las iniciativas de distribuciones de software libre autonómicas tales como Linex [8] y Guadalinux [6] entre muchas otras. Además, la administración tiende a implantar soluciones basadas en software libre para ofrecer servicios a los ciudadanos, tales como el correo del ciudadano ofrecido por la Junta de Andalucía. Es por ello que en el ámbito de las empresas locales se está produciendo un cambio en el modelo de negocio de soluciones de software propietario a SL promovido por la demanda de la administración.

Por el momento, estos cambios afectan mayormente al uso e implantación, no al desarrollo de Software Libre como tal. Las empresas locales que actualmente dan soporte tienden a emplear soluciones libres existentes, a adaptarlas a sus necesidades, y a revertir poco sobre la comunidad de desarrolladores. La liberación de software aún genera incertidumbre en el tejido empresarial, el modelo de obtención de beneficios mediante soporte a terceros es y ha sido muchas veces cuestionado.

En resumidas cuentas, por el momento los cambios provienen en su mayoría de estratos superiores. Las reformas introducidas son desarrolladas desde la administración hacia la sociedad, con los riesgos que ello supone. En la historia de España este patrón es de sobra conocido, desde principios de siglo XX los sucesivos gobiernos hasta el comienzo de la guerra civil introdujeron numerosas reformas, muchas de ellas sin contar con el apoyo de la sociedad, resultando en cambios efímeros dependientes de la política partidista del momento [17].

Si bien es cierto que la administración está tomando medidas de base para que produzcan los cambios sociales oportunos, como se trata de la obligatoriedad del uso de soluciones de Software Libre en centros de educación primaria, secundaria y oposiciones al cuerpo funcionarial [7], se corre el riesgo de que dichos cambios sean asimilados como la tendencia del momento. Es por ello, que la masa social que sustenta el software libre, también conocida como Comunidad, debe ser reforzada para dar el soporte que necesita la sociedad y hacer que el cambio se convierta en una realidad.

Por otro lado, con el fin de evaluar el estado actual de la comunidad hemos adoptado una serie de indicadores que nos permitan comparar la fortaleza de la comunidad nacional con respecto a la de otros países europeos. Concretamente, el proyecto Gnome cuenta con un 50% menos de desarrolladores, traductores y artistas en España con respecto a países como Francia y Alemania [5]. En el proyecto Debian [1], España ocupa la décimosexta posición en el índice de desarrolladores por millón de habitantes, una posición digna, aunque lejos de otros países europeos punteros en materia tecnológica.

En cuanto a eventos de referencia, los de mayor asistencia en España son organizados por la administración, tales como el Free Software/Open Source World Conference [4], frente a otras iniciativas, tales como el LinuxTag alemán [9] que surgen de la base social de la comunidad del Software Libre en este país.

Con respecto a la situación en las universidades españolas, ya existen experiencias en funcionamiento que tienen como fin la divulgación y el fomento del Software Libre, tales como las oficinas de software libre de la Universidad de Cádiz [11] y Las Palmas [10] que están realizando una labor fundamental de transformación local en su entorno académico universitario.

3 Motivaciones

Son múltiples los beneficios asociados al Software Libre:

3.1 Desarrollo de la economía local

El uso de soluciones propietarias crea dependencia hacia factores foráneos, lo que nos convierte en meros consumidores de tecnología producida en el exterior de la

que desconocemos por completo los detalles de implementación. Los acuerdos de uso de software propietario adquiridos en muchas universidades españolas permiten a los estudiantes la utilización de dicho software sin coste alguno durante la etapa de formación. Sin embargo, a la larga genera un colectivo de profesionales que serán dependientes de dichas soluciones. Lo que se acabará traduciendo en fugas de capital en concepto de formación y pago de licencias. Así, las empresas locales obtienen un margen de beneficio reducido en concepto de implantación, frente a los márgenes obtenidos por el productor y distribuidor del software, perpetuando un tejido de empresarial frágil totalmente dependiente del extranjero cuya base es el "Nosotros distribuimos, implantamos y damos soporte, y el productor se lleva el grueso de los beneficios".

3.2 Beneficios educativos

Desde el punto de vista docente universitario, la enseñanza debe mantener su naturaleza teórica neutral que ha de ser complementada con el conocimiento de una serie de tecnologías contemporáneas existentes. Es común que surga literatura teórica sobre dichas tecnologías con fines docentes, tales como manuales y referencias técnicas. Concretamente, en el caso del software propietario, el estudio de dichos manuales supone un acto de fé, ya que no es posible constatar si la fuente es correcta y precisa, lo que se traduce en ciertas situaciones en manuales manipulados por el fabricante, con fines claramente comerciales, que no reflejan la realidad, dichas prácticas son conocidas como *Fear, Uncertainty and Doubt (FUD)*, en castellano, miedo, incertidumbre y dudas [16] [20].

Desde el lado del alumnado, el Software Libre se presenta como un campo de pruebas idóneo, una oportunidad para adquirir experiencia real en el desarrollo de software sin necesidad de involucrarse en el ámbito laboral, una actividad difícilmente compatible con el desarrollo de unos estudios universitarios. Es por ello que el software libre da la posibilidad de complementar la formación recibida en la universidad, todo ello mediante la realización de contribuciones a proyectos de software libre. Dan Kegel, ex-ingeniero de Google, afirma [15] que la mayoría de los estudiantes de ingeniería informática con la titulación recién obtenida afirman no tener experiencia real más allá de las prácticas de la universidad. Por otro lado, Joel Spolsky [18] afirma que el perfil general de candidato a cubrir un puesto vacante en una empresa se caracteriza por personas que sean capaces de completar proyectos, es decir, que ya hayan superado anteriormente otros proyectos que estén relacionado con la posición que van a cubrir. Ambos autores afirman que el contratante tiene una forma efectiva de comprobar lo anterior sin necesidad de mirar al CV, para ello basta con hacer uso de un buscador como Google para comprobar qué ha hecho el candidato anteriormente. De esta forma, en caso de haber contribuido a algún proyecto de software libre, el contratante puede obtener el código que has programado, puesto que está disponible en la red, así como obtener información sobre tu reputación como diseñador e implementador de software.

Muchos son los beneficios que un estudiante puede obtener del desarrollo de Software Libre, tales como:

- Aprender de otros desarrolladores con experiencia contrastada en el desarrollo de software.
- Aprender el uso de herramientas potentes en el desarrollo de software tales como gestores de código, sistemas de bugtracking ...
- Desarrollo de capacidad de liderazgo. Este valor puede tener connotaciones negativas muchas de ellas asociadas al perfil de jefe, una posición que no otorga mágicamente liderazgo. El liderazgo es la capacidad de convencer a los demás de que las decisiones que tomas en un proyecto son apropiadas, valiéndose para ello de experiencia y conocimientos, por tanto, dando argumentación sólida a las decisiones tomadas. Es por ello que se puede considerar que el Software Libre también ayuda al desarrollo de habilidades sociales derivadas del trato con usuarios y desarrolladores.
- Aprender a trabajar en grupo

Para concluir, Richard M. Stallman [19] afirma que para aprender a producir código de calidad, un estudiante debe haber leído y escrito mucho código previamente.

3.3 Beneficios para la sociedad

Los beneficios que el Software Libre presenta a la sociedad son muchos, van desde la materialización del conocimiento humano en software que podrá ser reutilizado, y por tanto, acabar con la constante reinención de la rueda, hasta la generación de un tejido local de expertos en tecnología que podrán asesorar al conjunto de la sociedad.

Otro punto importante es el impulso del desarrollo local que puede promover el uso y el desarrollo del Software Libre, en un mundo globalizado en el que el capital circula de un punto a otro del globo de la noche a la mañana, la creación de una industria local fuerte puede garantizar una mayor independencia y estabilidad en la zona.

Además, el Software Libre lleva consigo también una serie de valores positivos inherentes que benefician al conjunto de la sociedad, tales son la conciencia de colectivo, la colaboración, el fomento de la creatividad e innovación, y la solidaridad entre pueblos.

4 Concurso Universitario de Software Libre

La administración y empresas tienen previsiones de incrementar el volumen de implantaciones de Software Libre en los próximos años. La creciente demanda de profesionales expertos en soluciones de software libre corre el riesgo de no ser

cubiertas con éxito lo que podría suponer una ralentización del proceso por falta de personal cualificado.

En un reciente estudio de Infonomics [14], se señala que el 70% de los desarrolladores de Software Libre están en alguno de los niveles de formación vinculados a la Universidad, ya sea primer, segundo o tercer ciclo. La edad media de un contribuidor se sitúa en 22.9 años, lo que coincide con la etapa final de formación de un estudiante de segundo ciclo universitario. Por otro lado, la edad a partir de la cual se comienza a realizar contribuciones a la comunidad del software libre se sitúa entre los 17 años y los 25 años.

El Concurso Universitario de Software Libre viene a cubrir este espectro de edades, con el objetivo de crear comunidad en torno al software libre dentro del ámbito universitario. El concurso se presenta no sólo como una oportunidad de tomar contacto con el software libre para aquellos que aún no se han iniciado, sino también para reforzar a aquellos estudiantes que ya están involucrados en algún tipo de desarrollo. Está iniciativa es sumadora a la apuesta de las administraciones públicas, y debe ser acompañadas de cambios pertinentes dentro de la universidad.

4.1 Estructura del Concurso

El concurso abarca tres fases que se detallan en los siguiente apartados:

Inscripción y Aceptación Durante la primera fase del concurso, los candidatos completará una solicitud detallando el proyecto de Software Libre al que quieren contribuir o que quieren comenzar. Para ello, se habilitará un sistema de registro web. Esta inscripción será evaluada por el comité del concurso que determinará si el proyecto puede formar parte de la competición. El proyecto quedará englobado en alguna de las categorías existentes definidas en las bases del concurso.

El proceso de inscripción estará abierto desde el 1 de Septiembre hasta el 13 de Octubre de 2006. La notificación de aceptación del proyecto tendrá lugar el 20 de Octubre de 2006. Tras el periodo de validación de la inscripción, se comunicará a los participantes la lista de proyectos aceptados.

Desarrollo del proyecto A partir de la fecha de aceptación, los participantes podrán comenzar con el desarrollo del proyecto. El desarrollo del software se canaliza fundamentalmente a través de dos medios:

- Una forja [3] es una aplicación colaborativa para la administración de proyectos de software, actuando como un repositorio de código fuente, además de proveer a los desarrolladores de distintas aplicaciones que faciliten el desarrollo y la administración de sus proyectos. Los proyectos del Concurso Universitario de Software libre están ubicando en la "Forja de Conocimiento

Libre de la Comunidad RedIRIS", la cual, tiene como objetivo principal fomentar los desarrollos de Software Libre en la comunidad RedIRIS así como servir de soporte a iniciativas de interés en el entorno académico-científico relacionadas con el "conocimiento libre".

- Un planet [13] es un sistema de sindicación de blogs, en el cual podemos encontrar las entradas más recientes de los blog relacionados. Los principales proyectos de Software Libre como Debian, Kde, Gnome.. los usan como punto de unión de la información publicada por los desarrolladores de dichos proyectos. Un blog o bitácores es un sitio web, periódicamente actualizado, que recopila cronológicamente textos o artículos de uno o varios autores, sobre un tema concreto. Todos los proyectos presentados al Concurso Universitario de Software Libre cuentan con un blogs donde quedan reflajados la evolución de sus proyectos, las dificultades que se les van presentando en su desarrollo y las decisiones tomadas durante la fase de desarrollo. Gracias a los comentarios que se pueden realizar en cada entrada publicada, cualquier persona puede entablar una conversación con los desarrolladores e incluso aconsejándole, proponiendo nuevas soluciones a problemas concretos o nuevas opciones para su proyecto, haciendo colaborativos los proyectos, llegando a crear comunidad en torno a ellos. En el planet del concurso encontramos recopiladas de las últimas entradas de los blogs de los proyectos participantes, facilitando el seguimiento de los distintos desarrollos.
- Lista de correo de participantes, una lista abierta en la que los participantes tienen completa visibilidad entre ellos y pueden responderse preguntas los unos a los otros sin necesidad de que la propia organización intervenga, desempeñando de esta forma la organización una tarea de mediación, coordinación y resolución de conflictos.

Entrega, evaluación y fase final Una vez completada la fase de desarrollo, se procederá a la entrega del proyecto junto con una pequeña documentación que será evaluada por el comité de evaluación. Los proyectos finalistas serán expuestos durante la fase final, en la que se procederá a realizar la entrega de premios.

La fase final consistirá en un ciclo de charlas y talleres: Un Ciclo de Conferencias, cuya temática tendrá un especial enfoque en el mundo del Software Libre. En esta sección podrán participar las empresas; y un Ciclo de Talleres, en los que los participantes expondrán los trabajos que han realizado a lo largo del concurso.

La fase final tendrá lugar durante los días 10 y 11 de mayo de 2007 en la Escuela Técnica Superior de Ingeniería Informática de la Universidad de Sevilla.

Bases del concurso

1. El concurso consiste en el desarrollo y presentación de un proyecto de Software Libre desarrollado íntegramente con una implementación libre de cualquier

lenguaje de programación. Los participantes y sus proyectos deberán cumplir los requisitos que se contemplan en los siguientes apartados.

2. El desarrollo del concurso consta de las siguientes fases:
 - Fase de inscripción: Durante este periodo, los participantes se inscribirán en el concurso mediante el formulario de la web.
 - Fase de aceptación: Se comunicará a todos los participantes la lista de proyectos aceptados.
 - Fase de desarrollo: Periodo destinado al desarrollo de los proyectos aceptados y finaliza con la entrega del proyecto.
 - Fase final: Consistirá en un ciclo de charlas y talleres donde serán expuestos los proyectos ganadores y se procederá a la entrega de premios.
3. Las fechas de las distintas fases en las que se compone el desarrollo del concurso está disponible en la sección de fechas importantes en la web del concurso. Todos los proyectos han de desarrollarse durante el periodo destinado para tal efecto, quedando fuera de concurso todo desarrollo realizado anteriormente al periodo estimado por la organización, a excepción de los proyectos basados en contribuciones de otros proyectos ya existentes, y en dicho caso, sólo se evaluará la aportación realizada al proyecto y nunca el conjunto del proyecto.
4. Los participantes al concurso han de ser mayores de edad y estar matriculados en primer o segundo ciclo alguna de las Universidades españolas durante el curso 2006/07 y estar en disposición de poder acreditarlo. Tras la notificación de aceptación, los participantes deberán acreditar la documentación correspondiente que certifique ser estudiante universitario de lo contrario el proyecto propuesto quedará fuera de concurso.
5. El máximo de participantes será de tres personas por proyecto.
6. Sólo se podrá participar en un proyecto a la vez.
7. Cada proyecto se incluirá en una de las siguientes categorías:
 - Ocio y Educación: Proyectos relacionados con la docencia, la educación y el entretenimiento. En esta zona caben todo tipo de proyectos con un marcado ámbito didáctico y de tiempo libre.
 - Web: Esta sección no incluye la realización de páginas web sino la implementación o mejoras de gestores de contenido existentes (CMS), interfaces web para dar un aspecto más amigable a utilidades existentes,...
 - Distribuciones: Proyectos relacionados con mejoras específicas en distribuciones de Linux.
 - Sistemas: Esta sección cubre todos los proyectos que no estén englobados en las categorías anteriores, tales como emuladores, drivers, seguridad informática, programas para entornos de escritorio (Gnome, KDE u otros)
 - ...
8. La organización pondrá a disposición de los participantes una forja en la que se alojarán los proyecto participantes en el concurso. Los participantes también dispondrán de un blog, en el que quedará reflejado la evolución del proyecto, las dificultades encontradas y las decisiones tomadas durante la fase de desarrollo. Tanto la forja como el blog serán empleados para evaluar la evolución de los proyectos.

9. La evaluación de los proyectos gira en torno a la pregunta "*¿Crea el proyecto comunidad?*", la cual se desglosa en los siguiente puntos:
 - Grado de documentación de la herramientas desarrolladas
 - Calidad del Desarrollo e Implementación y uso apropiado de la forja
 - Grado de finalización y proyección a medio y largo plazo
 - Índice de comunidad: grado de reusabilidad, estrategia de difusión empleada, relación con otras comunidades
10. El jurado estará compuesto por un comité de personal adjunto a la Universidad, Empresas, Instituciones y la Comunidad del Software Libre que evaluarán los proyectos en base a los puntos descritos anteriormente.
11. Durante la fase final se hará entrega de los premios para el ganador y finalista de cada categoría:
 - 1er premio: 1000 euros al proyecto.
 - Finalista: 400 euros al proyecto.

La organización se reserva el derecho de anular o declarar desiertos alguno de los premios en el caso de que los proyectos en alguna categoría no reúnan la calidad deseada o revocar cualquiera de los premios en caso de detectar alguna irregularidad.
12. La organización se reserva el derecho a tomar las medidas oportunas y/o a descalificar, previo aviso, aquellos participantes y/o proyectos que incumplan con algun punto de las bases durante el transcurso del concurso.
13. La organización se reserva el derecho a tomar las medidas oportunas y/o a descalificar, previo aviso, aquellos participantes y/o proyectos que incumplan con algun punto de las bases durante el transcurso del concurso.
14. El hecho de presentar la inscripción al concurso, implica la aceptación de estas bases en su totalidad.

5 Resultados Obtenidos

El número de participantes en el Concurso Universitario de Software Libre asciende a 135 participantes, con un total de 93 proyectos incluidos entre las distintas categorías del concurso. El mayor parte porcentaje de participantes los encontramos en la Universidad de Sevilla, lo que se justifica con el hecho de que la organización del concurso tiene base en esta ciudad (Fig. 1).

Por comunidades autónomas (Fig. 2) es Andalucía la que ocupa la primera posición en participantes, de nuevo mayormente debido a la alta participación de estudiantes de la Universidad de Sevilla, le siguen los núcleos universitarios más importantes del país tales como Madrid y la Comunidad Valenciana.

6 Conclusiones

El Concurso Universitario de Software Libre se trata de una actividad sumadora a la tendencia actual introducida por la administración y universidades cuyo objetivo principal es el de crear comunidad en torno al software libre en el ámbito académico. En la primera edición, las expectativas de participación estimadas

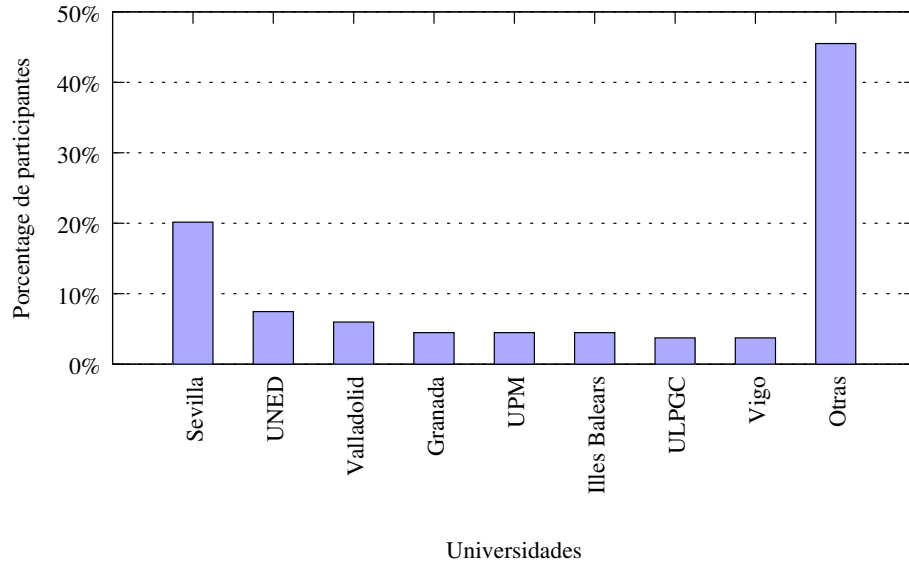


Fig. 1. Participación por Universidades

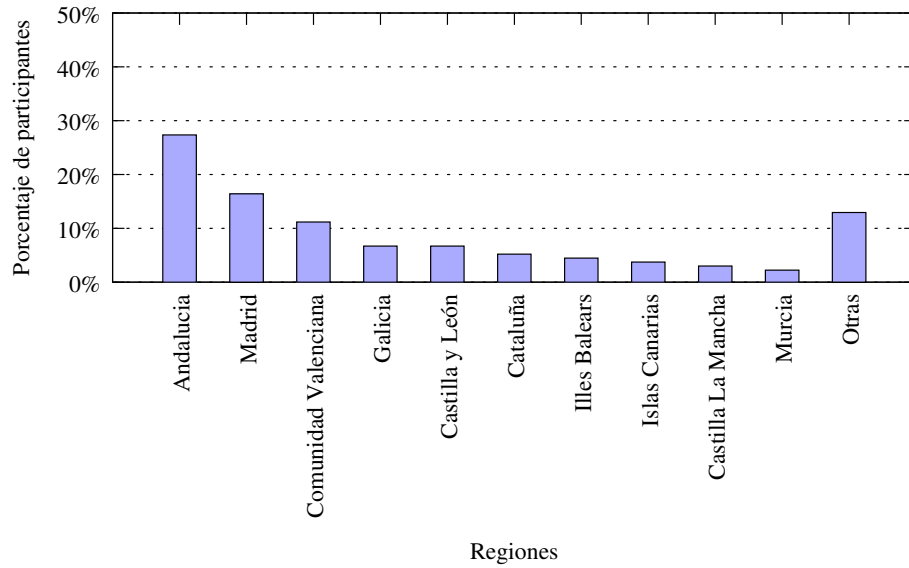


Fig. 2. Participación por Autonomías

por la organización han sido cubiertas con creces. Los pasos actuales del concurso van dirigidos a intentar consolidar la iniciativa a nivel nacional mediante la búsqueda del apoyo de otras universidades y organizaciones.

Agradecimientos

Agradecemos a las entidades públicas Cenatic, Junta de Extremadura y Junta de Andalucía el apoyo a ésta iniciativa, a las empresas Islanda y Yerbabuena por el esfuerzo realizado para participar apoyando el concurso, y a todos nuestros colaboradores por el apoyo y ayuda ofrecida y la confianza puesta en ésta iniciativa.

References

1. *Debian developers by country*. http://people.debian.org/~edward/globe/countries/pop-devel_a.html.
2. *Definición de software libre*. <http://www.gnu.org/philosophy/free-sw.es.html>.
3. *Forja de rediris*. <https://forja.rediris.es>.
4. *Free software world conference*. <http://www.freesoftwareworldconference.com/>.
5. *Gnome world wide: Developers world wide*. <http://live.gnome.org/GnomeWorldWide>.
6. *Guadalinux*. <http://www.guadalinux.org>.
7. *Guadalinux edu*. <http://www.juntadeandalucia.es/averroes/curso-guadalinux/>.
8. *Linux*. <http://www.gnulinex.net>.
9. *Linuxtag: Where .org meet .com*. <http://www.linuxtag.org/2007/en/home/aktuelles.html>.
10. *Oficina del software libre de la universidad de cádiz*. <http://www.softwarelibre.ulpgc.es>.
11. *Oficina del software libre de la universidad de las palmas de gran canarias*. <http://softwarelibre.uca.es>.
12. *Opensource initiative*. <https://www.opensource.org>.
13. *Planet concurso universitario de software libre*. <http://concurso-softwarelibre.us.es/planet/>.
14. Ghosh, Rishab Aiyer; Glott, Ruediger; Krieger, Bernhard y Robles, Gregorio. *Free/libre and open source software: Survey and study, part iv, survey of developers*. Informe técnico, Infonomics: Economics and Business of the Information Society.
15. Kegel, Dan. *How to get hired - what cs students should know*. <http://www.kegel.com/academy/getting-hired.html>.
16. Moody, Glyn. *A brief history of microsoft fud*. <http://laxer.com/module/newswire/view/57261/index.html>.
17. Paniagua, Javier. *España, Siglo XX: 1898-1931*.
18. Spolsky, Joel. *The guerrilla guide to interviewing*. <http://www.joelonsoftware.com/articles/fog0000000073.html>.
19. Stallman, Richard. *Por qué las escuelas deberían usar exclusivamente software libre*. <http://www.gnu.org/philosophy/schools.es.html>.
20. Sutor, Bob. *Microsoft press release about odf - its a start, but quit the fud*. <http://www.sutor.com/newsite/blog-open/?p=783>.

Cursos accesibles y reusables sobre la plataforma ALPE

Olga C. Santos¹,
 Jesús G. Boticario¹,
 Alejandro Rodríguez Ascaso¹
 Emmanuelle Gutiérrez y Restrepo²
 Carmen Barrera¹

¹ aDeNu Research Group. Computer Science School, UNED.
 C/Juan del Rosal, 16. Madrid 28040, Spain
 {ocsantos,jgb,arascaso}@dia.uned.es
 cbarrera@invi.uned.es
 http://adenu.ia.uned.es

² SIDAR Foundation-Universal Access.
 C/Sancho Dávila, 35, Madrid 28028, Spain
 emmanuelle@sidar.org
 http://www.sidar.org

Abstract. La plataforma de aprendizaje ALPE, desarrollada a partir del sistema de gestión del aprendizaje de código abierto dotLRN, facilita 1) la reutilización de contenidos y 2) la accesibilidad en el acceso a la información y la interacción con la interfaz. Más concretamente, ALPE consiste en un entorno de aprendizaje en línea accesible (de acuerdo con las pautas WAI del W3C) y multilingüe, basado en estándares educativos (SCORM, IMS-CP, IMS-QTI, IMS-MD, IMS-LD). Para facilitar la creación de los cursos de forma accesible, se utiliza una metodología durante el proceso de su elaboración, desde el diseño conceptual del curso hasta la importación del paquete con el diseño instruccional del curso en la plataforma.

Palabras clave: Estándares educativos, reutilización, SCORM, IMS QTI, IMS CP, IMS MD, plataformas educativas, código abierto, dotLRN, accesibilidad, WAI, metodología, desarrollo de cursos, diseño instruccional.

1. Introducción

La plataforma ALPE es un entorno de aprendizaje en línea accesible, multilingüe, basado en estándares educativos (SCORM¹, IMS-CP², IMS-QTI³, IMS-MD⁴, IMS-

¹ Sharable Content Object Reference Model: <http://www.adlnet.gov/scorm>

² IMS – Content Packaging: <http://www.imsglobal.org/content/packaging>

³ IMS – Question & Test Interoperability: <http://www.imsglobal.org/question>

⁴ IMS – Metadata: <http://www.imsglobal.org/metadata>

LD⁵) desarrollado sobre código abierto [1]. Más concretamente, la plataforma ALPE es un desarrollo hecho por el grupo aDeNu (Adaptación Dinámica de sistemas de Educación oN-line basada en el modelado del Usuario) de la Universidad Nacional de Educación a Distancia (UNED) sobre el sistema de gestión del aprendizaje dotLRN [2], con el objetivo de mejorar la reutilización de los contenidos y la accesibilidad en el acceso a la información y la interacción con la interfaz [3].

El soporte de estándares educativos por parte de herramientas de código libre (a veces llamadas FLOSS – Free/Libre Open Source Software) no es un hecho casual, sino que existen razones que lo justifican [4]. Por un lado, aunque el uso de código libre facilita la interoperabilidad entre diferentes sistemas, los contenidos de un sistema no siempre pueden ser portados fácilmente a otro. Sin embargo, la utilización de especificaciones abiertas para los contenidos permite su intercambio entre diferentes sistemas. En este sentido, el uso de especificaciones como SCORM e IMS es esencial para dar soporte a un aprendizaje en línea (eLearning) interoperable. Por otro lado, FLOSS puede dar soporte al aprendizaje constructivista, en donde se enfatiza la colaboración y la argumentación, los múltiples puntos de vista válidos y la idea de que las necesidades del estudiante han de ser cubiertas mientras va construyendo de su propio aprendizaje, teniendo en cuenta su entorno cultural. Además, las compañías de software necesitan que haya una masa crítica de usuarios (profesores y alumnos) que hagan uso de las especificaciones de eLearning para considerar invertir en el desarrollo de aplicaciones comerciales que lo soporten. En este sentido, en muchos casos (por ejemplo en el caso del IMS LD, dado lo extensa y compleja que es su especificación) hacen falta varias implementaciones de referencia que hayan sido generadas de forma colaborativa para aclarar los puntos conflictivos, y que además permitan el acceso a su código para que otros desarrolladores puedan implementar los mismos mecanismos. Si los desarrollos se hacen de forma independiente e interna, se tomarán decisiones diferentes a la hora de enfrentarse a los problemas de interpretación de las especificaciones que no harían posible la interoperabilidad buscada. El acceso al código desarrollado permite, además, que puedan enriquecerse desarrollos existentes con nueva funcionalidad, lo que agiliza la implementación de las especificaciones y facilita su adopción.

Actualmente, en un proyecto europeo eTEN del mismo nombre, ALPE, se va a validar el servicio ofrecido sobre esta plataforma, consistente en la entrega de cursos accesibles sobre habilidades básicas dirigidos fundamentalmente a usuarios con algún tipo de discapacidad visual o auditiva y a personas adultas en los mercados español, inglés y griego, que quieran mejorar sus competencias básicas con el fin de poder acceder a un empleo. Está previsto que la explotación del servicio pueda realizarse posteriormente en el resto de la Unión Europea. En este proyecto participan, además de la UNED y Soluziona en España, la Open University en el Reino Unido y PAB (“Panhellenic Association of the Blind”) en Grecia. El proceso de validación y los resultados de las primeras experiencias de evaluación con usuarios finales están recogidos en [5].

En este artículo se describe la importancia de garantizar la accesibilidad y se justifica el uso de los diferentes estándares educativos. Posteriormente, se detalla la

⁵ IMS – Learning Design:<http://www.imsglobal.org/learningdesign>

metodología definida para el desarrollo de los cursos accesibles siguiendo los estándares educativos descritos y se muestra como ejemplo el curso 'Recursos Web para enseñar a través de Internet', extraído de un curso impartido en la UNED dentro del programa de enseñanza abierta de esta universidad por parte de algunos de los miembros del grupo aDeNu. A continuación, se comentan los resultados de la primera experiencia con usuarios finales. Por último, se presentan algunas consideraciones y se detallan los trabajos futuros.

2. Cómo garantizar la accesibilidad en los contenidos

Si se tienen en cuenta ciertos criterios de accesibilidad a la hora de diseñar un curso el coste que supone su realización no es mucho mayor que el que supondría si no se tienen en cuenta dichos criterios, e infinitamente mucho menos si más adelante hubiera que mejorar el curso para hacerlo accesible.

La razón para abordar los requisitos de accesibilidad estriba en que en determinadas circunstancias o debido a la diversidad funcional de las personas, no se puede acceder a la información por alguno de los canales o formatos por los que se transmite. La forma de solucionarlo es especificar la información de manera que se pueda elegir el formato y canal por el que se quiere recibir la información. Por ejemplo, un gráfico que sintetice una idea no podrá ser recibido por una persona ciega, a menos que hayamos incluido una descripción textual de su contenido que explique la información que se ha querido transmitir en la imagen. Luego, la persona podrá elegir cómo quiere recibir dicha información, por ejemplo, a través de una línea Braille o mediante síntesis de voz.

La Iniciativa de Accesibilidad Web (WAI) ha elaborado unas Pautas de Acceso a los Contenidos Web (WCAG 1.0)⁶ y define unos puntos de chequeo que hay que validar para que los contenidos desarrollados sean accesibles. Además, para el caso particular de información científica con notación numérica, es necesario abordar la forma de especificarla, tanto internamente (e.g. MathML, TeX, LaTeX) como el formato de salida (e.g. Nemeth, Marburg, Math Braille). Trabajos en esta dirección están siendo realizados por el International Group for Universal Math Accessibility⁷ y los proyectos MathGenie⁸, Infty⁹, Lambda¹⁰, Bramanet¹¹ o HenterMath¹².

Si se cumplen estos puntos de chequeo se facilita el uso de las ayudas técnicas que necesitan los estudiantes con discapacidad, como líneas Braille, magnificadores de pantalla, pantallas táctiles de gráficos, etc. así como la integración de lenguaje de signos, subtítulos y soporte a símbolos. Además, también proporciona mejores contenidos para todos los estudiantes, ya que un buen diseño repercute en la calidad

⁶ <http://www.w3.org/TR/WCAG10-TECHS/>

⁷ <http://karshmer.lklnd.usf.edu/~igroupuma/index.html>

⁸ <http://karshmer.lklnd.usf.edu/~mathgenie/index.html>

⁹ <http://www.inftyproject.org/>

¹⁰ <http://www.lambdaproject.org/ASP/index.aspx?IDMenuAPP=0>

¹¹ <http://handy.univ-lyon1.fr/MH/bramanet/bramanet.php>

¹² <http://www.hentermath.com/>

de los mismos y permite su uso también en situaciones no ideales, como entornos de mucho ruido donde no se pueden oír las audiciones, navegadores ubicados en sitios públicos que no tienen instalados los plug-ins correspondientes o problemas con la descarga de imágenes u otro tipo de recursos multimedia por una baja velocidad en la línea de conexión.

Sin embargo, a la hora de preparar los contenidos, conviene tener en cuenta una serie de pautas que son de fácil aplicación para usuarios no técnicos y que combinan los criterios de las WCAG que hacen referencia a evaluaciones manuales con el trabajo realizado por Theofanos y Redish [6]. Estas pautas se han agrupado en tres aspectos diferentes [7]:

- **Estructurar los contenidos:** a) hacer la estructura clara y obvia, b) usar lenguaje claro y frases directas, c) dividir grandes bloques de información en párrafos con una idea clave cada uno, d) hacer un correcto uso de los encabezados para estructurar los contenidos, e) incluir la clave del contenido al principio de los encabezados, listas, párrafos, etc., f) usar listas de puntos para introducir afirmaciones, g) empezar los enlaces y encabezados con palabras relevantes y que sean diferentes de unos enlaces o encabezados a otros, h) no colocar contenido relevante en zonas bajas de la página, i) incluir enlaces de ancla para referirse a contenidos de la misma página.
- **Terminología:** a) prestar atención a cómo se pronuncian las palabras importantes por un lector de pantalla, y escribirlas consecuentemente (por ejemplo, en inglés conviene escribir 'home page' en vez de 'homepage' para que el lector de pantalla lo pronuncie correctamente), b) usar nombres estándares para servicios y secciones estándar, c) asegurarse de escribir explícitamente en el texto las palabras clave que los usuarios van a buscar en los contenidos, d) no crear imágenes para introducir texto, e) ser consistente con el vocabulario utilizado en el texto, en las descripciones textuales y en el contenido no textual, f) comprobar la correcta ortografía.
- **Información adicional sobre los contenidos:** a) incluir una descripción exacta que refleje el significado real de la imagen que se pretende describir, b) en el caso de las imágenes decorativas, usar el atributo "alt" nulo y no utilizar el atributo longdesc, c) escribir descripciones útiles para los enlaces, d) identificar cuándo se utiliza una abreviatura o un acrónimo, e) identificar los encabezados.

Estas pautas se centran en la accesibilidad de los contenidos, y deben ser tenidas en cuenta para generar contenido verdaderamente accesible. No obstante, garantizar únicamente estas pautas de accesibilidad no significa necesariamente que los contenidos sean accesibles para todos los usuarios. No es suficiente tratar de cumplir dichas pautas durante el desarrollo de los contenidos, sino que una vez desarrollados los cursos, hay que validar que efectivamente el contenido es accesible. Esta validación requiere dos pasos. Un primer paso que puede hacerse con herramientas automáticas, y un segundo paso que necesariamente ha de realizarse de forma manual.

Más concretamente, la metodología recomendada para hacer la validación de los contenidos es la siguiente:

1. Validación sintáctica del documento, incluyendo XHTML y CSS, mediante herramientas automáticas

2. Validación de la accesibilidad mediante herramientas automáticas como , TAW¹³ o WebXact¹⁴
3. Verificación manual de las pautas WAI no validadas de forma automática (puede hacerse con la ayuda de una herramienta como HERA¹⁵)
4. Verificación manual en navegadores en modo texto como Lynx
5. Verificación manual en navegadores gráficos como IExplorer, Firefox, Opera
6. Verificación de los colores de la página, con herramientas como GrayBit o Colour Contrast Analyser
7. Verificación empleando ayudas técnicas, como el lector JAWS o el magnificador Magic
8. Verificación del contenido, revisando i) ortografía y gramática, ii) claridad y simplicidad del lenguaje y iii) claridad en la organización y estructuración de los contenidos
9. Verificación de la navegabilidad y empleo de mecanismos de navegación de manera consistente

Por otro lado, existen diferentes tipos de herramientas que pueden utilizarse a la hora de desarrollar contenidos accesibles: editores, validadores de sintaxis XHTML, validadores de accesibilidad y reparadores de accesibilidad. Para reducir la carga de trabajo, como se dijo antes, lo mejor es tener en cuenta los criterios de accesibilidad desde el momento del diseño, y por tanto, utilizar un editor que ayude a garantizar la accesibilidad. Se ha dicho ‘ayudar a garantizar la accesibilidad’ porque es necesaria una componente humana. Por ejemplo, las herramientas pueden obligar a poner un texto descriptivo a la imagen que explique lo que se ve en ella. Pero sólo obligan a que dicha etiqueta existan y no entran en el contenido de la misma. Por ejemplo, si el texto dice ‘Esto es una imagen’ no está dando ninguna información por lo que no sería verdaderamente accesible. Sin embargo, para las herramientas de revisión automática sí lo sería. En este sentido, la WAI ofrece diferentes tipos de validadores¹⁶.

3. Uso de estándares educativos en ALPE

Los estándares favorecen la máxima integración entre distintas tecnologías, convirtiéndose en un componente facilitador de la accesibilidad y la reutilización. Además de los estándares propios de accesibilidad de la WAI (WCAG, ATAG, UAAG), es importante la consideración de estándares tecnológicos como XML, XHTML, CSS, XSL, SMIL, SVG, MathML, ECMAScript. En lo que se refiere al campo de los estándares en tecnologías de la educación su uso va a facilitar la reutilización de materiales, la creación de cursos más ricos y una formación más

¹³ Web Accessibility Test: <http://www.tawdis.net/taw3/cms/en>

¹⁴ WebXact: <http://webxact.watchfire.com/>

¹⁵ HERA es una herramienta diseñada para la revisión manual de la accesibilidad: <http://www.sidar.org/hera/>

¹⁶<http://www.w3.org/WAI/ER/tools/complete>

global, dado que los materiales creados en una herramienta pueden ser vistos por otra y exportados para una tercera.

Actualmente, los líderes en la industria y la investigación tanto de Europa como de América están trabajando juntos y colaborando en la continua evolución del estándar Sharable Content Object Reference Model (SCORM). En él se define un modelo de referencia común, que trata de satisfacer la accesibilidad, adaptabilidad, durabilidad, interoperabilidad y reutilización de los cursos y sus objetos educacionales. Éste es el estándar con mayor aceptación a la hora de estructurar los materiales. El empaquetado de todo el curso se realiza siguiendo el estándar IMS Content Packaging (IMS-CP), y el estándar por excelencia para caracterizar los Objetos Educacionales, es el IEEE Learning Object Meta-Data (IEEE LOM) que define y especifica un esquema de metadatos que permite múltiples implementaciones. IMS Learning Resource Meta-Data (IMS-LRM) referencia a este estándar en todos los aspectos teóricos del esquema, y además lo complementa con aspectos más prácticos, incluyendo las guías y herramientas necesarias para su uso. A través del estándar IMS Question and Test Interoperability (IMS-QTI) se describen objetos educacionales para la evaluación.

La plataforma ALPE facilita la reutilización de contenidos gracias al uso que hace de los estándares educativos, SCORM 1.2, IMS-CP 1.1.2, IMS-QTI 1.2.1, IMS-LRM 1.2.1. Aunque la plataforma dotLRN soporta también el estándar IMS Learning Design, inicialmente en ALPE no se ha considerado su utilización por diversas razones. Entre ellas, cabe mencionar la dificultad que supone para los autores la creación de cursos siguiendo este estándar, ya que aún no existe un soporte adecuado mediante el uso de plantillas. Esta carencia en el estado del arte actual de editores de IMS-LD ha sido detectado en diversas evaluaciones de proyectos centrados en el uso de dicho estándar, como por ejemplo, en el proyecto aLFanet [8].

Mediante el uso de IMS MD, los distintos materiales del curso vienen descritos por sus características básicas como formato, tamaño, nivel de interactividad, dificultad, tiempo previsto de aprendizaje, o idioma, lo cual facilita, la búsqueda y localización de recursos adecuados para componer un curso. A la hora de evaluar el nivel de conocimiento de los alumnos se suelen utilizar exámenes, cuestionarios o tests en forma de preguntas. Para su correcta interpretación se ha definido el estándar IMS QTI, en el que además de describir las preguntas y las múltiples posibles respuestas, se indica la forma de evaluar cada una de ellas.

En este escenario, ALPE recibe un curso, en formato SCORM, compuesto por recursos o materiales caracterizados según IMS MD. Algunos de estos materiales serán de evaluación, acordes con el estándar IMS QTI. Durante la interacción con el alumno, ALPE le ofrece la ruta de aprendizaje más adecuada a sus necesidades, basándose en SCORM, selecciona entre los materiales los diseñados para ese tipo de audiencia, teniendo en cuenta los metadatos de IMS MD (como ejemplo ilustrativo podemos tener distintos materiales referidos a los mismos conceptos, pero están diseñados para usuarios con diversas necesidades, con un video, una animación con lenguaje de signos, un audio o una imagen) y realizará un diagnóstico del conocimiento del alumno según los resultados de los test de conocimiento en formato IMS QTI.

Entrando en detalles más técnicos, SCORM está dividido en una colección de “libros técnicos”, agrupados en tres temas principales: El “Modelo de Agregación de Contenidos (CAM)” que especifica cómo describir, empaquetar y definir componentes usados en una experiencia de aprendizaje; el “Entorno de Ejecución (RTE)” el cual describe el Sistema de Gestión de Aprendizaje (LMS), los requisitos para gestionar el entorno de ejecución en términos del protocolo de comunicación entre el LMS y los Objetos de Contenido Compartibles (SCO), y los elementos del modelo de datos usado para pasar información relevante de la experiencia del estudiante con el contenido; y la “Secuenciación y Navegación (SN)” que describe cómo el contenido conforme a SCORM podría ser secuenciado a través de un conjunto de eventos de navegación iniciados por el estudiante o iniciados por el sistema.

4. Metodología para el desarrollo de cursos accesibles

Para facilitar la creación de los cursos de forma accesible y poder utilizarlos en el proyecto ALPE [5], el grupo aDeNu ha trabajado en la metodología a seguir para su elaboración, desde el diseño conceptual del curso hasta la importación del paquete SCORM en la plataforma.

Como se introdujo en [5], la situación que se ha encontrado es que la mayoría de los profesores tenían sus contenidos preparados en Word, y a partir de ahí, había que preparar el curso para hacerlo accesible y empaquetarlo en SCORM. Como se vió en el apartado anterior, lo más recomendable es abordar los requisitos de accesibilidad desde el principio. Sin embargo, dada las circunstancias, inicialmente se buscó una solución que permitiera la conversión del material existente en unos contenidos accesibles que posteriormente pudieran empaquetarse en SCORM. Para ello, se propuso el uso de una herramienta comercial, identificada en un análisis del estado del arte de editores accesibles que generan contenido en SCORM [3] llamada Course Genie, que consiste en un plug-in para Word que permite validar de forma automática algunos criterios de accesibilidad (principalmente imágenes y tablas) y posteriormente convertir dicho contenido en páginas HTML 4.1 y empaquetarlo en SCORM. Sin embargo, los autores de este trabajo seguimos trabajando en generalizar dicha metodología, haciendo uso de herramientas abiertas para tener en cuenta la amplia gama de opciones existentes.

Para que un curso sea accesible, los contenidos tienen que incluir las descripciones adecuadas para que se le pueda ofrecer al estudiante la información en el formato adecuado, como se vio en el apartado 2. Por su parte, para realizar un curso basado en estándares, no basta con preparar los contenidos, sino que es necesario definir el diseño instruccional asociado, es decir, hay que definir la forma en que los estudiantes deben trabajar dichos contenidos definiendo las actividades que deben realizar sobre los mismos, su planificación temporal, las autoevaluaciones y exámenes para evaluar el aprendizaje de los contenidos, etc. Por las razones expuestas en el apartado anterior (i.e. hoy por hoy, el estándar más implementado en las plataformas de aprendizaje es SCORM, basado en IMS-CP) la metodología se

basa en el uso de dicho estandar, complementado con IMS-QTI para elaborar las autoevaluaciones y los exámenes e IMS-MD para caracterizar el curso y los materiales.

De acuerdo con la metodología definida, para realizar un curso basado en estándares y accesible, se deben seguir los siguientes pasos:

1. **Diseñar el curso conceptualmente.** A nivel conceptual, hay que definir los contenidos, las actividades y los cuestionarios de evaluación del curso y cuál va a ser la forma de ofrecerlos a los estudiantes para que los trabajen. Es conveniente también definir la planificación temporal del mismo.
2. **Crear los materiales.** Este paso implica crear tanto los contenidos como las actividades y los cuestionarios (estos últimos en IMS-QTI) teniendo en cuenta los criterios definidos en el apartado 2. para que sean accesibles. Para garantizar su accesibilidad, se recomienda que los materiales generen con cualquier editor cuya salida sea HTML 4.01 estricto y luego aplicar una hoja de estilos que tenga en cuenta criterios de accesibilidad y recomendaciones de uso que pueden concretarse en una plantilla para los editores de texto más comunes.
3. **Validar la accesibilidad de los materiales.** Es conveniente hacer una validación según la metodología definida en el apartado 2. para asegurar la accesibilidad del material diseñado. Cuanto más estricto se haya sido siguiendo las pautas de accesibilidad, menos esfuerzo habrá que dedicar a la validación. Previamente a la validación se pueden pasar herramientas de limpieza de código, como HTML Tidy.
4. **Definir el diseño instruccional.** Ello implica jerarquizar los contenidos del curso e indicar el momento en que debe utilizarse cada material, creando para ello el árbol de contenidos en SCORM. Aunque existen diversos editores, el más utilizado es Reload, implementado de forma abierta.
5. **Añadir metadatos a los materiales.** Es conveniente añadir metadatos (siguiendo la especificación de IMS-MD) a los contenidos, es decir, especificar la naturaleza de los mismos para que se pueda hacer un uso más apropiado dentro del curso, indicando, por ejemplo, su nivel de dificultad, su nivel de interactividad, el tiempo estimado que requiere trabajar con él, etc. Estos metadatos se pueden añadir con el editor SCORM del curso (e.g. con Reload) o a posteriori en la propia herramienta donde se importa el paquete SCORM que se genera en el siguiente paso. Por otro lado, de cara a facilitar la adaptación de los materiales del curso a las necesidades (tanto de aprendizaje como de interacción) de los estudiantes conviene ampliar dichos metadatos con el trabajo que está realizando el grupo Dublin Core – Accessibility¹⁷.
6. **Empaquetar el curso.** El resultado de las tareas anteriores se debe exportar en un fichero comprimido (.zip) que contendrá los materiales, las evaluaciones, los metadatos y las indicaciones del diseño instruccional siguiendo el estándar SCORM junto con IMS-CP.

¹⁷ <http://dublincore.org/groups/access/>

7. **Importar el curso en la plataforma.** El último paso consiste en publicar el curso a una plataforma que soporte los estándares SCORM e IMS (CP, QTI y MD) para que los estudiantes puedan acceder a los contenidos siguiendo el diseño instruccional definido anteriormente.

5. Curso 'Recursos Web para enseñar a través de Internet'

Para poder llevar a cabo la evaluación con usuario finales, se preparó un curso siguiendo la metodología definida en el apartado anterior. Los contenidos de dicho curso se extrajeron de un curso impartido en los programas de enseñanza abierta de la UNED por parte de algunos de los miembros del grupo aDeNu desde el año 1999 denominado 'Aprender a Formar a través de Internet'¹⁸. Se seleccionó el capítulo sobre 'Recursos web para enseñar a través de Internet'.

Dicho curso fue enriquecido por la Fundación SIDAR con una hoja de estilos con requisitos de accesibilidad, imágenes con su texto alternativo y descripción correspondiente, valores a los title de los enlaces que estaban nulos, vídeo con subtítulos en SMIL y vídeo en streaming (con sus correspondientes subtítulos incrustados en la pista visual) además de una revisión del marcado de las listas de definición, acrónimos, abreviaturas, palabras en otro idioma, etc.

Además, se elaboraron cuestionarios en IMS QTI, tanto en la versión Lite como en la completa (concretamente en la versión 1.2.1), para ver las preferencias por parte de los usuarios. El curso además, hacía referencia a algunos servicios de colaboración, como el Foro y el Área de Almacenamiento. A continuación, se muestra el resultado del mismo a través de las correspondientes capturas de pantalla.

2.1 Ejemplo del curso

En este apartado se muestran algunas capturas de pantalla del curso, tanto desde el punto de vista del estudiante como del profesor. Por un lado, el estudiante tiene acceso a los contenidos a través del curso empaquetado en SCORM, que incluye además evaluaciones en IMS QTI y acceso a los servicios del foro y el área de almacenamiento. Por su parte, el profesor, además de acceder al curso y a los servicios, puede configurarlos convenientemente, acceder a los metadatos del curso y a las trazas de ejecución de los estudiantes.

Vista del alumno

¹⁸ <http://www.ia.uned.es/~jgb/docencia/apr-for>

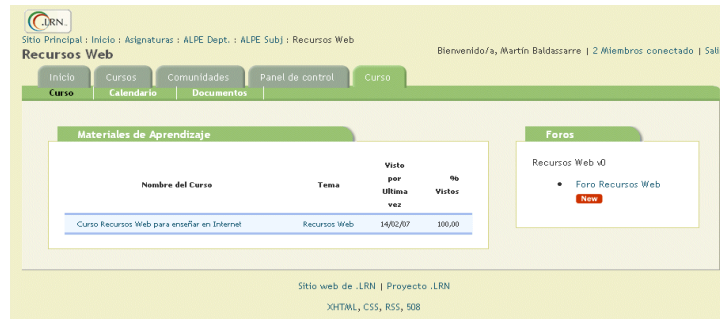


Figura 1. Entrada al espacio del curso, donde se encuentran los materiales de aprendizaje (curso SCORM) y un foro de apoyo al curso, además del acceso al calendario y al área de almacenamiento. El estudiante tiene acceso además a otros cursos y comunidades en donde esté matriculado, y a su panel de control para configurar sus preferencias.

The screenshot shows a course page with a tree view on the left and a slide titled 'Recursos para enseñar a través de Internet'. The tree view includes: Course Index, Recursos web para enseñar a través de Internet, Descripción, Contenidos del curso (1. Introducción a los Recursos web, 2. Recursos web tradicionales, 3. Motores de búsqueda, 4. Recursos de la Web), Recursos Adicionales, Gráficos del curso, Evaluación, and Servicios. The slide content is as follows:

1. Introducción a los recursos web

Internet ofrece diferentes tipos de servicios que pueden ser usados en diversos campos de aplicación. Todos los servicios se apoyan en un **protocolo**, que no es más que un conjunto de reglas que indican cómo se comunican unos ordenadores con otros. Estos protocolos se basan en el **modelo cliente/servidor**, que diferencia a los ordenadores en **servidores** o **clientes** en función de cómo sea el flujo de peticiones entre ellos. Así, los ordenadores que están conectados a la red continuamente esperando a recibir peticiones de servicio por parte de otros ordenadores se denominan **servidores** y comparten sus recursos con el resto de usuarios. Los ordenadores que realizan las peticiones de servicio se denominan **clientes**.

The slide also features a diagram of the client-server model. It shows a central cloud labeled 'RED' (Network). To the left, there are two boxes labeled 'Cliente' (Client). To the right, there is one box labeled 'Servidor' (Server). Blue arrows indicate bidirectional communication between the clients and the network, and between the network and the server.

Figura 2. Contenido del curso propiamente dicho, siguiendo el diseño instruccional definido por SCORM, que se muestra en el árbol de la izquierda. A la derecha se muestran los contenidos de cada una de las secciones del curso.

The screenshot shows a course page with a tree view on the left and a slide titled 'Recursos para enseñar a través de Internet'. The tree view includes: Course Index, Recursos web para enseñar a través de Internet, Descripción, Contenidos del curso (1. Introducción a los Recursos web, 2. Recursos web tradicionales, 3. Motores de búsqueda, 4. Recursos de la Web), Recursos Adicionales, Gráficos del curso, Evaluación, and Servicios. The slide content is as follows:

Evaluación

En este curso se ofrecen 2 cuestionarios, en diferente formato:

Cuestionario QTI Lite: [Cuestionario realizado en IMS QTI Lite \(requiere Javascript\)](#)

Cuestionario QTI 1.2.1: [Cuestionario realizado en IMS QTI 1.2.1](#)

Curso para ALPE

Figura 3. Acceso a la evaluación del curso, que se ofrece tanto en IMS QTI Lite como en IMS QTI 1.2.1.

The screenshot shows a course page with a tree view on the left and a slide titled 'Recursos para enseñar a través de Internet'. The tree view includes: Course Index, Recursos web para enseñar a través de Internet, Descripción, Contenidos del curso (1. Introducción a los Recursos web, 2. Recursos web tradicionales, 3. Motores de búsqueda, 4. Recursos de la Web), Recursos Adicionales, Gráficos del curso, Evaluación, and Servicios. The slide content is as follows:

Cuestionario

Por favor, responde a las siguientes preguntas.

1. La Web 2.0 no tiene en cuenta tema de privacidad de los datos, ya que el valor está en los datos y no en el software que lo utiliza.

Responde por favor si es verdadero o falso

a) True

b) False

Check your answer

Figura 4. Evaluación del curso implementada en IMS QTI Lite.

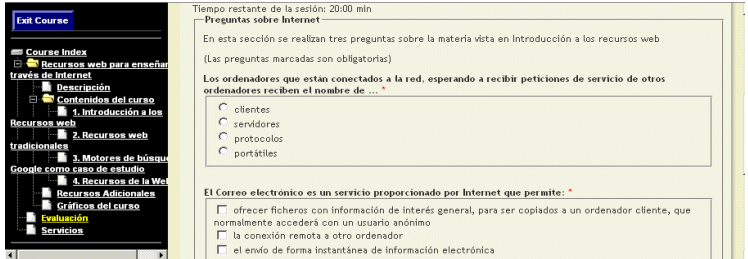


Figura 5. Evaluación del curso implementada en IMS QTI 1.2.1.

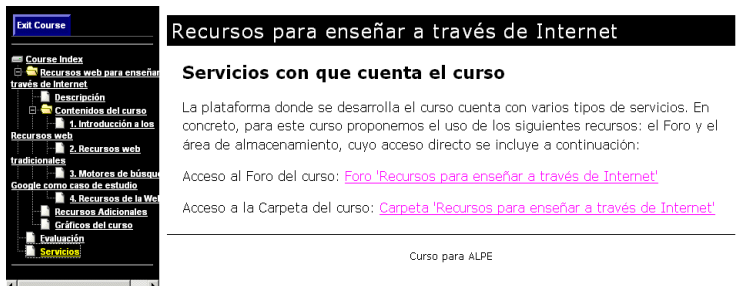


Figura 6. Acceso a los servicios ofrecidos en el curso, un foro y el área de almacenamiento.

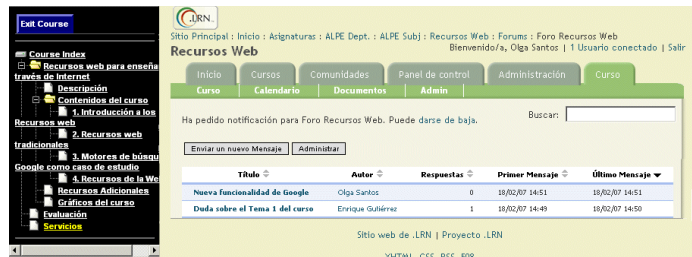


Figura 7. Vista del foro ofrecido para realizar el trabajo colaborativo del curso e intercambiar dudas con el profesorado.




Figura 8. Vista del área de almacenamiento donde el profesor puede incluir información adicional a la contenida en el curso SCORM, y donde los

estudiantes también pueden colgar los trabajos que vayan realizando e intercambiar referencias.

Vista del profesor

Además de la funcionalidad anterior, el profesor también tiene acceso a la configuración del curso y de los servicios asociados, y al seguimiento de la interacción del alumno en el curso.



Curso

Recursos Web para enseñar en Internet

Cursos Disponibles	Metadatos?	SCORM?	Estado	Rastreado?	¿Entrega por defecto?	Sesión SCORM	Dueño	Fecha de Creación	Exportar	Administrar Curso
Curso Recursos Web para enseñar en Internet	Yes	No	Enabled	Yes	delivery	Click here	Olga Santos	14/02/07 18:10	[zip]	[Admin]
Curso Demo	Yes	No	Parabola	No	delivery	Click	Olga	14/02/07	[zip]	[Admin]

Figura 9. Acceso a la administración del curso, donde se pueden ver los metadatos del curso, las interacciones de los alumnos y cambiar algunos parámetros de configuración del mismo.



Curso

MD: Ciclo de Vida MD, Meta MD, MD Técnica, MD Educativo, Derechos MD, Realización MD, Anotación MD, Clasificación MD

Tipo de Objeto: Versión y Esquema de MD

ims_manifest_object

Metadatos General

Títulos: en Curso Demo v1.0

Entradas de Catálogo: Ninguna Entrada de Catálogo Disponible

Figura 10. Metadatos del curso especificados en IMS Metadata.

Título	Visto Por	Total de Vistas	Visto por Usuario vez
Contenido del curso	Olga Santos	2	14/02/07 09:27
Descripción	Padre Herrera	3	14/02/07 09:06
2. Recursos web tradicionales	Padre Herrera	4	14/02/07 09:02
3. Misiones de búsqueda. Google como caso de estudio	Padre Herrera	4	14/02/07 09:02
4. Recursos de la Web 2.0	Padre Herrera	4	14/02/07 09:04
2. Recursos web tradicionales	Emanuela Quiñero	2	14/02/07 09:04
3. Misiones de búsqueda. Google como caso de estudio	Emanuela Quiñero	3	14/02/07 09:04
4. Recursos de la Web 2.0	Emanuela Quiñero	3	14/02/07 09:05
Recursos Adicionales	Emanuela Quiñero	1	14/02/07 09:05
Objetos del curso	Emanuela Quiñero	1	14/02/07 09:05
Recursos web para enseñar a través de Internet	Padre Herrera	6	15/02/07 09:02
Contenido del curso	Padre Herrera	6	15/02/07 09:02
Recursos Adicionales	Padre Herrera	6	15/02/07 09:05
Objetos del curso	Padre Herrera	4	15/02/07 09:07
Descripción	Padre Herrera	6	15/02/07 09:08
Recursos web para enseñar a través de Internet	Elena del Campo	3	17/02/07 10:02
1. Introducción a los Recursos web	Elena del Campo	2	17/02/07 10:29
Recursos web para enseñar a través de Internet	Marta Balasara	3	14/02/07 10:29
Contenido del curso	Marta Balasara	1	14/02/07 10:27
Recursos Adicionales	Marta Balasara	1	14/02/07 10:28
Objetos del curso	Marta Balasara	1	14/02/07 10:28
1. Introducción a los Recursos web	Marta Balasara	1	14/02/07 10:46
2. Recursos web tradicionales	Marta Balasara	1	14/02/07 10:46
3. Misiones de búsqueda. Google como caso de estudio	Marta Balasara	1	14/02/07 10:54
4. Recursos de la Web 2.0	Marta Balasara	1	14/02/07 10:56
Resumen	Marta Balasara	1	14/02/07 10:56
2. Recursos web tradicionales	Elena del Campo	2	17/02/07 10:24
3. Misiones de búsqueda. Google como caso de estudio	Elena del Campo	2	17/02/07 10:24
Descripción	Emanuela Quiñero	2	15/02/07 11:06
Recursos web para enseñar a través de Internet	Emanuela Quiñero	3	15/02/07 11:46
1. Introducción a los Recursos web	Emanuela Quiñero	3	15/02/07 11:46
Recursos web para enseñar a través de Internet	Enrique Quiñero	1	15/02/07 12:08
Descripción	Enrique Quiñero	2	15/02/07 12:18
Contenido del curso	Enrique Quiñero	2	15/02/07 12:11
Resumen	Padre Herrera	4	14/02/07 09:06

Figura 11. Seguimiento de las interacciones de los alumnos en el curso.

6. Experiencia con usuarios finales

En [5] se detalla la primera experiencia realizada con usuarios finales utilizando un curso en SCORM desarrollado siguiendo la metodología definida anteriormente e importado en la plataforma ALPE. Como se ha comentado en el apartado anterior, de forma previa a la evaluación con usuarios se hizo una revisión del contenido generado por las herramientas que se están utilizando como mediadoras entre el original creado por el profesor y su conversión para ser introducido en la plataforma, detectando algunos fallos de accesibilidad en dicho contenido generado. Por ello, se procedió a “limpiar” los contenidos generados, siendo conscientes de que hará falta trabajar sobre los sistemas mediadores para garantizar la accesibilidad de los contenidos.

El objetivo de esta evaluación con usuarios finales es detectar los posibles fallos de accesibilidad y adaptabilidad que pudiera tener aún la plataforma y los contenidos generados para ella, así como el grado de satisfacción del usuario. La evaluación de usuario final se hizo con el concurso de usuarios con deficiencia visual, deficiencia auditiva y adultos, de acuerdo con los objetivos del proyecto ALPE. Entre los participantes había usuarios de ayudas técnicas y de diversos navegadores, así como usuarios con diverso nivel de acercamiento al uso de Internet y de plataformas de e-learning. Es decir, había usuarios que nunca habían tenido acceso anteriormente a la formación en línea y otros que tenían ya experiencia con distintas plataformas. El curso tenía contenidos y características controlados, de manera que pudieran probarse la recepción e interacción por parte de los usuarios, y unos objetivos pedagógicos claramente perfilados para detectar qué necesidades de usuario quedaban cubiertas y cuáles no.

Para la evaluación se preparó un cuestionario con una primera parte en la que se recogían los datos personales de los alumnos, su situación relativa al grado de discapacidad visual, auditiva o de otro tipo, si utiliza o no ayudas técnicas, su experiencia en el uso de Internet, su experiencia en el uso de plataformas de e-learning, y su disposición hacia el uso de estas plataformas. El cuestionario contaba con una segunda parte en la que se pretendía detectar si los usuarios habían encontrado algún fallo de accesibilidad debido a la plataforma, a los contenidos en sí o a su poca experiencia en el uso de Internet o de su propio navegador y/o ayuda técnica. Para ello, se incluyeron en el cuestionario las preguntas sobre los doce fallos de accesibilidad más comunes, tal como indican los recientes estudios encargados por la Disability Right Comission¹⁹ del Reino Unido y la Organización de Naciones Unidas²⁰, pero redactadas en un lenguaje claro para los usuarios y poniendo de relieve su percepción sobre los contenidos y no tanto los aspectos técnicos de estos. También se incluyeron aquí cuestiones sobre los objetivos pedagógicos del curso y sobre la satisfacción o insatisfacción en el uso de la plataforma.

En los resultados de este primer análisis no ha habido ninguna sorpresa en cuanto a los requisitos de accesibilidad o la percepción de los usuarios sobre su carencia, cuando la había. Se detectó que los usuarios adultos y con poca experiencia en el uso de Internet tienen un gran desconocimiento de las opciones que les ofrece su navegador. Por tanto, puesto que puede darse el caso de que un usuario de la plataforma acceda a un curso dado con ese desconocimiento, habría que considerar la necesidad de ofrecer una formación previa en el uso de Internet y de los agentes de usuario o bien la de ofrecer una información básica sobre las opciones de accesibilidad que todos tienen a su alcance cuando los utilizan. Esta primera evaluación han permitido corroborar los objetivos del proyecto ALPE como una necesidad y las tareas que se están llevando a cabo como de suma importancia. En la evolución del proyecto será necesario, por tanto, arbitrar las distintas opciones de adaptabilidad previstas, y generar guías para los usuarios finales de la plataforma en sus dos vertientes: profesores y estudiantes.

Como conclusión general de la evaluación, todos los estudiantes coinciden en su satisfacción sobre el nivel de aprendizaje alcanzado y su valoración general de la experiencia ha sido positiva.

7. Consideraciones

El proyecto ALPE se encuentra en su primera fase de desarrollo, y por tanto, está trabajando en la preparación de los cursos y de la plataforma para comenzar la validación del mercado prevista a partir de abril de 2007. Para ello, se están adecuando cursos existentes sobre habilidades básicas para que cumplan los requisitos de accesibilidad y reutilización establecidos siguiendo la metodología

¹⁹ <http://www.drc-gb.org/PDF/2.pdf>

²⁰ <http://www.nomensa.com/resources/research/united-nations-global-audit-of-accessibility.html>

expuesta en el apartado 2. Por otro lado, se está mejorando el soporte ofrecido por la plataforma en ambos niveles, accesibilidad y estándares educativos. Con respecto a la accesibilidad, dentro del proyecto Zen²¹ de dotLRN se está trabajando para proporcionar una nueva interfaz más accesible que se construya a partir de código de widget correctamente formado. Además, se están teniendo en cuenta evaluaciones externas [9] sobre la accesibilidad de los paquetes educativos (por el momento LORS y Assessment, aunque la referencia anterior también incluye el paquete IMS-LD) para garantizar que la salida de los mismos ofrezca un nivel aceptable que permita el acceso por parte de usuarios con algún tipo de discapacidad. No obstante, a pesar de las carencias detectadas en dicha evaluación externa, las primeras evaluaciones con usuarios finales han sido positivas, como se puede ver en el apartado 6. Por desgracia, los usuarios con discapacidad están demasiado acostumbrados a enfrentarse a sitios no accesibles, y han desarrollado sus propias estrategias de navegación para soslayar las barreras que encuentran al acceder a la información, cuando ello es posible. Es cierto que hay bastantes aspectos que mejorar y que serán bienvenidos por los usuarios. Pero no es menos cierto que se cuenta con los recursos necesarios para llevarlo a cabo en los plazos establecidos dentro del proyecto ALPE.

8. Trabajos futuros

El grupo aDeNu, en colaboración directa con la comunidad de código abierto de OpenACS/dotLRN está actualmente trabajando en el proyectos Zen y en la mejora de los paquetes LORS y Assessment de dotLRN para garantizar que en el momento de comenzar la validación del mercado de la plataforma dentro del proyecto ALPE (abril 2007), el servicio ofrecido a los usuarios es verdaderamente accesible y ofrece cursos reutilizables (empaquetamiento de materiales en IMS-CP, diseño instruccional en SCORM, caracterización de los contenidos en IMS-MD y evaluación del aprendizaje mediante cuestionarios en IMS-QTI) sobre habilidades básicas que van a ayudar a usuarios con algún tipo de discapacidad a mejorar su nivel de empleo.

Por otro lado, se va a ampliar este curso sobre recursos web sobre cómo aplicar los recursos existentes en el desarrollo de contenidos accesibles para la web.

Reconocimientos

La validación del mercado de la plataforma ALPE está parcialmente financiado por el programa eTEN de la Comisión Europea (eTEN 029328). Las primeras evaluaciones con usuarios finales de la plataforma ALPE han sido coordinados por la Fundación SIDAR, y quisiéramos agradecer especialmente la participación de Martín Baldassare, Pedro Novoa y Enrique Gutiérrez.

²¹ <http://openacs.org/xowiki/dotlrn-zen-project>

Referencias Bibliográficas

1. Santos, O.C. and Boticario, J.G.: Building virtual (learning) communities to support people with special needs upon ALPE platform. Proceedings of the IADIS International Conference on Web Based Communities 2006, p. 312-316, (2006)
2. Santos, O.C., Boticario, J.G., Raffenne, E., Pastor, R.: Why using dotLRN? UNED use cases. 1st International Conference on FLOSS: Free/Libre/Open Source Systems. (2007, in press)
3. Santos, O.C.: Technology Enhanced Life Long eLearning for All. In K. Maillet and R. Klamma: Proceedings for the 1st Doctoral Consortium on Technology Enhanced Learning. European Conference on Technology Enhanced Learning, p.66-71, (2006)
4. Griffiths, D., Blat, J.: Open Source and IMS Learning Design: Building the Infrastructure for eLearning. In Proceedings of the First International Conference on Open Source Systems, Genova, pp. 329-333, (2005).
5. Santos, O.C., Boticario, J.G., Fernández del Viso, A., Pérez de la Cámara, P., Rebate Sánchez, C. and Emmanuelle Gutiérrez y Restrepo. Basic skills training to disabled and adult learners through an accessible e-Learning platform. 12th International Conference on Human-Computer Interaction (HCI 2007), July 2007 (in press).
6. Theofanos, M.F. and Redish, J.: Bridging the gap: between accessibility and usability'. Interactions, Volume X, Issue 6, November-December 2003, pages 38-51, (2003).
7. Santos, O.C., Boticario, J.G.: Requirements for Building accessible web-based communities for people with functional diversity. In International Journal of Web Based Communities (in press, 2007).
8. Santos, O.C. and Boticario, J.G.: Meaningful pedagogy via covering the entire life cycle of adaptive eLearning in terms of a pervasive use of educational standards: the aLFanet experience. In the First European Conference on Technology Enhanced Learning (EC-TEL'06), p. 691-696, (2006).
9. Revilla Muñoz, O.: Accessibility Requirements for the educational packages in dotLRN. 1st International Conference on FLOSS: Free/Libre/Open Source Systems. (2007, in press)

Sistema de Colas Condor

Ana Silva Gallego¹

¹ Centro Informático Científico de Andalucía
asilva@cica.es

Abstract. En este artículo expongo el sistema de Colas que hemos configurado en el cluster que se ofrecerá como recurso compartido por los investigadores andaluces, en el Centro Informático Científico de Andalucía.

Keywords: Sistema de colas Condor, Cluster.

1. Introducción

Para comenzar nos situaremos en el concepto “Cluster”, ¿Qué es un cluster? podríamos definirlo como un conjunto de ordenadores que trabajan de forma conjunta, que permite la distribución de una gran carga de trabajo entre ellos. ¿Por qué usamos cluster? Por su bajo coste frente a una sola máquina con gran potencia de cálculo, es decir con un gran número de procesadores.

Hay distintos tipos de clusters, tales como de alta disponibilidad, de reparto de carga y de alto rendimiento. Este último es el que se usa en sistemas de cálculo, como es el que exponemos a continuación.

El cluster con el que cuenta el CICA, es un cluster compuesto a nivel hardware por las siguiente máquinas:

Ocho servidores con 2 CPUs Xeon dual core y 3GB de RAM cada uno (total: 32 cores), que será ampliado por 110 servidores con CPU Core 2 DUO y 4 GB por nodo.

A nivel de Software, se está equipando con software de cálculo y aplicaciones necesarias para lanzar trabajos de investigación.

Contamos con Compiladores Intel (Fortran OpenMP y C++), bibliotecas matemáticas tales como Lapack, Blas e Intel MKL, y de paso de mensajes como son las OpenMPI 1.1.4.

Contamos con un sistema de colas Condor y herramientas de gestión como C3. Monitorizamos recursos con Ganglia.

Comenzaremos ahora con una breve introducción al Sistema colas CONDOR.

1.1. ¿Qué es Condor?

A nivel general podemos definirlo como un sistema de gestión de carga para tareas de computación intensivas.

Proporciona un sistema de colas ante tareas enviadas por los usuarios, políticas de planificación de ejecución, esquema de prioridades, monitorización y gestión de recursos.

1.2. ¿Para qué sirve Condor?

Es un sistema que nos permite abordar tareas de cálculo que sobrepasan a la capacidad de cálculo de una máquina individual, en resumen, nos permite aprovechar al máximo los recursos de un cluster.

Aplicaciones actuales de un sistema de colas:

- simulación de resistencia de materiales.
- rendering de imágenes, creación de animaciones.
- Etc...

Actualmente en el cluster que cuenta el cica, se ejecutan aplicaciones paralelizadas en Fortran y C++, que ejecutadas en un ordenador simple tardarían meses.

Programas de Investigadores de la Escuela de Ingenieros, que necesitan procesar repetidas veces el mismo programa con distintas entradas de datos.

2. Condor

Pasos previos a su instalación en el cluster de CICA.

Descargamos la última versión estable de Condor : `condor-6.8.2-linux-x86-rhel3-dynamic.tar.gz`.

Instalamos librerías : `compat-libstdc++-33.x86_64` y para 32 bits `compat-libstdc++-33.i386`.

Realizamos configuración de los nodos de cálculo, que serán dentro del sistema de colas como nodo execute y una máquina externa al cluster que se encargará de la función de Central – Manager y Submit.

2.1. Definiciones previas

Pool: colección de procesadores que usa Condor.

2.2. Tipos de máquinas Condor

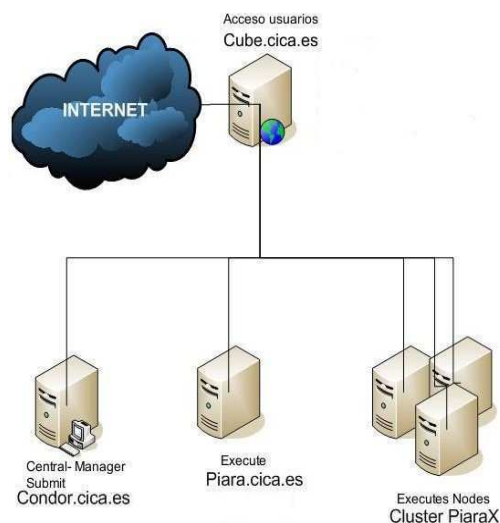
Un sistema colas, cuenta con 3 tipos de máquinas, combinables entre sí con algunas restricciones.

Tipos:

Submit: máquinas desde donde los usuarios lanzan sus tareas.

Execute: máquinas donde se ejecutan las tareas.

Central-manager: máquina desde la cual se monitorizan los nodos y las tareas enviadas.



[Figura 1]

Figura 1. Esquema de configuración de las máquinas del cluster en CICA. Contamos con una máquina servidora de un portal acceso a los recursos del cluster “cube.cica.es”, una máquina Central-manager/Submit “condor.cica.es” a la que le hemos realizado una full- install de Condor 6.8.2 y nodos execute con una instalación básica de Condor 6.8.2 que componen el cluster “piara.cica.es”. No tienen de momento espacio compartido de dominio ni de disco. Cada procesador Condor la ve como una máquina independiente. Los usuarios que deseen lanzar tareas a condor, lo podrán realizar en breve a través de la máquina servidora cube.cica.es o bien a través de condor.cica.es.

2.3. Definición de trabajos en Condor

El envío de tareas se realiza mediante un fichero de información creado para Condor : ClassAd.

El ClassAd lo debe crear el usuario para cada tarea que se vaya a lanzar.

Debe informar de la tarea en sí, su ejecutable, sus argumentos, los ficheros de datos a procesar, los requerimientos de la máquina que necesite usar...

Un breve ejemplo, supongamos lo siguiente:

Mi ejecutable es hola.exe

Necesita los argumentos `arg1` y `arg2`

Debe ejecutarse sobre Linux con arquitectura de 64 bits

Crearíamos un fichero con la siguiente información, entre otras:

```
executable = hola.exe
requirements = (Arch == "X86_64" && OpSys == "LINUX" )
arguments = arg1 arg2
```

2.4. Atributos de ClassAd

Condor facilita los atributos disponibles para la definición del ClassAd con la ejecución de comandos, que diferenciando a nivel de tareas o máquina mostrará una lista con los disponibles en cada caso.

Para la descripción de tareas:

`condor_q -l`, aparecerá una lista de los atributos disponibles.

Para la descripción de máquinas:

`condor_status -l`, aparecerá una lista de los atributos disponibles.

Ambos podrán usarse para la creación del ClassAd, ya sea como requerimiento o valor.

Una salida posible sería la siguiente:

de la ejecución de `condor_status -l`, aparecería una lista con todos los atributos a nivel de máquina de los nodos que componen el sistema de colas.

```
[...]
MyType = "Machine"
TargetType = "Job"
Name = "vm4@piara240.cica.es"
Machine = "piara240.cica.es"
CpuBusy = ((LoadAvg - CondorLoadAvg) >= 0.500000)
DedicatedScheduler =
"DedicatedScheduler@condor.cica.es"
CondorVersion = "$CondorVersion: 6.8.2 Oct 12 2006 $"
CondorPlatform = "$CondorPlatform: I386-LINUX_RHEL3 $"
VirtualMachineID = 4
VirtualMemory = 1279173
Disk = 14296957
CondorLoadAvg = 0.000000
LoadAvg = 0.000000
KeyboardIdle = 319074
ConsoleIdle = 319074
Memory = 754
Cpus = 1
Arch = "X86_64"
OpSys = "LINUX"
HasIOProxy = TRUE
CheckpointPlatform = "LINUX X86_64 2.6.x normal"
```

```

TotalVirtualMemory = 5116692
TotalDisk = 57187828
TotalCpus = 4
TotalMemory = 3017
KFlops = 860320
Mips = 2591
LastBenchmark = 1171961378
TotalLoadAvg = 0.000000
TotalCondorLoadAvg = 0.000000
ClockMin = 684
ClockDay = 2
TotalVirtualMachines = 4
[...]
```

3. Universos Condor

En primer lugar nos planteraremos, ¿qué es un universo para Condor?

“Es un entorno de ejecución”

Conjunto de recursos que Condor pone a disposición de las tareas que van a ejecutarse en el cluster.

Condor para definir sus tareas, las agrupa por universo. Cuenta con los siguiente predefinidos:

- Universo Standard
- Universo Vanilla
- Universo Parallel
- Universo Java

3.1. Universo Standard

Se usa para lanzar tareas que han sido especialmente preparadas para su ejecución bajo Condor.

Les hace adquirir características, como por ejemplo : migrar de nodo, establecer Checkpoints, que hará que Condor saque un “foto” al proceso que está corriendo y llevarlo a otro nodo en caso de fallo, así no se perderían datos ya procesados.

Un ejemplo:

```

#include <stdio.h>
int main(void)
{ printf("hello,Condor\n");
  return 0;
}

condor_compile gcc hello.o -o hello
gcc -c hello.c -o hello.o

#####
# Submit description file for hello program #
#####
```

```

Executable = hello
Universe = standard
Output = hello.out
Log = hello.log
Queue

```

3.2. Universo Vanilla

Se usa para lanzar tareas que no han sido “Condorizadas”. Es el universo mas utilizado. Devolverá los resultados a un fichero de salida.

Un ejemplo:

```

#####
#                                     #
# Example condor submit file for Matlab #
#                                     #
#####
Universe = vanilla
Executable = /afs/engr.wisc.edu/apps/bin/matlab
Arguments = -nodisplay -nojvm
Input = condor.input
Output = condor.output
Error = condor.error
Log = condor.log
should_transfer_files = yes
when_to_transfer_output = on_exit
Queue

```

3.3. Universo Parallel

El universo Parallel permite la ejecución de tareas MPI. Necesita el atributo `machine_count` que indica el número de máquinas que se van a usar para ejecutar esa tarea.

Este universo consta de un Planificador dedicado, definido en la máquina submit del sistema de colas, en nuestro caso es “condor.cica.es”, se define en la configuración de Condor con el fichero `condor_config_local`, estableciendo unas políticas de recursos dedicados. El planificador dedicado lo define el administrador del sistema, su función es reservar recursos del cluster antes de lanzar una tarea que se ejecuta bajo este universo.

Los recursos que se reservan, son los recursos dedicados, igualmente definidos en cada nodo. Cumplirán una política de ejecución, dependiendo de las peticiones de los usuarios del cluster.

Un ejemplo:

```
#####
# Example submit description file #
# for LAM MPI #
#####
universe = parallel
executable = lamscrip
arguments = my_lam_linked_executable arg1 arg2
machine_count = 4
should_transfer_files = yes
when_to_transfer_output = on_exit
transfer_input_files = my_lam_linked_executable
queue
```

4. ClassAds lanzados en el Cluster CICA

Dada la arquitectura y la configuración del cluster “piara.cica.es” hay atributos que serán comunes en el ClassAd:

- Universe : siempre habrá que indicarlo en el ClassAd.
- requirements : indicará las características de los nodos executes. En nuestro caso su valor siempre será, requirements = (Arch == “X86_64” && OpSys == “LINUX”)
- should_transfer_files : este atributo es necesario cuando no se tienen memoria compartida, como es el caso del cluster “piara.cica.es”. Por lo que su valor será “YES”
- when_to_transfer_output : este atributo indicará cuando transfiere la ejecución los ficheros de salida. En nuestro caso indicaremos “ON_EXIT”, que nos dará los resultados al término de la tarea.
- Scheduler : Tan solo cuando lancemos tareas al universo Parallel indicaremos en este atributo el siguiente valor :
Scheduler = “DedicatedScheduler@condor.cica.es”
- queue : indica que la tarea será puesta en la cola. Es un parámetro obligatorio.

Un ejemplo:

```
universe = universe
executable = executable
arguments = arg1 arg2 ...
requirements = (Arch == "X86_64" && OpSys == "LINUX")
```

```
log=logfile
output=outputfile.$(NODE)
error=errfile.$(NODE)

should_transfer_files=yes
when_to_transfer_output=on_exit
Scheduler="DedicatedScheduler@condor.cica.es"

queue
```

Tabla 1. Comandos básicos Condor.

Comando	Descripción
condor_status	Mostrará una lista con todos los nodos del sistema de colas y su ocupación. Se podrá ejecutar en cualquier nodo.
condor_q	Mostrará la lista de tareas pendientes, en ejecución o retenidas que están en cola. Tan solo se podrá ejecutar en el Central-manager, o bien añadiendo -global desde cualquier otro nodo.
condor_submit	Se debe ejecutar desde el directorio donde se encuentren todos los ficheros que usen el ClassAd (ejecutable, argumentos, transferidos...).
condor_rm	Se debe ejecutar desde el directorio donde se encuentren todos los ficheros que usen el ClassAd (ejecutable, argumentos, transferidos...).

```
Name OpSys Arch State Activity LoadAv Mem ActvtyTime
condor.cica.es LINUX INTEL Owner Idle 0.070 502 0:02:09:04
vnl@pcara218 LINUX X86_64 Unclaimed Idle 1.000 754 0:00:04:13
vnl@pcara218 LINUX X86_64 Unclaimed Idle 1.000 754 0:00:04:11
vnl@pcara218 LINUX X86_64 Unclaimed Idle 1.000 754 0:00:04:11
vnl@pcara218 LINUX X86_64 Unclaimed Idle 1.240 754 0:00:04:09
vnl@pcara225 LINUX X86_64 Unclaimed Idle 1.160 754 0:00:03:22
vnl@pcara229 LINUX X86_64 Unclaimed Idle 1.000 754 0:23:53:43
vnl@pcara225 LINUX X86_64 Unclaimed Idle 1.000 754 0:23:53:44
vnl@pcara229 LINUX X86_64 Unclaimed Idle 1.000 754 0:03:55:07
vnl@pcara226 LINUX X86_64 Unclaimed Idle 1.170 754 0:14:20:20
vnl@pcara236 LINUX X86_64 Unclaimed Idle 1.000 754 1:06:47:41
vnl@pcara236 LINUX X86_64 Unclaimed Idle 1.000 754 0:01:10:00
vnl@pcara236 LINUX X86_64 Unclaimed Idle 1.000 754 2:01:11:12
vnl@pcara237 LINUX X86_64 Unclaimed Idle 1.110 754 0:14:15:37
vnl@pcara237 LINUX X86_64 Unclaimed Idle 1.000 754 0:01:20:05
vnl@pcara237 LINUX X86_64 Unclaimed Idle 1.000 754 2:01:16:08
vnl@pcara237 LINUX X86_64 Unclaimed Idle 1.000 754 2:01:16:09
vnl@pcara238 LINUX X86_64 Unclaimed Idle 1.050 754 0:14:11:03
vnl@pcara238 LINUX X86_64 Unclaimed Idle 1.000 754 0:01:15:05
vnl@pcara238 LINUX X86_64 Unclaimed Idle 1.000 754 2:01:16:09
vnl@pcara238 LINUX X86_64 Unclaimed Idle 1.000 754 2:01:16:10
vnl@pcara239 LINUX X86_64 Unclaimed Idle 1.000 754 0:14:16:33
vnl@pcara239 LINUX X86_64 Unclaimed Idle 1.000 754 0:01:15:05
vnl@pcara239 LINUX X86_64 Unclaimed Idle 1.000 754 2:01:16:15
vnl@pcara239 LINUX X86_64 Unclaimed Idle 1.000 754 2:01:16:16
vnl@pcara240 LINUX X86_64 Unclaimed Idle 1.000 754 0:00:02:27
vnl@pcara240 LINUX X86_64 Unclaimed Idle 1.000 754 0:00:02:26
vnl@pcara240 LINUX X86_64 Unclaimed Idle 2.000 754 0:00:02:25
vnl@pcara240 LINUX X86_64 Unclaimed Idle 1.050 754 0:00:02:23

Total Owner Unclaimed Matched Preempting Backfill
INTEL/LINUX 1 1 0 0 0 0 0
X86_64/LINUX 28 0 0 28 0 0 0
Total 29 1 0 28 0 0 0
```

```
- Submitter: condor.cica.es [1-109.108.1.251:60250] - i condor.cica.es
ID OWNER SUBMITTED RUN TIME ST PRI SIZE CMD
29.0 rafaelp 2/2 13:15 0:00:00:00 I 0 5.8 g01_01_spc.sh
30.0 rafaelp 2/2 13:15 0:00:00:00 I 0 5.8 g01_02_spc.sh
31.0 rafaelp 2/2 13:15 0:00:00:00 I 0 5.8 g01_03_spc.sh
32.0 rafaelp 2/2 13:15 0:00:00:00 I 0 5.8 g01_04_spc.sh
33.0 rafaelp 2/2 13:15 0:00:00:00 I 0 5.8 g01_05_spc.sh
34.0 rafaelp 2/2 13:15 0:00:00:00 I 0 5.8 g01_06_spc.sh
35.0 rafaelp 2/2 13:15 0:00:00:00 I 0 5.8 g01_07_spc.sh
36.0 rafaelp 2/2 13:15 0:00:00:00 I 0 5.8 g01_08_spc.sh
37.0 rafaelp 2/2 13:15 0:00:00:00 I 0 5.8 g01_09_spc.sh

0 jobs; 0 idle, 0 running, 0 held
```

Figuras 1 y 2. se puede observar la salida a la ejecución “condor_status” y “condor_q”.

5. Referencias

1. Douglas Thain, Todd Tannenbaum, and Miron Livny, "Distributed Computing in Practice: The Condor Experience" Concurrency and Computation: Practice and Experience, Vol. 17, No. 2-4, pages 323-356, February-April, 2005

2. Todd Tannenbaum, Derek Wright, Karen Miller, and Miron Livny, "Condor - A Distributed Job Scheduler", in Thomas Sterling, editor, *Beowulf Cluster Computing with Linux*, The MIT Press, 2002. ISBN: 0-262-69274-0
3. Scott Fields, "Hunting for Wasted Computing Power", 1993 Research Sampler, University of Wisconsin-Madison.
4. Mark Silberstein, Dan Geiger, Assaf Schuster, and Miron Livny, "Scheduling Mixed Workloads in Multi-grids: The Grid Execution Hierarchy", *Proceedings of the 15th IEEE Symposium on High Performance Distributed Computing (HPDC)*, 2006.
5. Condor Project Homepage.

Why using dotLRN? UNED use cases

Olga C. Santos¹,
Jesús G. Boticario¹,
Emmanuelle Raffene¹
Rafael Pastor²

¹ aDeNu Research Group. Computer Science School, UNED.
C/Juan del Rosal, 16. Madrid 28040
<http://adenu.ia.uned.es>

{ocsantos,jgb,eraffene}@dia.uned.es

² Innova Section. CINDETEC, UNED.
C/Francos Rodriguez, 77. Madrid 28039
<http://www.innova.uned.es>
rpastor@dia.uned.es

Abstract. UNED uses dotLRN learning management system (LMS) in two different scopes i) Exploitation and ii) Research due to the integration capabilities, adaptivity, reusability and accessibility support. The paper introduces the main features of OpenACS/dotLRN architecture and provides an historical overview of the usage of dotLRN in Innova (exploitation) and aDeNu (research) groups. It details the reasons why OpenACS/dotLRN was chosen in both groups and presents a comparison among dotLRN and other LMS, with special emphasis on Moodle (the most commonly used open source LMS) to show their pros and cons. The paper describes how it is being used in each group and what contributions have been done to the community so far.

Keywords: Learning management systems, dotLRN, Moodle, adaptivity, accessibility, reusability, standards, personalization, use case, eLearning, universities, higher education.

1. Introduction

The Spanish National University for Distance Education (UNED) uses dotLRN learning management system (LMS) in two different scopes i) Exploitation and ii) Research. On the one hand, from the institutional side, back in 2000 Tec-InFor (Technical Unit for Research and Training in Technological Resources) proposed the use of computer supported collaborative learning (CSCL) for training faculty staff and tutors in online learning and also to promote new ways of communication and collaboration among students, tutors and staff. To develop the wide variety of required services for an increasing number of potential users (180.000 at UNED) Tec-InFor selected ArsDigita Community System (ACS) developed by the Massachusetts Institute of Technology (MIT) under GPL license. ACS was

customized to UNED's needs and called aLF (active Learning Framework) [1]. In turn, ACS evolved on time and diverse circumstances led to the creation of OpenACS, an open source community around ACS platform. On top of OpenACS, dotLRN application for learning management was developed. At UNED, aLF usage increased over the years, and in 2002 Innova Group was created to support the demands on course hosting and functional development. Three years later, aLF core (based on ACS) was migrated to aLF3 (based on OpenACS with dotLRN application on top) [2]. aLF offers an open architecture of services based on technological and educational standards upon which support the integration of ICT services and provides a set of tools for supporting courses and collaboration communities of varied nature (from administrative to research).

On the other hand, in 2003 aDeNu (Adaptive Dynamic online Educational systems based on User modelling) Research group at UNED chose dotLRN both as the platform to manage the collaborative work of aLFanet (IST-2001-33288) and SAMAP projects, and as the kernel to support the Interaction Module of aLFanet project [3]. dotLRN has been chosen for its support for adaptivity, reusability and accessibility [4, 5]. Moreover, within dotLRN infrastructure (i.e. OpenACS framework) several new packages¹ have been recently developed to support web services (e.g. XoSOAP, TWiST and SOAP-DB) which facilitates the integration of external components in the architecture. Currently, dotLRN platform goes on providing both the collaborative support and the technical infrastructure for the developments of aDeNu group research projects (FAA [6], ADAPTAPlan [7], EU4ALL [8], ALPE [9]), some of them focused on accessibility and diverse functionality issues. Moreover, the developments of these research projects are expected to provide new features to OpenACS/dotLRN following a standard-based, accessible open source approach.

The paper introduces the main features of OpenACS/dotLRN and provides an historical overview of the usage of dotLRN in Innova (exploitation) and aDeNu (research) groups. Next, the reasons why OpenACS/dotLRN was chosen in both groups are given. A detailed comparison among dotLRN and Moodle is done to show their pros and cons. The paper describes also how dotLRN is being used in each group and what contributions have been done to the community so far.

2. dotLRN Learning Environment

A wide variety of integrated solutions for learning management systems (LMS) are developed and listed elsewhere (e.g. EdTechPost or EduTech). In particular, EduTech has been performing several evaluations in the last years. In year 2003 the evaluation considered only commercial products. However, due to that many open source products have improved in quality and gained considerable acceptance in higher education organization, the 2005 the EduTech group was repeatedly asked to report on the quality of current open source e-learning and course management

¹ http://openacs.org/forums/message-view?message_id=595597

systems. In order to select the most relevant LMS, the 2005 evaluation [10] defined the following 7 killer criteria: 1) Support for multiple languages, 2) Multiple operating systems, 3) Integrated/homogeneous learning environment, 4) Active development, with at least 2 full time developers, 5) Active community, 6) Basic e-learning tools are available and 7) Basic documentation is available.

According to this evaluation, only six LMS fulfill these criteria: ATutor, Claroline, dotLRN, Ilias, Moodle and OLAT. All these systems provide basic functionality such as learners' tools (communication, productivity, students involvement) and support tools (administration, course delivery, content development) according to the features categorization defined by EduTools.

However, there are three key elements of major importance for higher education institutions that are not considered in these evaluations: adaptivity, reusability and accessibility. Regarding adaptivity, there is not currently any system that supports full adaptiveness. Nevertheless, according to [4] Moodle and dotLRN seem to be the LMS better prepared to support adaptivity. Moreover, a state-of-the-art analysis regarding accessibility and reusability (in terms of educational standards support) was performed in [5]. This review concluded that only dotLRN supports the wider range of educational standards (SCORM, IMS) and can guarantee that the functionality meets accessibility requirements.

The above works justify the selection of dotLRN instead of other LMS. Nevertheless, more detailed reasons on why dotLRN was chosen at UNED are given in the next section. Now, we present the framework and the main features of dotLRN, the support for developers and the security features.

2.1 dotLRN framework: OpenACS toolkit

dotLRN is an enterprise-class software for supporting e-learning communities developed on top of OpenACS toolkit for building scalable, community-oriented web applications. In particular, OpenACS (Open Architecture Community System)² is a n-tier architecture web application toolkit built on 1) OpenACS toolkit (OpenACS subsystem), 2) An interpretive markup language (TCL), 3) A robust HTTP server (AOLserver), 4) A mature relational database management system (Postgresql or Oracle) that follows the SQL standard and uses a procedural language (PL/SQL or PL/pgSQL) and 5) Unix-like operating system (os-nix). It works well with minimum hardware requirements (256MB RAM and 1GB hard disk space free). More information is available on the OpenACS wiki³.

OpenACS is an advanced, robust toolkit for building scalable, community-oriented web applications, dynamic content driven sites and enterprise-level web applications. It consists on a collection of pre-built applications and services upon a custom web-site or application can be built. It is derived from the ArsDigita Community System (ACS). ArsDigita (now part of Red Hat, Inc.) kindly made their work available under the GPL, making all of this possible. Through a modular architecture, OpenACS has packages

² <http://openacs.org/xowiki/docs-end-user>

³ <http://openacs.org/xowiki/openacs-system>

for user/groups management, content management, e-commerce, news, FAQs, calendar, forums, bug tracking, full-text searching, etc.

OpenACS/dotLRN strengths can be summarized as follows:

- Ready "out of the box" for common features of collaborative web sites.
- Proven architecture since components of the OpenACS/dotLRN are proving themselves in the most demanding of applications.
- Proven in the field since OpenACS/dotLRN is working well, deployed at sites that have upwards of 40.000 users.
- Responsive Community.
- Commercial Support. In this sense, a collection of commercial providers work together to maintain OpenACS/dotLRN in addition to competing for clients. Sometimes vendors work together for the same client. Most importantly, no client is ever left without support, even if his or her original provider goes out of business. There are also many independent consultants available for hire.
- Documentation is continually evolving and improving.
- Institutional commitment, including MIT Sloan School of Management which has initiated and led the development of dotLRN and many other institutions⁴.

2.2 dotLRN features

dotLRN covers a wide range of functionalities, as it can be looked up in EduTools website. EduTools is owned and operated by the Western Cooperative for Educational Telecommunications (WCET). It provides independent reviews, side-by-side comparisons, and consulting services to assist decision-making in the e-learning community. EduTools reviews are submitted typically by the product developers but EduTools reserves the right to choose whether the review is published or not, and whether changes are to be made if incorrect statements exist. They are created using a structured format to help the carried out of comparative analysis of learning systems across multiple products. The structured format was created and is maintained by the EduTools team, and has been developed and tuned since 1997.

In the case of dotLRN, the current review was submitted by the dotLRN Leadership Team⁵ on January 2007[11], corresponding to the latest release of dotLRN 2.2.1 by Jan 3, 2007. A short description of the functionality provided is presented next following EduTools structured format:

A) Communication Tools

- **Discussion Forum:** students can enable or disable posts to be sent to their email, receive posts by email as daily digests of subject lines or whole posts and subscribe to forum RSS feeds. Moreover, a spell-checker is available for student and instructor responses.
- **Discussion Management:** instructors can set up moderated discussions where all posts are screened.

⁴ <http://www.dotlrn.org/users/>

⁵ http://openacs.org/xowiki/%2eLRN_Leadership_Team

- **File Exchange:** students can submit assignments using drop boxes and share the contents of their personal folders with other students. In turn, administrators can define disk space limitations for each user.
- **Internal Email:** students can use the built-in email functionality to email individuals or groups. Instructors can email the entire class at once at a single address or alias. Both can select to forward their mail to an external address.
- **Online Journal/Notes Real-time Chat:** the chat tool supports unlimited simultaneous group discussions and archive logs for all chat rooms are created.

B) Productivity Tools

- **Bookmarks:** this feature is available in OpenACS and could be used in dotLRN with minimal effort.
- **Calendar/Progress Review:** Instructors and students can post events in the online course calendar and subscribe to RSS feeds to be notified of changes to materials. Instructors can post announcements to a course announcement page. A personal home page lists all courses in which the student is enrolled, new email and course and system-wide events.
- **Searching Within Course:** students can search all course content and all discussion threads.

C) Student Involvement Tools

- **Groupwork:** instructors can assign students to groups which have its own discussion forum and chat. Groups may be private or instructors can monitor groups. Students can also self-select groups.
- **Community Networking:** students can create online clubs, interest, and study groups at the system level. Moreover, students from different courses can interact in system-wide chat rooms or discussion forums.
- **Student Portfolios:** The portfolio is site wide. A student creates a personal page that can be used for any materials, not course specific. Moreover, dotLRN can integrate dotFolio, an open source e-portfolio system to support lifelong personal learning and development, also built on top of OpenACS web application framework.

D) Administration Tools

- **Authentication:** the system can support multiple organizational units and virtual hosts within a server configuration. Administrators can allow guest access to all courses. The system can authenticate against an external LDAP server. Administrators can set up fail-through authentication against a secondary source (e.g. the system's own database) in the event that the primary source (e.g. LDAP server) fails.
- **Course Authorization:** the system supports restricting access based on roles and roles can also be customized by the service provider. Administrators can create an unlimited number of custom organizational units and roles with specific access privileges to course content and tools and can distribute the permissions and roles across multiple institutions or departments hosted in the server environment. Instructors or students may be assigned different roles in different courses.

- **Registration Integration:** instructors can add students to their courses manually or allow students to self-register. Administrators can batch add students to the system using a delimited text file and transfer student information bidirectionally between the system and a student information system using delimited text files or IMS Enterprise Specification v1.1 XML files via web services.
- **Hosted Services:** there exist hosting and support services from Commercial Affiliates.

E) Course Delivery Tools

- **Test Types:** Different types are supported e.g. multiple choice, multiple answer, short answer, survey questions, essay. Moreover, questions can contain other media elements (images, videos, audio) and custom question types can be defined.
- **Automated Testing Management:** instructors can create self-assessments, set a time limit on a test, permit multiple attempts and specify whether correct results are shown as feedback. Students are allowed to review past attempts of a quiz.
- **Automated Testing Support:** the system can randomize the questions and answers. Instructors can create system wide test banks or import questions from external test banks that support IMS QTI 1.2.1.
- **Online Marking Tools Online Gradebook:** instructors can add grades for offline assignments, export the scores in the gradebook to an external spreadsheet and create a course grading scale that can employ either percents, letter grades, or pass/fail metrics.
- **Course Management:** instructors can personalize access to specific course materials based on group membership.
- **Student Tracking:** usage statistics can be aggregated across courses or across the institution.

F) Content Development Delivery Tools

- **Accessibility Compliance:** Self-reports show that the software complies with the WAI WCAG 1.0 Level A guidelines⁶. Moreover, currently dotLRN is working on the dotLRN Zen project⁷ to achieve level AA.
- **Content Sharing/Reuse:** instructors can share content with other instructors and students through a central learning objects repository. Moreover, tools are available to enable version tracking and linking to specific versions as well as the creation and management of workflows for collaborative content creation and review.
- **Course Templates:** course content may be uploaded through WebDAV. The system allows administrators to use an existing course or a pre-defined template as a basis for a new course.
- **Customized Look and Feel:** the system provides default course look and feel templates, and instructors can change the order and name of menu items for a course. The system can support multiple institutions, departments, schools or other organizational units on a single installation where each unit can apply its own look and feel templates as well as institutional images, headers and footers.

⁶ <http://openacs.org/xowiki/Accessibility>

⁷ <http://openacs.org/xowiki/dotlrn-zen-project>

- **Instructional Design Tools:** instructors can organize learning objects, course tools, and content into learning sequences that are reusable. Assessment package includes the possibility of creating questionnaires and within the Evaluation package learning sequences can also be defined.
- **Instructional Standards Compliance:** several standards are supported, such as IMS Content Packaging 1.1.4, IMS QTI 1.2.1, IMS Enterprise 1.1, SCORM 1.2, IMS Metadata 1.2.1 and IMS Learning Design (levels A, B and C)⁸.

G) Hardware/Software

- **Database Requirements:** the system supports Oracle and Postgres databases.
- **UNIX Server:** a Unix version is available that runs in multiple distributions⁹.
- **Windows Server:** a Windows XP version is available¹⁰.

H) Company Details/Licensing

- **Costs / Licensing:** GPL
- **Open Source:** the software is distributed under one of the OSI-approved licenses.

2.3 Developers support

From the developers point of view OpenACS framework has the following enterprise-quality features¹¹:

- High-performance XML data processing with easy, powerful Tcl scripting functionality using tDom
- XoTCL object-oriented scripting, which combines the ideas of scripting and object-orientation in a way that preserves the benefits of both
- Automated testing (Selenium and TclWebtest)
- Flexible and easy caching, for improving the performance of websites
- Programming in Tcl with AOLserver, a lightweight, simple, extremely fast scripting language that features a clean, easy-to-understand API (Application Programming Interface) for generating websites from the database
- Pooled database connections (which reduces database connection startup and teardown time), much like the technique JDBC uses, but predating it by many years
- Component package system for easy installation and upgrading of packages
- Upgrade paths for code and database schemas
- Full internationalization, including an excellent workflow for translating content into new languages
- Fully functional content repository and content management system
- An elegant templating system that separates code from presentation of content

⁸ http://openacs.org/xowiki/Educational_Standards

⁹ <http://openacs.org/xowiki/os-nix>

¹⁰ <http://openacs.org/xowiki/openacs-system-install-xp>

¹¹ <http://openacs.org/xowiki/docs-eng>

- An object system that resides on top of the database, permitting site developers to create complex applications using an object API. Examples include an object level permissions system, audit trails, and ability to relate one object to another
- OpenACS is released as open source under the GPL license, with millions of lines of open-sourced applications available to use as examples. There are also pre-written packages to use or adapt

2.4 Security features

OpenACS/dotLRN implements several important security features such as sophisticated authentication procedures, a role-based security model, scripting language, page contracts and SQL variable "encapsulation". A brief description of each of them is provided next:

- Authentication procedures. The security system must authenticate users in both secure and insecure environments. The security model implements:
 - **Cookies** (RFC 2109, HTTP State Management Mechanism): Cookies provide client side state. They are used to identify the user. Expiration of cookies is used to demark the end of a session.
 - **SHA-1**: This secure hash algorithm enables to digitally sign cookies which guarantee that they have not been tampered with. It is also used to hash passwords.
 - **SSL with server authentication (SSL v3)**: SSL establishes a secure connection between two parties and provides the client with a guarantee that the server is actually the server it is advertised as being. It also provides a secure transport.
- It supports a fine-grained, role-based and flexible permission scheme, allowing configuring access permissions to critical information sources based on roles or profiles of users, including external project members.
- Scripting Language. The code is written in a scripting language (TCL). Software written in such scripting languages is immune against "buffer overflow" errors, the most frequent source of software vulnerabilities that make up the vast majority of all Internet security holes. This is because the management of buffers is handled by the system and outside of the control of the programmer. So programmer mistakes cannot cause buffer overflows.
- Page Contracts. A "page contract" is a formal description of the input variables of an application page, similar to the procedure or method header in a procedural programming language. The reason for page contracts is that every application page is exposed to a "hostile environment" (the Internet). Every input parameter could be manipulated by malicious users. So it is important to check the type in a comfortable way for the developers.
- SQL Variables. Another frequent source of security holes in Internet applications are variables values that are included in SQL statements. Such variables can be altered by a malicious user to contain additional SQL statements in order to extract information from the DB or to cause damage. As a solution, "SQL Variables" are evaluated by the database driver. This way, the variable value is taken as a whole, effectively avoiding this type of vulnerability.

Moreover, the application service is provided using AOLServer, a leading Internet and application server for large online communities. AOLServer provides a much higher degree of security compared to popular web servers such as Apache or Internet Information Server because of its origins as the platform of America Online and the fact that AOLServer is not popular in the home user segment. This makes it less attractive to virus and worm writers. There was only a single vulnerability of AOLServer in the last four years, while there were hundredths of such incidents for the Microsoft IIS and Apache web servers. But even this vulnerability did not have an "exploit" (=computer program to exploit the vulnerability).

3. Use cases at UNED

This section describes the chronological path that led to the current usage of dotLRN at UNED. The technical reasons for these choices are based on the features provided by the tool and already described in the previous section. This section complements that justification with the specific criteria used by Innova and aDeNu groups to choose dotLRN. Moreover, this section presents also the developments that have been contributed back to the OpenACS/dotLRN community from these two groups.

3.1 Exploitation: aLF platform

aLF platform chose initially ACS and evolved to dotLRN as the kernel because of the following reasons:

- **Virtual community approximation.** All the operations made in the dotLRN LMS are around the concept of group and use cases. These use cases define the way of structuring and using all the available tools (services and applications in dotLRN language). For instance, a course in a group of users that are involved in the learning process with different roles (student, teacher, assistant teacher, tutor, on line tutor) and tools like news, grade book, evaluation, assessments, IMS/SCORM content, file storage, forums, etc.
- **User centered space** (named user portal) which contains all the personal context of user in the platform (news from courses, new messages from forums, new events in the groups calendars, open and answered assessments in courses, statistics of usage in communities, etc). Also, the user portal is customizable and can be used like a personal agenda and virtual hard disk over the Internet due to the possibility of using the calendar and file storage tools.
- **Collaborative spaces** (Communities). Another scenario that covers dotLRN is the use of different tools in a more democratic space called community. In this space there are different roles like administrator and members who can use the tools in a more collaborative mode, thanks to the permissions system of dotLRN. For example, a community with a file storage is configured to allow all members to write and create file and folders, so they are automatically shared with the rest

of members (providing the generation of shared reports via version control of documents -WebDAV-). There are other typical tools for a collaborative environment like wiki or weblog components.

- **Email centered work.** All the actions and events in dotLRN can be emailed by the use of notifications. All the objects (messages, forums, documents, events) have notifications incorporated in the programming model, so the platform can inform about every user actions in a group (write a message, download/upload a document, etc.) in personal notifications. This means that all the work in dotLRN can be supervised by email (the massive communication method), a very important feature in a distributed environment like UNED.
- **Technical efficiency.** All the components of dotLRN application are very robust and efficient (AOLserver, Oracle/PostgreSQL and TCL programming language). Also dotLRN is running as an OpenACS application, so a web application programming framework is giving support to the new tools developed in OpenACS and incorporated into dotLRN. OpenACS gives the software infrastructure (Web Services, LDAP/Kerberos authentication, IMS Enterprise support, etc.) to build several packages over it. This feature allows to build a good quality software using the general API provided by OpenACS,

Statistics show that since 2000, aLF platform usage has increased according to UNED needs in terms of users and sessions but also of functionalities provided by the tool. aLF is currently hosting more than 350 courses and communities, providing services to 55,000 students, professors and tutors. The number of users is increasing year after year on 22% average. On the other side, the number of sessions is increasing more significantly, 33%, with an average of 4 millions transactions per day.

Contributions from Innova group to OpenACS/dotLRN community

Innova has been working with the community of developers since 2000. The main work load is used on porting several packages and applications to have complete Oracle support for main core distribution of dotLRN. It also provides for the community test servers both for PostgreSQL and Oracle databases, in order to do bugs sessions (necessary for code errors resolution). Moreover, several improvements have been added in different packages (forums, file storage, calendar, user tracking, learning object repository system) and new packages like portlets for new forum messages, new calendar events and new documents/hiperlinks created in the file storage.

Due to the particularities of UNED, new portals named Faculty and Department spaces have been developed which replicate the organizational structure of the university. These spaces allow professors and staff to have a common virtual community in which they can exchange information using the documents space (with version control, individual and group permissions, folders structure, files and hiperlinks storage) , forums and calendar. The calendar is very useful to the staff member in order to communicate to professors events related to the administrative work in the faculty or department (like official meetings for discussing faculty problems and make the corresponding decisions). All the calendar events are

programed to send email automatically, a predefined time before the event (defined by the administrator user). Of course the complete integration of UNED structural model has been achieved: automatic student enrollment, teachers teams enrollment and administration, tutors enrollment, course portals customized to the UNED pedagogical model, faculty and department portals accessibility from UNED members intranet, collaborative communities for different staff departments of UNED (vice rectors offices, rector government team, government team UNED, Innova, USO-PC, COIE, IUED, etc.).

3.2 Research: aDeNu platform

aDeNu research work focuses on how to cope with adaptivity, reusability and accessibility issues in learning management systems to improve the learners experience at eLearning settings. In 2003 aDeNu (Adaptive Dynamic online Educational systems based on User modelling) Research group at UNED chose dotLRN both as the collaborative platform to manage the workload of aLFanet (IST-2001-33288) and SAMAP projects, and as the kernel to support the Interaction Module of aLFanet project [3]. On the one hand, the following features were demanded to manage the development of the project:

- Common space to work on the project reports, with support for different versions of the same document
- A file storage where permissions could be easily administrated to control the scope of the deliverables of the project
- Discussion forums where project decisions were taken
- Notifications send to users' e-mail of the messages posted in the forums to be aware of new contributions
- Event management in a calendar to coordinate the project milestones
- Subgroups to manage, in an independent way, the different packages of the project
- An integrated view of the contributions provided in the different subgroups
- An easily configurable community interface, to distribute the functionality in the community space in a way that better suits the members' needs
- Permissions to grant different access rights to the project coordinator and workpackage responsables

On the other hand, aLFanet Interaction Module had to provide the different educational services that allow the learners to perform a course in the system, using both modes: individual learning and collaborative tasks. In particular, it should deal with collaborative services in three different context: the personal area (workspace), the course area and the activities area. For the personal area it had to provide an integrated vision of each one of the services and contents the user has access in the different courses s/he is enrolled, and the contributions performed in them. The course area had to provide the services that have to be used each course, both if they are specified at design time by the author or at run time by the tutor or professor. The

activity area had to be similar to the course area, but under the scope of an activity inside the course (some activities may require additional and specific services to be performed, such as collaborative activities). The collaborative functionality to be provided was: Forums, File Storage, Calendar, Notifications, Comments, News and FAQ's. For the tutor, (an administrator of the course), the Interaction Module had to provide administration functions for each one of the services. All this functionality was available in dotLRN. Other functionality, not provided by dotLRN, had to be developed in aLFanet, such as Ratings and Access to information about Learning Objects.

The management of aLFanet in dotLRN was a great success, with more than 500 files uploaded, near 30 forums created, 70 members and 10 subgroups. For the following research projects, dotLRN was migrated to version 2.2, to benefit from the following functionalities: internationalization, support for educational standards, friendly interface (more usable and accessible), user tracking, blogs, RSS, feeds and wiki pages. Currently, there are 8 active communities in aDeNu platform to manage the different projects aDeNu member are involved. 165 users are taken part in them, contributing to more than 150 forums and 100 top level folders.

Apart from the usage for the coordination of current projects (FAA [6], ADAPTAPlan [7], EU4ALL [8], ALPE [9]), OpenACS/dotLRN framework is also being used as the basis for the project developments, which deal with adaptation, accessibility and educational standards support. In the next subsection, we present the contributions to OpenACS/dotLRN community from aDeNu group. The support for SOAP services is being used to integrate existing web service systems developed by other project members.

Contributions to OpenACS/dotLRN community from aDeNu group

aDeNu group has contributed and continuous contributing to OpenACS/dotLRN community in several ways, mainly consultancy (bug reporting¹² and accessibility evaluations¹³), workload (release management¹⁴ and bugs fixing¹⁵, participation in periodic Board¹⁶ and Leadership team¹⁷ meetings), dissemination (presenting dotLRN functionality and usage at international conferences¹⁸ and journals) and source code delivery. Regarding this last item, the main contributions produced by aDeNu research works to the community includes:

- aLFanet Interaction Module [3]:
 - The first support for IMS Learning Design (*course-install package*) was developed at aLFanet project¹⁹.
 - Comments and ratings to learning material

¹² <http://openacs.org/bugtracker/openacs/>

¹³ <http://openacs.org/storage/view/openacs-dotlrn-conference-2006/dotLRN-accessibility.ppt>

¹⁴ http://openacs.org/xowiki/%2eLRN_2%2e3

¹⁵ http://openacs.org/xowiki/%2eLRN_2%2e2_bugs

¹⁶ http://openacs.org/xowiki/%2eLRN_Board_of_Directors

¹⁷ http://openacs.org/xowiki/%2eLRN_Leadership_Team

¹⁸ <http://dotlrn.org/news/one-entry?entry%5fid=162850>

¹⁹ http://openacs.org/forums/message-view?message_id=325767

- Access to IEEE-LOM metadata in *imsl-d-lo package*
- Development produced by students under the direction of aDeNu members:
 - Logic Framework Approach²⁰. The standard version is already implemented, and the collaborative extension is under development.
 - Implementation of Felder Learning Styles Index (to be tested in Oracle)
 - Support for IMS QTI Selection & Ordering features (still at initial stages)
- dotLRN Zen project. aDeNu members are very directly involved in this project to improve the accessibility level of dotLRN²¹, working on the clean up of HTML code.
- Improvements of educational standards packages. Several functionality tests have been performed for the functionality offered (e.g. LORS²², IMS-LD²³ and Assessment²⁴), as well as external accessibility evaluations [12].

4. dotLRN vs. other LMS

In 2006 a internal report was written in order to compare several approaches of LMS. The report includes a commercial LMS (WebCT, also used at UNED) and two FLOSS LMS: dotLRN (in particular, aLF, the UNED customization of dotLRN) and Moodle. The comparison analyzed the following LMS: WebCT (versión 4.1), dotLRN (version 2.2.0) and Moodle (version 1.6). The first two were chosen for being used at UNED ever since 2000, and Moodle for being an emergent LMS with a wide projection in the LMS market. To carry out the comparison, quality factors were chosen taking in account the different roles involved in using the platforms, namely managing and administrating courses/communities. The chosen criteria were defined so that they can be weighed according to the particular objectives that a given institution would like to apply at any given moment. The final purpose here is to support the decision making process on ICT resources to be included in the institution strategic plan. In Table 1 the chosen criteria (mostly technical) are shown with a valuation for dotLRN and Moodle. These valuations come from other comparisons performed by WCET and EduTech, which provide additional information on the most common features provided by current LMS.

Table 1. Comparison table for dotLRN and Moodle

Criteria	dotLRN	Moodle
Development	High, complete customization	High, complete customization support
Usability	User centered approach focused in collaborative work (forums, blogs, activities)	Pedagogical approach centered in controlled by teacher,

²⁰ http://openacs.org/forums/message-view?message_id=499254

²¹ http://openacs.org/xowiki/dotlrn-zen-project_packages

²² http://openacs.org/forums/message-view?message_id=590351

²³ http://openacs.org/forums/message-view?message_id=578693

²⁴ http://openacs.org/forums/message-view?message_id=444217

	shared document space, calendar), need an special usability model for courses	very intuitive for courses
Software Enterprise Architecture	Web Framework based (OpenACS). Multilayer services oriented for enterprise development	Scripting based, with no software infrastructure model, multiple maintenance and security problems
Applications availability	A lot of services and applications, used in several domains: courses, communities, e-business, electronic administration.	A very high number of modules, all to be used only for courses.
E-Learning Standards support	IMS-CP, IMS-QTI, IMS-LD, SCORM, IMS-MD	IMS-CP, IMS-QTI, SCORM
Development Control	Consortium based, all the releases are controlled by a leadership team elected by the consortium members	Managed by a person, with a group of collaborators. All the officials release decisions are personal.
Quality Level of Security	High, the multilayer services model allows to isolate the security problems (security vulnerabilities are located easily)	Poor, the php scripts based infrastructure forces to update or patch a lot of code
Collaborative support	There are specific spaces for collaborative work (communities), among specific tools (wiki, blogs, shared documents space,...)	No direct support for collaborative work, but it can used several tools inside the courses for collaborative learning.

Another detailed comparison among OpenACS and LAMP (Linux-Apache-MySQL-Php) applications is available at the OpenACS wiki²⁵. That comparison covers the following issues: a) AOLserver vs Apache, b) PostgreSQL or Oracle vs MySQL, c) OpenACS Business Logic and Data Model vs Perl/PHP/Python, and d) OpenACS vs LAMP communities to show why OpenACS is a preferred approach to developing scalable, complex web applications instead of LAMP approaches.

5. Discussion

Higher Education reforms aims at supporting a “student-centered approach” for lifelong learning based on ICT applications which integrate educational, management and research services. To make such personalized framework sustainable universities are selecting the best combinations of open source software (OSS) solutions and standard-based service architectures to supports application integration and cost reduction. Current challenges go beyond implementing learning and teaching strategies which focus on the promotion of students’ learning in a personalized way and insist on addressing the lifelong learning (LLL) paradigm, where students are to be equipped, following a student-centered approach, with the attitudes and skills to learn for themselves both in formal education and long after

²⁵ http://openacs.org/xowiki/OpenACS_Lamp_Comparison

they have graduated [13]. Furthermore, instead of relying on a single technology platform or a few proprietary software vendors, universities should be able to select the best combinations of available software, where open source software and an open architecture supports application integration and cost reduction [14].

Current efforts are being made, especially in mega-universities with thousands of users and a wide variety of services, to provide a more integrated provision of ICT services. This objective is quite relevant considering the interoperability required to provide many functionalities. For instance, depending on the profile of a particular student (i.e., background knowledge, special needs, preferences, etc.) alternative sources of information and consultancy services can be provided. Likewise, for instance, a lecturer can develop contents, syllabus, and course calendars in a non-proprietary standard format that could be eventually delivered via alternative means (web, printed study guides, books, etc.).

These objectives are essential at UNED, and therefore since 2000 our institution has relied on the combination of open software tools, like dorLRN with the rest of the provided ICT services. aLF platform usage has increased over the years according to UNED needs in terms of users and sessions but also in terms of functionalities provided by the tool. The usage of the platform at UNED has evolved from the initial support of collaboration tasks in communities of varied nature and the eLearning training of tutors and faculty to a more global coverage of administrative departments, faculty units and university departments, as well as courses offered in different educational areas: PhD, undergraduate, ongoing education and occasional courses came from institutional agreements with other institutions worldwide, specially with educational institutions in Latin America.

6. Conclusions

Ever since 2000 UNED's headquarters and aDeNu research group at UNED have been using dotLRN and its predecessors for exploitation and research purposes. The former covers two types of issues. First, collaboration support for a wide variety of uses: study groups, users' associations, administrative services, faculty groups, research projects, etc. Second, eLearning services for end-users and for the faculty and tutors as well. The latter focuses on how to cope with adaptivity, reusability and accessibility issues in learning management systems to improve the learners experience at eLearning settings.

The contributions of both groups to the community of developers have been also of diverse nature. On the one hand, the Innova group (i.e., the current headquarters' unit supporting dotLRN) has been working with the community of developers on porting several packages and applications to have complete Oracle support. It has also provided test servers both for PostgreSQL and Oracle databases, and several improvements have been added in different packages, as well as new packages for specific purposes. On the other side, aDeNu has provided an interaction module which first supported IMS Learning Design integration in the aLFanet project, which includes comments and ratings to learning materials and access to IEEE-

LOM metadata. More recently, with the collaboration of last year students of the Computer Science School at UNED, several modules have been developed: a CSCL framework called the Logic Framework Approach, implementation of Felder Learning Styles Index, and the support for IMS QTI Selection & Ordering feature, which is currently under development.

Throughout this paper dorLRN distinctive features have been introduced, comparing them with other well known LMS, stressing their respective pros and cons, and showing their usage by the aforementioned groups.

All these developments have been supporting a continuous improvement process in the delivery of ICT services at UNED, whose current challenges are framed within the so called European Higher Education Area, and in general, with a wider perspective aiming at supporting a personalized life long learning paradigm, which takes into consideration individual needs of ALL, including those related to accessibility and functional diversity issues.

Acknowledgements

Special thanks to UNED's Innova teamwork over these years (2000-2006), to the responsible people who support the development of aLF at UNED and to the European Commission for funding some of the aDeNu group research projects (i.e., ALFANET, EU4ALL and ALPE). Likewise, the authors would like to express their gratitude to the Ministry of Education in Spain for supporting the SAMAP and ADAPTAPlan projects and to the Plan Galego de Investigación, Desenvolvemento e Innovación tecnolóxica, Xunta de Galicia, for their support to the FAA project.

References

1. Boticario, J.G., Gaudioso, E. and Catalina, C.: Towards personalised learning communities on the Web. First European Conference on Computer-Supported Collaborative Learning, pp. 115-122 (2001)
2. Boticario, J.G., Raffenne, E., Aguado, M., Arroyo, D., Cordova, M.A., Guzmán, J.L., García, T., Hermira, S., Ortíz, J., Pesquera, A., Romojaro, H. and Valiente, S.: Active Learning in aLF through Virtual Communities and CSCL. In Proceedings of the workshop "LT applied research in Spain", within the joint meeting between CEN/ISSS Learning Technology and IEEE Learning Technology Standardization Committee. UNED, (2004).
3. Santos, O.C., Boticario, J.G., Barrera, C.: aLFanet: An adaptive and standard-based learning environment built upon dotLRN and other open source developments. Foro hispano de .LRN. Spain, (2005)
4. Kareal, F. and Klema, J.. Adaptivity in e-learning. In A. Méndez-Vilas, A. Solano, J. Mesa and J. A. Mesa: Current Developments in Technology-Assisted Education, Vol. 1, pp. 260-264, (2006)
5. Santos, O.C.: Technology Enhanced Life Long eLearning for All. In K. Maillet and R. Klamma: Proceedings for the 1st Doctoral Consortium on Technology Enhanced Learning. European Conference on Technology Enhanced Learning, p.66-71, (2006)

6. Santos, O.C. and Boticario, J.G.: Open and Accessible Training. In Méndez-Vilas, A., Solano, A., Mesa, J. and Mesa, J.A.: Current Developments in Technology-Assisted Education, Vol. 2, pp.1107-1110, (2006).
7. Santos, O.C. and Boticario, J.G.: An alternative for Learning Design: automatic support for learning routes generation in ADAPTAPlan. In Proceedings of the 13th International Conference on Artificial Intelligence in Education - AIED 2007 (in press, 2007)
8. Santos, O.C. and Boticario, J.G.: European Unified Approach for Accessible Lifelong Learning'. In Méndez-Vilas, A., Solano, A., Mesa, J. and Mesa, J.A.: Current Developments in Technology-Assisted Education, Vol. 2, pp. 1102-1106, (2006)
9. Santos, O.C., Boticario, J.G., Fernández del Viso, A., Pérez de la Cámara, S., Rebate Sánchez, C., Gutiérrez y Restrepo, E.: Basic skills training to disabled and adult learners through an accessible e-Learning platform. In proceedings of the 12th International Conference on Human-Computer Interaction – HCI 2007 (in press, 2007)
10. Edutech, Evaluation of Open Source Learning Management Systems – 2005. Available at: <http://www.edutech.ch/lms/ev3/>, (2005).
11. EduTools, dotLRN/OpenACS review (January 2007)
Available at: <http://www.edutools.info/compare.jsp?pj=4&i=562> (2007)
- 12.Revilla Muñoz, O.: Accessibility Requirements for the educational packages in dotLRN. 1st International Conference on FLOSS: Free/Libre/Open Source Systems. (2007, in press)
- 13.Knapper, C.K. And Cropley, A.J.: Lifelong learning in higher education. Kogan, (2000)
- 14.Carey, P.F. and Gleason, B.W.: Business Challenges and Innovative Application of Technology in Higher Education. IBM white paper, (2005)

Authors

Olga C. Santos is the R&D Technical Manager of aDeNu (Adaptive Dynamic online Educational systems based on User modelling) Research Group at UNED and belongs to aDeNu group since 2001. Her current research interest focuses on taken into account adaptation features and accessibility requirements based on combining user modelling and machine learning techniques in multi-agent architectures and making a pervasive use of educational and technological standards, and open architectures and technologies. She has coordinated the technical developments of aDeNu research projects (aLFanet, SAMAP, FAA, ADAPTAPlan, EU4ALL and ALPE) and published several papers in various international conferences and journals regarding the results obtained in them. She is member of the Leadership team of dotLRN.

Jesús G. Boticario, is an Associate Professor of the Artificial Intelligence Department at the School of Computer Science (CSS) at UNED. He has held several positions at UNED in the area of e-learning and ICT's (Director of Innovation, Innovation and Technological Development Vice-principal, General Director of the Centre of Innovation and Technological Development). He has published more than 100 research articles in the areas of adaptive interfaces, user modelling and e-learning. He has participated in 13 R&D funded projects. He is currently the head of the aDeNu Research group and the scientific coordinator of funded projects in the area of eInclusion. He is the coordinator of the ICT-Rectors Working Group on Accessibility Issues of the Higher Education Board of Rectors in Spain. He belongs to the Board of Directors of the dotLRN LMS.

Emmanuelle Raffenne is a currently a member of aDeNu (Adaptive Dynamic online Educational systems based on User modelling) Research Group at UNED. Her current work focuses on improving the accessibility of LMS systems such as dotLRN, the open source e-learning system built on top of OpenACS toolkit for community-oriented web applications. She is a member of the Leadership team of dotLRN, and strongly involved in the release process and management of the toolkit. She has been coordinating the Innova group, in charge of the development of the UNED eLearning platform aLF, since 2000.

Rafael Pastor received his M.S. degree in physics in 1994 from University Complutense of Madrid and his Ph.D. in 2006 from UNED. Since 1994, he has been working at UNED Department of Computer Sciences and Automatic Control as an Assistant Professor. Also currently he is working as Innovation Manager of the Innovation and Development Centre of UNED since 2004, adding innovative services in the learning model of

UNED. As part of his work, he is the head of the open source UNED e-learning platform called aLF (Active Learning Framework). He is member of the IEEE Spanish Education Society and IEEE member.

***Euspell*: corrección ortográfica del euskera en software libre**

Iñaki Alegria¹, Klara Ceberio¹, Nerea Ezeiza¹, Aitor Soroa¹, Gregorio Hernandez²

¹ Grupo Ixa, Euskal Herriko Unibertsitatea
i.alegria@ehu.es

² Eleka S.L.

Resumen. En este artículo se presenta la primera versión en software libre del corrector ortográfico para el euskera desarrollada entre el grupo de investigación IXA de la Universidad del País Vasco y la empresa Eleka. Debido a la complejidad morfológica del euskera las soluciones basadas en los programas libres *ispell*, *aspell* y *myspell* no son satisfactorias y ha habido que esperar a la difusión de *hunspell* para encontrar una herramienta de software libre idónea para idiomas de morfología compleja. La solución propietaria de nombre *Xuxen* ha tenido un gran éxito, por lo que esperamos que la distribución de *euspell* sea un éxito en un gran número de aplicaciones, sobre todo como complemento a los programas de ofimática (*OpenOffice*), edición para blogs (*Firefox2*) y correo electrónico. Como estos últimos aún no han adoptado *hunspell*, se ha desarrollado una versión derivada compatible con *myspell*.

Keywords: software libre, corrección ortográfica, ingeniería lingüística, morfología

1. Introducción

Desde 1992 está disponible de forma gratuita un corrector ortográfico para el euskera llamado *Xuxen*, que inicialmente fue desarrollado como producto independiente para PC y Mac reconociendo los formatos de *MSWord* y *WordPerfect* y posteriormente como plugin de *MSOffice* para PC y Mac. Adicionalmente se ha preparado una versión especializada para corrección en OCR. También hubo un desarrollo cerrado para *OpenOffice1*. Sin embargo el desarrollo de un software libre de corrección ortográfica para el euskera se ha demorado mucho por diversas razones. Las más importantes son las siguientes y están estrechamente relacionadas entre ellas:

1. el carácter aglutinante de la morfología vasca
2. la falta de un estándar adecuado en el entorno de software libre para lenguas de alta flexión.
3. la no disposición en software libre de un equivalente de la librería de Xerox¹ que se utilizaba en la mayoría las aplicaciones mencionadas anteriormente.

¹ www.xrce.xerox.com/competencies/content-analysis/fst/

Hay que tener en cuenta que el corrector ortográfico base es muy complejo y eficiente.

Morfología del euskera

La morfología del euskera es especialmente rica sobre todo a nivel de flexión [1]. No existen las preposiciones y, sin embargo, hay un gran número de casos de declinación, que además varían con el número y la determinación. Los dos casos de genitivo (lugar y persona) son aglutinantes, es decir, tras el genitivo pueden aparecer cualquier otro caso.

El verbo auxiliar cambia además de en función del tiempo verbal y la persona del sujeto, en función de la persona del objeto y la del objeto indirecto.

La derivación también es bastante rica, sobre todo con respecto a la nominalización del verbo. Cualquier raíz verbal puede ser nominalizada y a partir de ahí tomar toda la flexión de los nombres. También las formas conjugadas del verbo auxiliar y principal pueden nominalizarse y encadenar todos los sufijos de nombre.

Por ejemplo, en el sintagma *alabaren etxean* (en la casa de la hija) los lemas son *alaba* (hija) y *etxe* (casa), siendo la *a* de los sufijos morfema de determinante, *ren* morfema de genitivo y *n* morfema de inesivo. Pero también se podría hacer una elipsis y decir o escribir *alabarenean* (en la de la hija). Incluso se podrían permitir dos elipsis, por ejemplo en *etxeakoarena*, que se se podría traducir *el de la de la casa*, es una forma que en un buscador se puede encontrar. Una palabra no demasiado extraña como *egitekoarekin* (algo así como *con lo que se ha de hacer*) se compone de morfema: *egin* (hacer), *te* (nominalización), *ko* (genitivo singular), *aren* (genitivo singular) y *kin* (asociativo)².

Con esta morfología el número de formas que pueden ser generadas a partir de una raíz varían según la categoría, pero para los nombres, que es la categoría más común, es más de mil. Por lo tanto una solución a la morfología o a la corrección ortográfica basada en una lista de palabras no es adecuada.

En Europa son idiomas de similares características morfológicas y con la misma problemática para la corrección ortográfica el finlandés, el húngaro y el turco.

Morfología de dos niveles y corrección automática

Para este tipo de idiomas la mejor manera de implementar un corrector ortográfico es basarse en un analizador morfológico [4]. Así, una palabra será presumiblemente correcta si a partir de ella es posible obtener algún análisis morfológico correcto. A la hora de hacer propuestas para las palabras incorrectas, se sigue el denominado método inverso, es decir, se generan posibles palabras introduciendo cambios a partir

² Buscando en *google* *alabarenean* aparece 9 veces, *etxeakoarena* 3 y *egitekoarekin* más de 100.

de la original y se comprueba que existan en el idioma verificándolas morfológicamente. En consecuencia el análisis deberá ser rápido.

Para crear un analizador morfológico para este tipo de idiomas se ha demostrado que la morfología de dos niveles y sus mejoras [2][6] son los formalismos más adecuados. Las ventajas que ofrece son muy importantes:

1. Diferencian el léxico, la morfotáctica y la morfofonología con lo que la definición y el mantenimiento de toda la información es más sencillo.
2. No se pone límites a ninguno de los fenómenos morfológicos nombrados, por lo que se pueden considerar formalismos universales.
3. Se compila en transductores de estados finitos, por lo que la eficiencia del analizador está garantizada.
4. Sirve tanto para análisis como para generación, por lo que el trabajo puede ser reutilizado para distintas aplicaciones (la generación se puede usar en sistemas de traducción automática por ejemplo).
5. Permite añadir y compilar conjuntamente léxicos y reglas no estándares, lo que permite corregir muy precisamente los errores de competencia léxica.

Debido a estas ventajas en 1993 adoptamos esta formalismo y definimos primero nuestro analizador/generador morfológico y, posteriormente, el corrector ortográfico. Estas herramientas son muy precisas pero tienen el inconveniente que usan una librería propietaria.

2. Software libre para corrección ortográfica

Desgraciadamente no existe software libre estandarizado para implementar morfología de dos niveles. Según nuestras referencias, solamente *mmoph*³ aplica un formalismo de ese tipo, pero con un inconveniente, solo funciona para generación. Para el análisis se deben generar todas las formas posibles, lo cual es inviable para el euskera.

Con respecto a la familia de correctores ortográficos estándares clásicos para software libre (*aspell*, *ispell* y *myspell*), podemos decir que no ofrecen la expresividad necesaria para una fácil adaptación. Hay que tener en cuenta que siempre se han diseñado para el inglés y después se han intentado aplicar a otros idiomas.

ispell es el programa más antiguo de ellos y el que diseñó la división de los datos lingüísticos entre raíces y afijos. Adicionalmente permite reglas para eliminar y/o añadir caracteres cuando se encadena un lema con un afijo. Aunque esto le da algo de suficiente expresividad para el inglés y los idiomas latinos, es insuficiente para idiomas de las características del euskera, entre otras razones porque permite un número muy limitado de paradigmas⁴.

³ packages.debian.org/unstable/misc/mmorph

⁴ Llamamos paradigma a cada conjunto diferenciado de sufijos que puede encadenarse tras un lema

aspell es el corrector estandar para GNU pero las únicas mejoras que aporta están dirigidas a mejorar las propuestas a los errores, y no a soportar idiomas morfológicamente más ricos.

myspell es una implementación en C++ parecida a *ispell* y orientada a integrarla en *OpenOffice* y aunque permite algo más de expresividad para la flexión, esta no es suficiente.

Las características y diferencias precisas entre estas herramientas pueden ser consultadas en la wikipedia (en inglés).

Hubiera sido sencillo prepara una lista de palabras integrable en *aspell*, o con más trabajo hacer una adaptación a los requisitos de *ispell*, pero esto habría presentado dos problemas que consideramos importantes:

1. falta de coherencia de los diagnósticos del corrector. Mientras ciertas declinaciones de un lema serían tomadas por correctas, otras algo menos frecuentes se darían por erróneas (falsos negativos). Dicho de otra forma, no se consigue una cobertura adecuada.
2. necesidad de mantenimiento de dos grandes y complejas bases de datos para el mismo problema.

Ante esta situación nuestro propósito fue solventar ambos problemas con el desarrollo de un software que posteriormente fuera liberado y presentado para su estandarización en el entorno del software libre. Sin embargo esta tarea demoró y fue más compleja de lo que esperábamos, y, además, coincidió con el desarrollo y estandarización de *hunspell* [7].

hunspell desarrollado inicialmente para el húngaro, soluciona la mayoría de los problemas señalados (salvo el de no obtener buenas propuestas para errores de competencia léxica), ya que permite mayor número de paradigmas y doble encadenamiento de sufijos y, por lo tanto, nos permite generar, a partir de la definición registrada en la base de datos, un corrector preciso y de gran cobertura. Además ya ha sido adoptado por *OpenOffice* y va a serlo por *Mozilla*.

3. Adaptación

Para hacer la adaptación de los datos lingüísticos del euskera se ha llevado a cabo un proceso de exportación/importación entre el sistema original, basado en la morfología de dos niveles y los transductores léxicos propuestos por Karttunen [6], a la especificación de *hunspell*. Los pasos necesarios (simplificados) han sido los siguientes:

1. Se han desarrollado en el sistema original reglas adicionales para restringir el poder generativo de la morfología vasca. Debido al gran poder generativo, ya descrito en la sección 1, la listas de afijos podía ser interminable. La decisión fue restringir el número de sufijos a tres, con ciertas excepciones que permiten sufijos adicionales (sufijos de plural, de nominalización, genitivo).
2. Añadiendo al sistema original esas reglas se ha construido un analizador/generador restringido cuyo funcionamiento será comparado con el del resultado final. Este sistema será usado también en las siguientes etapas.

3. Por un lado, de las reglas fonológicas se han obtenido las terminaciones de lema que pueden conllevar cambios al encadenarse con sufijos⁵. Por otro lado, se ha obtenido del léxico original un lema como representante de cada paradigma y terminación, agrupando las terminaciones que no producen cambios especiales en dos grupos, vocales y consonantes. Posteriormente, usando el generador construido en el paso anterior, se han generado todas las formas posibles para los lemas seleccionados.

4. A partir de las formas generadas por cada lema y el lema original se ha logrado separar las raíces y los afijos, generándose así un primer nivel de encadenamiento. Esto podría ser suficiente para *myspell*, pero como se ha mencionado, aparecen dos problemas: demasiados paradigmas y listas de afijos demasiado largas.

5. En los sufijos correspondientes a cada paradigma se ha buscado la aparición del genitivo (ya que es el caso que multiplica el poder generativo) y se ha dividido el sufijo en dos partes, la anterior y la posterior. La primera parte se ha mantenido en un primer nivel de encadenamiento y la segunda ha pasado a un segundo nivel, que es aceptado por *hunspell*.

6. Debido a que conjuntos de sufijos de segundo nivel pueden aparecer repetidos, se han identificado esos conjuntos repetidos minimando el segundo nivel de sufijos.

7. Finalmente se han añadido todos los lemas al fichero de raíces, modificando, cuando era necesario, los últimos caracteres de la misma forma que se hizo con su representante en el paso 4.

8. Tras estos pasos se ha conseguido la información necesaria para su adaptación a *hunspell*. Con un simple cambio de formato se ha terminado la conversión. Tras diversas pruebas y correcciones, los ficheros se han liberado en enero de 2007, con una nueva versión en febrero.

4. (Re)Utilización

A continuación describimos dos productos adicionales obtenidos en este proceso y la utilización de las herramientas de corrección ortográfica.

Adicionalmente la descripción morfológica generada ha sido reutilizada en otras dos herramientas interesantes: versión *myspell* para el euskera y generador morfológico para un programa libre de traducción automática.

Datos del euskera para *myspell*

Debido a que *myspell* no puede gestionar todos los paradigmas generados, se han preparado los datos necesarios para *myspell* seleccionando los paradigmas más

⁵ Por ejemplo la *a* final suele desaparecer, las terminaciones verbales *tz*, *ts* y *tx* al encadenarse con una consonante suelen perder la *t*, la *r* final suele doblarse delante de vocal, etc.

probables y añadiéndole a las raíces una larga lista de palabras obtenidas de la siguiente forma:

1. Generación de formas más habituales a partir de los lemas más probables que no están en esos paradigmas previamente incluidos. Para hacer esta generación hemos utilizado el generador morfológico original restringiendo el poder generativo a un solo sufijo.
2. A partir de un gran corpus de euskera, hemos obtenido todas las formas correctas usando el corrector original y hemos detectado aquellas que *myspell* no podía reconocer, ni con los paradigmas más frecuentes, ni con la lista anterior. La lista obtenida se ha añadido.

Esta versión de *myspell* la hemos hecho pública, ya que funciona bastante correctamente a pesar de que presenta la falta de coherencia comentada en el apartado 2. Hay que tener en cuenta que muchas herramientas, como las de *Mozilla*, no integran de momento *hunspell*.

Generación morfológica en *Matxin*

Además se ha usado la exportación conseguida, con información morfológica adicional, para integrarla en la generación morfológica de un ingenio libre de traducción automática de nombre *Matxin*⁶ [3], que junto al ingenio de nombre *Apertium*⁷ [5] (ver en estas actas) se han desarrollado dentro del proyecto *Opentrad*⁸. Aquí se demuestra lo conveniente de combinar corrección ortográfica con análisis/generación morfológica, ya que así a partir de un mismo desarrollo se pueden generar distintos productos. Esta misma idea está en el propio paquete *hunspell*, que desde su diseño ha apostado por la doble función.

Utilización

Los paquetes para el euskera en *hunspell* y *myspell* han sido desarrollados por el grupo IXA de Universidad del País Vasco y la empresa Eleka y se distribuyen bajo licencia GPL. Se pueden obtener en el sitio de descargas del Gobierno Vasco⁹ y en los sitios correspondientes a las herramientas de software libre¹⁰.

A continuación se presenta un ejemplo de la utilización en OpenOffice:

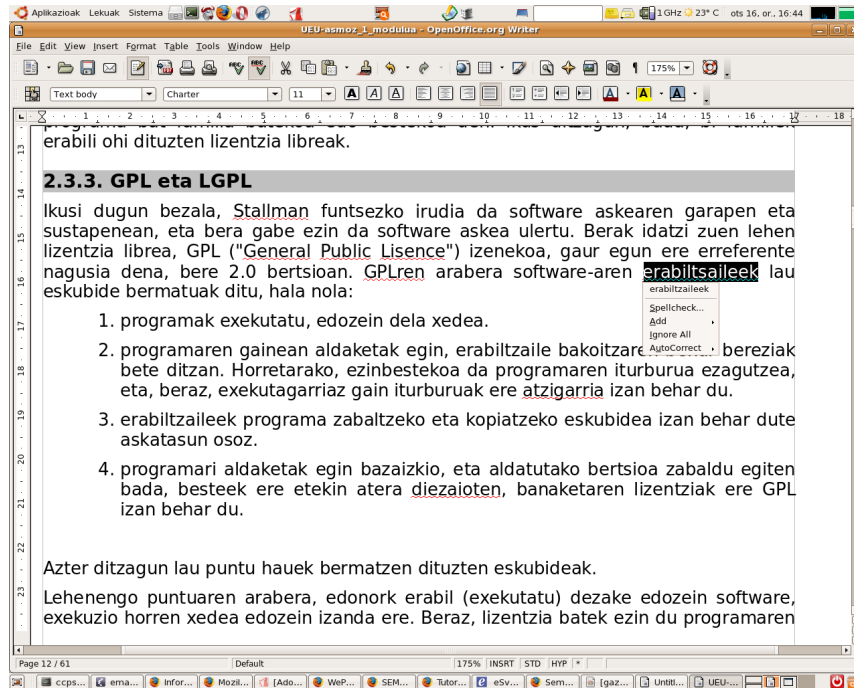
⁶ matxin.sourceforge.net

⁷ apertium.sourceforge.net

⁸ www.opentrad.org

⁹ <http://www.euskara.euskadi.net/>

¹⁰ <http://www.librezale.org/mozilla/firefox/> entre otros



5. Mejoras y trabajo futuro

La versión generada para *hunspell* presenta dos limitaciones que pensamos deben ser mejoradas:

1. Al no haberse utilizado todas las potencialidades de *hunspell*, los datos todavía pueden mejorarse, sobre todo para conseguir una mayor compactación de los mismos y, por lo tanto, mayor eficiencia en su acceso. Para ello se pueden generar reglas fonológicas de eliminación/adición de caracteres al encadenar lemas y sufijos, con lo que se reduciría el número de paradigmas. Esto se puede realizar a muy corto plazo.
2. De todas formas, el uso de *hunspell* no resulta totalmente satisfactorio ya que no soporta *hunspell* más de dos encadenamientos de sufijos. Aunque esto lo hemos paliado alargando la lista de sufijos del segundo nivel, sería más conveniente permitir más niveles de encadenamiento de sufijos. Este es un trabajo a un plazo más largo.

También queremos poner en marcha un sitio de referencia para futuras mejoras y desarrollos relacionados.

En cualquier caso, de cara al futuro, sería conveniente, y creemos que este tema ya está planteado en la comunidad científica, la generación de un FLOSS de las mismas características expresivas de los transductores léxicos. Con esto se conseguirían

procesadores morfológicos y correctores ortográficos de gran calidad y cobertura para un buen número de idiomas, a partir de los ya desarrollados utilizando este tipo de herramientas. Además, se permitirían definir reglas de corrección para errores de competencia léxica o reglas para errores en OCR, que harían la corrección más precisa.

Agradecimientos

El trabajo descrito en este artículo ha sido parcialmente subvencionado por Departamento de Política Lingüística del Gobierno Vasco y por el Ministerio de Industria dentro del programa PROFIT (FIT-350401-2006-5).

6. Referencias

1. Alegria I., Artola X., Sarasola K., Urkia M. Automatic morphological analysis of Basque. *Literary & Linguistic Computing* Vol. 11, No. 4, 193-203. Oxford University Press. Oxford. (1996)
2. Alegria I. Morfología de estados finitos Procesamiento del Lenguaje Natural (SEPLN) Revista no. 18, 1-26. Donostia (1996)
3. Alegria I., Díaz de Ilarraza A., Labaka G., Lersundi M., Mayor A., Sarasola K. Transfer-based MT from Spanish into Basque: reusability, standardization and open source. LNCS 4394. *Cycling 2007*. ISBN-10: 3-540-70938-X (2007)
4. Aldezabal I., Alegria I., Ansa O., Arriola J., Ezeiza N. Designing spelling correctors for inflected languages using lexical transducers. *EACL'99* (1999)
5. Armentano-Oller C., Corbí-Bellot A., Forcada M., Ginestí-Rosell M., Bonev B., Ortiz-Rojas S., Pérez-Ortiz J., Ramírez-Sánchez G., Sánchez-Martínez F., "An open-source shallow-transfer machine translation toolbox: consequences of its release and availability", in *Proceedings of OSMaTran: Open-Source Machine Translation, A workshop at Machine Translation Summit X* (2005),
6. Beesley K.R. and Karttunen L. . *Finite State Morphology*. CSLI Publications, Palo Alto, CA. (2003)
7. Nemeth V., Tron P., Halacsy A., Kornai A., Rung I. Leveraging the open source ispell codebase for minority language analysis. *Proceedings of SALT MIL* (2004)

HIGEIA, la interoperabilidad sanitaria

R. Manjavacas¹, J.P. de Ruz¹, A. Rodriguez¹

¹ Área de Tecnologías de la Información, Servicio de Salud de Castilla – La Mancha
{rmanjavacas,jruzo, arodriguez}@sescam.jccm.es

Resumen. Los entornos sanitarios se caracterizan actualmente por presentar una gran cantidad de sistemas de información heterogéneos, monolíticos y aislados. La creciente necesidad de integración de estos sistemas ha provocado la implantación de tecnología EAI (Enterprise Application Integration). Desde el Área de Tecnologías de la Información del Servicio de Salud de Castilla – La Mancha se ha impulsado la adopción de este tipo de tecnologías dentro de su línea estratégica de Software Libre, adoptando como punto de partida tras previo estudio de mercado la solución BIE-GPL, puesto que éste satisfacía perfectamente las necesidades iniciales. Esta solución ha sido evolucionada, mejorada y adaptada a este entorno sanitario para poder ser implantado en los sistemas críticos de producción actuales. Como resultado ha nacido una nueva rama del proyecto bautizada con el nombre de HIGEIA. Su empleo en el proyecto de integración de Atención Primaria con los laboratorios de las Unidades Hospitalarias ha sido pieza clave tecnológica.

Palabras clave: Sescam, Software Libre, Higeia, EAI, integración de sistemas.

1. Introducción

El Servicio de Salud de Castilla – La Mancha (SESCAM) nace el 1 Enero del año 2002 como fruto de la transferencia de las competencias en materia de sanidad a la comunidad autónoma de Castilla – La Mancha. Desde su consolidación el Sescam comenzó a diseñar un proyecto dirigido a propiciar un cambio importante en la sociedad. Durante su primer año se creó la infraestructura tecnológica y el modelo de informática del Sescam, un modelo corporativo basado en la creación de un Área de Tecnologías de la Información (Área de TIC), así como de un modelo informático para la Historia Clínica Electrónica (HCE). Este área ha seguido manteniendo una línea estratégica fundamentada en la implantación de soluciones de código abierto desde su creación. Dentro de esta línea de trabajo, diversos proyectos han sido materializados hasta la actualidad contribuyendo cada uno de ellos en la consecución de una historia clínica electrónica unificada e integrada.

El proyecto Esculapio se configuró como el primer proyecto clave en la aplicación de tecnología de código abierto dentro de este entorno sanitario. Iniciado en el año 2003, este proyecto tenía por objetivo la informatización de los centros de salud de Castilla – La Mancha acercando la administración sanitaria al ciudadano a nivel de

atención primaria, es decir, propicia que los sistemas de información sanitarios sean accesibles por los profesionales desde cualquier lugar de la región, “que viaje la información, no el ciudadano”. La infraestructura software desplegada sobre la que descansa el sistema de información de Atención Primaria se fundamenta entre otros en Linux (Red Hat 8.0) como sistema operativo para servidores, Samba como servidor de archivos, Amanda para la gestión de copias de seguridad, Squid como proxy-caché, Webmin para administración remota y OpenOffice como paquete ofimático en las estaciones cliente. Actualmente este proyecto ha llegado a más de 233 centros sanitarios de Atención Primaria entre centros de salud y consultorios.

Posteriormente, el proyecto de migración del sistema de información hospitalaria HP-HIS de su plataforma nativa HP-UX a otra bajo el entorno Linux (Red Hat Enterprise 3.0 AS) ha sido otro de los hitos en la aplicación de tecnología de código abierto en este caso a nivel de Atención Especializada. Por una lado, esta migración supone solamente cambios en la arquitectura de los servidores del hospital y por otro lado, proporciona principalmente una notable mejora de rendimiento del aplicativo ya que el ofrecido por éste en su configuración inicial no era el adecuado. Un ejemplo del aumento del rendimiento está en los tiempos de proceso de cara al usuario y paciente que está en ventanilla del hospital, la consulta de cita más costosa que tardaba entorno a unos 15 minutos se ha reducido prácticamente a 2 segundos. Actualmente esta migración se ha llevado a cabo en casi toda la totalidad de complejos hospitalarios.

En la actualidad, en el camino de mejora continua de la calidad asistencial sanitaria y consecución de una historia clínica electrónica unificada e integrada se ha hecho patente la necesidad de integración de los diferentes sistemas de información que componen el entorno sanitario, debido por un lado a que actualmente éstos se estructuran como sistemas heterogéneos, monolíticos y aislados y por otro lado, a que cada vez es mayor el número de tareas cotidianas que requieren la interconexión de dichos sistemas de información tanto a nivel de datos como de proceso. Como fruto de esta necesidad surgió el proyecto HIGEIA, el cual, ha supuesto un salto cualitativo en la aplicación de tecnología libre, puesto que es el primer proyecto de este entorno sanitario que conlleva una contribución manifiesta a la comunidad. En los siguientes puntos de la presente comunicación se desarrollan en primer lugar, los aspectos más relevantes del proyecto HIGEIA tales como situación de partida y solución adoptada, y a continuación se describe el primer marco real de aplicación donde este proyecto entró a formar parte junto con su papel desempeñado.

2. HIGEIA

Los entornos sanitarios, al igual que otros tipos de entornos corporativos, presentan un amplio abanico de sistemas de información heterogéneos en sus diferentes niveles para llevar a cabo sus tareas cotidianas. Normalmente, estos sistemas no han sido desarrollados de manera coordinada bajo un marco global convirtiéndose en sistemas

autónomos y aislados, los cuales, no comparten ni datos ni procesos [1]. Este tipo de configuración conlleva que la información clínica electrónica de un paciente esté dispersa, duplicada e incluso no relacionada, lo cual, supone un gran obstáculo para conseguir que la historia clínica electrónica de un paciente se encuentre unificada e integrada. En la actualidad, dichos sistemas de información tienden a aumentar su nivel de cohesión puesto que, cada vez es mayor el número de tareas cotidianas que requieren su interconexión tanto a nivel de datos como de proceso. Dentro de este marco donde las necesidades de integración entre sistemas de información son patentes es donde las “tecnologías de integración” juegan su papel principal [2] . Este tipo de tecnologías se aglutinan básicamente sobre el término de EAI (Enterprise Application Integration) [3]. Los EAI's son sistemas “middleware” que aglutinan las funcionalidades necesarias en un marco de integración de sistemas de información, siendo éstas: *transporte de datos* adoptando los diferentes estándares tanto a nivel de protocolo de comunicación (p. ej.: http, ftp, webservices, sockets, ...) como a nivel de encapsulado de los datos (p. ej.: HL7), *transformación de datos* con el objetivo de adecuarse a las interfaces de los diferentes sistemas y *encaminamiento de datos* ofreciendo mecanismos que permitan definir la lógica necesaria para su tratamiento. La aplicación de la tecnología EAI se convierte en una pieza clave para flexibilizar la integración de los sistemas de información existentes en sus diferentes niveles en organizaciones tales como las sanitarias [4], respetando principalmente su configuración, lo cual, se traduce en un ahorro global de los costes de integración y mantenimiento. La implantación de los EAI en entornos sanitarios se podría llevar a cabo en los siguientes niveles:

- Nivel global del entorno: donde se integran los sistemas de las unidades hospitalarias y de Atención Primaria entre sí (figura 1).
- Nivel de Unidad Hospitalaria: donde los sistemas HIS (Hospital Information System) y aplicativos departamentales de Atención Especializada son interconectados (figura 2).



Fig. 1: Nivel global del entorno sanitario

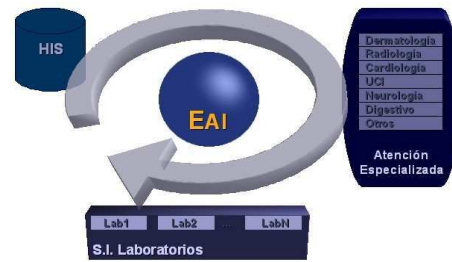


Fig. 2: Nivel unidad hospitalaria

2.1 Estudio de alternativas EAI

Basándose en estudios de mercado realizados por entidades especializadas en el sector de las Tecnologías de la Información (TIC)[5], se han valorado varias propuestas de EAI's existentes para estimar su adecuación al entorno sanitario del SESCAM. Entre estos podemos citar los siguientes: Orion Rhapsody [6], BIE-GPL [7], Fiorano Business Integration Suite [8] y Cast Iron Application Router 1000 [9]. La evaluación y análisis de impacto de la implantación de cada una de estas alternativas se realizó atendiendo a una serie de requisitos propuestos desde el Área de Tecnologías de la Información del SESCAM. Entre dichos requisitos destacan:

- Múltiples protocolos de comunicación (p. ej.: HTTP, FTP, Sockets TCP, WebServices, SMTP)
- Capacidad de encaminamiento de datos
- Multiplataforma
- Basado en el modelo de desarrollo y mantenimiento de software de código abierto (Open Source)
- Soporte de estándares propuestos por IHE e HIPAA (p. ej.: HL7)
- Soporte de múltiples formatos de mensajes (p. ej.: EDI, XML, TAB, CSV, HL7)
- Conectividad con bases de datos corporativas (p. ej.: OracleDB, Informix, PostgreSQL, Microsoft SQL Server, MySQL)

El requisito de valorar soluciones basadas en el modelo de “código abierto” se encuadra dentro de la línea impulsada desde el Área de Tecnologías de la Información por la implantación de este tipo soluciones. En este caso, la posibilidad legal de modificar y adaptar el software a las características de la organización es quizás la ventaja más destacable de entre todas las ventajas que presenta este modelo, puesto que, debido a las características del entorno sanitario de Castilla La-Mancha se precisa de un EAI extensible, con alta capacidad de adaptación para facilitar la integración de los numerosos sistemas de información existentes y que pueda ser implantado dentro de los sistemas de producción actuales. Este último aspecto es muy importante, ya que estos sistemas además de estar certificados por las entidades que dan soporte, cumplen la normativa vigente de la LOPD (Ley Orgánica de Protección de Datos de carácter personal) y disponen de los mecanismos necesarios para ofrecer servicios de alta disponibilidad, características fundamentales en un entorno tan crítico como el sanitario.

Como resultado del estudio se concluyó que el EAI BIE-GPL se presentaba como el EAI que mejor cumplía con las necesidades planteadas por el SESCAM. Además de los requisitos anteriormente comentados, el EAI BIE-GPL ofrece otras características que resultan de especial interés. Éstas son:

- Una herramienta de desarrollo de diagramas de flujo de procesamiento basada en el estándar BPML (Business Process Management Language).

- Monitorización de procesos a nivel de actividad
- Motor de procesamiento de flujos BPM basado en una arquitectura extensible basada en el concepto de “plug-ins”
- Administración y edición de diagramas de flujos BPM basada en Web
- Herramienta gráfica que permite diseñar transformaciones de modelos de datos en XML de forma fácil e intuitiva.
- Basado en un sistema de mensajería JMS (Java Messaging Service) que aporta una infraestructura fiable y robusta de gestión de mensajes.

A continuación se expone la evolución que desde el SESCOAM se ha llevado a cabo sobre el EAI BIE-GPL para adaptarlo a su entorno de producción así como a otros requerimientos funcionales necesarios.

2.2 Higeia, resultado de una evolución

La elección del EAI BIE-GPL tras el previo estudio de mercado realizado supuso el punto de partida en la apuesta de poseer un EAI propio y adaptado al entorno del SESCOAM. El EAI BIE-GPL es un software de integración multiplataforma desarrollado bajo el estándar J2EE. La versión disponible en la red, además de ofrecer las características deseables para cualquier tipo de EAI, presenta otras características adicionales las cuales lo convierten en una solución inicial muy atractiva tal y como se mostró en el punto anterior. A pesar de ello, fue necesario llevar a cabo un proceso de evolución, mejora y adaptación al entorno sanitario del SESCOAM. Esta adaptación ha sido realizada en el Centro en Investigación Sanitaria en “Open Source” (CISOS) perteneciente a la Unidad de I+D+i del Área de Tecnologías de la Información. Este grupo de trabajo fundado en el año 2005 tiene por objetivo incentivar el uso e implantación de tecnologías y proyectos basados en Software Libre en el ámbito sanitario del Servicio de Salud de Castilla – La Mancha. Como fruto de este trabajo ha nacido una nueva rama del proyecto bautizada con el nombre de “HIGEIA”.

El proceso de evolución y adaptación seguido ha sido enfocado principalmente en las dos líneas de actuación siguientes: a nivel arquitectural y a nivel funcional. A nivel arquitectural, la última versión 6.0.5 de BIE-GPL disponible en la red se caracteriza porque se presenta como un producto, el cual, lleva integrados una serie de componentes que proporcionan los servicios que precisa el propio motor de integración. Estos componentes embebidos son: un servidor de aplicaciones J2EE (JBoss 3.2.3), un sistema gestor de bases de datos relacional (HSQLDB 1.7.1) y un sistema gestor de bases de datos XML (eXist 0.9.2). Este modo de brindar el producto presenta sus ventajas y sus inconvenientes. En este sentido, su ventaja principal es la facilidad que ofrece para su distribución, instalación y configuración bajo cualquier plataforma pero por el contrario, esto supone ligarse a unas tecnologías en concreto para cada uno de los servicios comentados anteriormente. Este último hecho puede provocar la aparición de una serie de conflictos previos a su implantación dentro de una organización, puesto que ésta puede presentar otras

tecnologías alternativas ya implantadas para cubrir los servicios requeridos por BIE y evitar la tendencia a incurrir en un maremágnum de tecnologías para servicios similares. En concreto, la situación del SESCAM se ajustaba a ésta última y se decidió por adaptar BIE-GPL para eliminar este tipo de ligaduras tecnológicas. En esta línea, se ha avanzado hacia un software de integración que pueda ser fácilmente configurable sobre cualquier sistema gestor de base de datos y desplegable bajo cualquier servidor de aplicaciones J2EE (figura 3). Para ello, se han realizado diferentes esfuerzos de desarrollo que han implicado cambios en la capa de acceso a datos, en el servicio de colas de mensajes empleado (incluyendo el empleo de JORAM como servicio de mensajería de código abierto para Java), en la forma de acceder a los servicios externos al propio motor, etcétera. Estos esfuerzos han proporcionado de manera implícita la posibilidad de incorporación de este EAI, bajo la configuración de alta disponibilidad existente en los servicios de informática de este entorno sanitario.

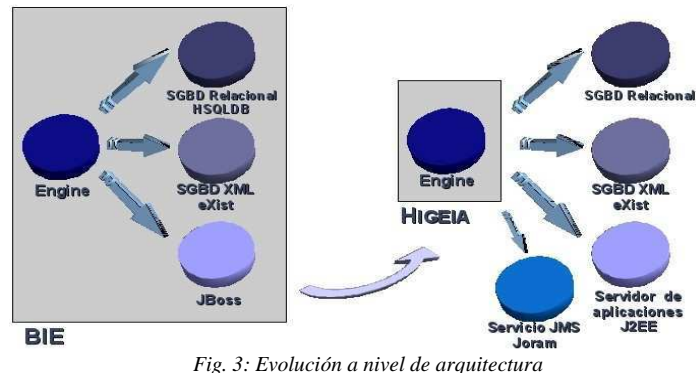


Fig. 3: Evolución a nivel de arquitectura

A nivel funcional, los esfuerzos realizados se han enfocado principalmente en aumentar el número de funcionalidades disponibles tanto del propio motor de integración como del propio editor de flujos BPM. En este último caso, se han incorporado las siguientes funcionalidades entre otras: capacidad de validación de documentos en formato XML frente a esquemas XSD, comunicación con otros sistemas mediante tecnología “socket”, gestión de codificación de caracteres, mayor soporte del estándar de intercambio, gestión e integración de información electrónica sanitaria HL7 [10], etcétera.

En líneas generales, a parte de las mejoras y adaptaciones comentadas anteriormente se han detectado y solucionado diferentes “errores” y se ha logrado mejorar el rendimiento de la propia plataforma de integración, traduciéndose directamente en una reducción considerablemente de los tiempos de ejecución de cada flujo de procesamiento BPM (ejemplo de flujo BPM figura 4).

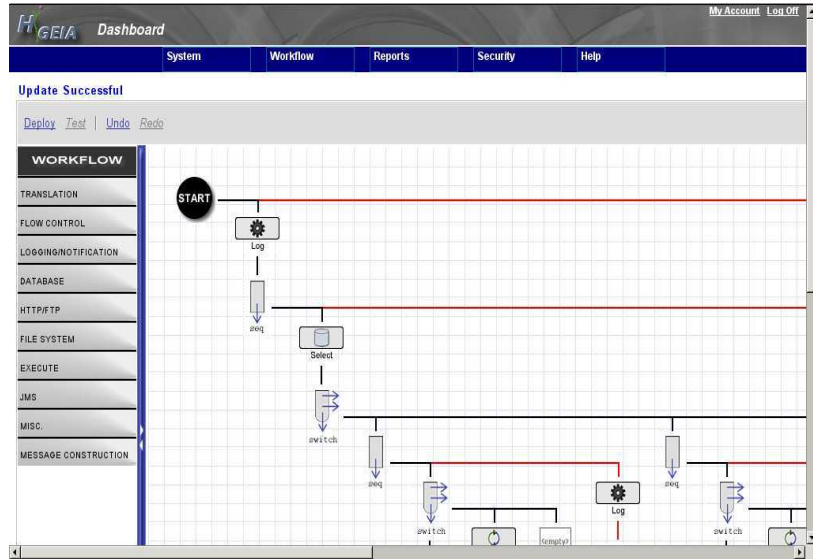


Fig. 4: Diseño de un flujo de negocio

3. HIGEIA, clave en la integración de Atención Primaria-Laboratorios

Entre los diferentes sistemas de información existentes dentro del Sescam se encuentra el sistema de información de Atención Primaria conocido como “Turriano”. Este sistema tiene por objetivo la gestión a nivel regional de la Historia Clínica Electrónica (HCE) de un paciente a nivel de atención primaria. Una de las demandas de mejora existentes era la conexión automática con los sistemas de laboratorios de las unidades hospitalarias con el objetivo de mejorar el circuito de petición y recepción de solicitudes de análisis clínicos. Este hecho se convirtió en el germen del proyecto de integración entre los sistemas de información de Atención y Laboratorios, el cual, supone un salto cualitativo en la gestión de la HCE.

Este proyecto de ámbito regional conllevó afrontar una serie de dificultades iniciales entre las que se encontraban: identificación única del paciente en todas las entidades, número y diversidad de laboratorios, definición de un catálogo centralizado de pruebas, mapeo de dicho catálogo centralizado con los catálogos existentes en los diferentes laboratorios, ... Pero con la consecución del mismo se han conseguido entre otros los siguientes beneficios a nivel de atención primaria: agilización del circuito de petición y recepción de solicitudes de análisis clínicos, incorporación automática de los resultados a la HCE del paciente, consolidación de un catálogo centralizado y único de pruebas de análisis clínicos, eliminación de posibles errores de las peticiones en el modo tradicional incluido su extravío e instaurar el CIP (Código de Identificación de Paciente) como único identificador válido de los pacientes.

La elección de HIGEIA como EAI dentro de este proyecto de integración entre los sistemas de información de Atención Primaria y Laboratorios se ha convertido en el eje central del proyecto y en pieza clave y fundamental a nivel tecnológico para el éxito del mismo. Entre las aportaciones más relevantes que ha supuesto la aplicación de este EAI se encuentran las siguientes: flexibilizar la comunicación entre extremos, validación de mensajería, enrutamiento de mensajería, aseguramiento de la entrega y recepción de mensajería, histórico de mensajería y por último garantiza la mantenibilidad y escabilidad de la solución con la posibilidad de incorporar fácilmente por ejemplo nuevos sistemas de información de laboratorio o modificaciones sin que ello implique la modificación de la lógica de negocio en el resto de extremos.

3.1. Modelo de integración

El modelo de integración planteado para este proyecto de integración se fundamenta en una arquitectura centralizada en la cual, HIGEIA se muestra como eje central de la misma (figura 5). Desde un punto de vista funcional tanto Turriano como los sistemas de información de los laboratorios se responsabilizan únicamente de implementar la interfaz de comunicación con el EAI y del envío y recepción de solicitudes o de resultados junto con su gestión asociada según corresponda. Desde un punto de vista técnico, la implantación y mantenimiento de HIGEIA se ha llevado a cabo en una única instancia ubicada en los Servicios Centrales del Sescam a la que deben conectarse por un lado los diferentes sistemas de información de los laboratorios ubicados en las unidades hospitalarias y por otro lado, Turriano, el cual, debido a su carácter regional entre otras muchas razones se encuentra de igual modo desplegado en los Servicios Centrales.



Fig. 5: Modelo de integración

Uno de los objetivos perseguidos en este modelo de integración ha sido la aplicación de estándares por las ventajas que este hecho supone. Como ejemplo más destacable, cabe resaltar la adopción del estándar HL7[10] en su versión 2.5, como estándar para el intercambio, gestión e integración de información electrónica sanitaria, con objeto de ajustarse a las especificaciones definidas por IHE (Integrating the Healthcare Enterprise) [11] para laboratorios.

3.2 Experiencias piloto

El hecho de que este proyecto de integración sea por un lado un proyecto de alcance regional que engloba a las diferentes gerencias de Atención Primaria y Especializada y por otro, un proyecto en el que los sistemas de información de los diferentes Laboratorios pueden diferir entre gerencias de Atención Especializada o incluso dentro de la misma gerencia dependiendo de su especialidad, ha conducido a la definición de una serie de directrices para abordar la implantación.

Estas directrices se fundamentan básicamente en la definición de un pilotaje atendiendo a cada uno de los sistemas de información de Laboratorio y a la gerencia de Atención Primaria y de Especializada involucradas en cada uno de ellos. Cada piloto diferenciado es supervisado y coordinado por la gerencia de especializada y de primaria afectada. En primera instancia, en cada piloto se selecciona un centro de salud de referencia con un volumen reducido de extracciones a la semana con objeto de poner en marcha el piloto. Cuando el funcionamiento del mismo es correcto tras varias semanas de pruebas este se amplía con un centro de salud con un volumen mayor de extracciones. Cuando un piloto funciona sin incidencias con un cierto volumen de extracciones se procederá a su puesta en producción. Durante todo este tiempo de pilotaje se mantiene un paralelismo entre el proceso tradicional en papel y el electrónico con el fin de que el personal tanto facultativo como técnico de laboratorio certifiquen el funcionamiento correcto. Actualmente se encuentran definidas tres experiencias piloto asociadas cada una de ellas a la gerencias de Guadalajara, Alcázar de San Juan y Puertollano.

4. Conclusiones

El servicio de Salud de Castilla – La Mancha desde su Área de Tecnologías de la Información mantiene desde su creación en el año 2002 una línea estratégica fundamentada en la implantación de soluciones bajo tecnología libre, ejemplo de ello resultan ser proyectos tales como “Esculapio” o la migración del HP-HIS de HP-UX a Linux. En la actualidad, la creciente necesidad de integración de los diferentes sistemas de información que componen el entorno sanitario ha conducido a la realidad del proyecto HIGEIA, el cual, ha supuesto un salto cualitativo en la aplicación de tecnología de código abierto, puesto que es el primer proyecto de este entorno sanitario que conlleva una contribución manifiesta a la comunidad. HIGEIA, es un EAI libre fruto de un proceso de evolución (tanto a nivel arquitectural como funcional), mejora y adaptación del EAI BIE-GPL al entorno sanitario del SESCAM con el objetivo de poder ser implantado en los sistemas críticos de producción

actuales. La implantación exitosa del mismo dentro del proyecto de integración de Atención Primaria con los diferentes laboratorios de las Unidades Hospitalarias, lo ha consolidado y convertido en una pieza clave tecnológica.

5. Referencias

1. A. Laroia, *Healthcare's Digital Heartbeat*, EAI Journal. Agosto 2002
2. P. Toussaint, A. Bakker, L. Groenewegen. *Integration of Information Systems: Assessing its Quality*. Computer Methods and Programs in Biomedicine, vol 64, pp. 9-35 (2001).
3. D. S. Linthicum, *Enterprise Application Integration*. Addison-Wesley (1999).
4. K. Khoubati, M. Themistocleous, Z. Irani, *Integration Technology Adoption in Healthcare Organisations: A Case for Enterprise Application Integration*. Proceedings of the 38th Hawaii International Conference on System Sciences (2005).
5. D. Macvittie, *Keeping Integration Simple*, RealWorld Labs (2004).
6. Orion rhapsody. Disponible en la url: <http://www.orionhealth.com/rhapsody/index.htm>
7. BIE - GPL(Business Integration Engine), disponible en la url: <http://sourceforge.net/projects/bie-gpl>
8. Fiorano, disponible en la url: <http://www.fiorano.com/products/fesb/fioranobis.htm>
9. Cast Iron Application Router, disponible en la url: <http://www.castironsys.com/products.shtml>
10. Health Level Seven (HL7), disponible en la url: <http://www.hl7spain.org>
11. Integrating the Healthcare Enterprise (IHE), disponible en la url: <http://www.ihe.org/>

Código fuente de Linux en la docencia de Sistemas Operativos I

José Antonio Gómez¹,
Buenaventura Clares¹

¹ Dpto. De Lenguajes y Sistemas Informáticos, Universidad de Granada
{jagomez,
bclares}@ugr.es

Abstract. El presente artículo pretende ilustrar como un sistema operativo de fuentes abiertas como Linux puede usarse con éxito en la docencia de una asignatura de fundamentos de sistemas operativos. Su uso tiene una doble ventaja: de un lado, se ilustran los conceptos básicos de la asignatura de Sistemas Operativos I a través de implementación de un sistema real, lo cual es muy motivador; de otro, se utilizan los ejemplos de este sistema para introducir al alumno al estudio del software de fuentes abiertas que le sea útil para otras asignaturas y en su carrera profesional. Se desarrollan algunos de los ejemplos usados en la asignatura y se comentan los resultados obtenidos.

Keywords: sistemas operativos, docencia, Linux.

1. Introduction

Conocida es la importancia del uso de código de un sistema operativo para la docencia de tal disciplina. Para ello, se han desarrollado proyectos como Nachos [1] o Minix [2]. En el prefacio de su libro [3], G. Nutt reconoce “There are only a few widely used commercial operating systems. While studying these systems is valuable, there are practical barriers to experimenting with any of them in the classroom. First, commercial operating systems are very complex since they must offer full support to commercial applications. It is impractical to experiment with such complex software because it is sometimes difficult to see how specific issues are addressed within the software”

Si bien estos sistemas operativos docentes de “juete” antes mencionados cumplen su objetivo de cada a la enseñanza, tiene el inconveniente de no motiva a muchos estudiantes que los consideran poco interesantes para su futuro académico y profesional [4]. Nuestra experiencia docente en la que se han explicado diferentes sistemas de este tipo, nos indica que estos sistemas son vistos como poco útiles por un amplio abanico de estudiantes.

Desde esta visión, crece la tendencia en las asignaturas de sistemas operativos de apoyarse en los sistemas operativos reales de fuentes abiertas, en concreto Linux, para cubrir sus objetivos [5].

En nuestra universidad, los temas relacionados con sistemas operativos se imparten en tres asignaturas: Sistemas Operativos I, Sistemas Operativos II, y Diseño de Sistemas Operativos. La primera troncal, la segunda obligatoria de universidad, y la tercera optativa. En Sistemas Operativos II como en Diseño de Sistemas Operativos se emplea Linux para la docencia tanto teórica como práctica. La enseñanza de Linux es similar a la seguida en otras asignaturas, tanto en España como en el resto de mundo, donde a nivel teórico se estudia la estructura de Linux en general y los componentes en particular, y a nivel práctico se implementan módulos de carga dinámica, se reimplementan algunas llamadas al sistema, etc.

En teoría de Sistemas Operativos I se estudian los conceptos básicos de los sistemas operativos siguiendo los libros de texto clásicos de la asignatura [3,6,7]. En esta asignatura tiene un número importante de conceptos y abstracciones realizadas en los sistemas operativos actuales que se ven desde un punto de vista teórico, pero que para los estudiantes son a veces difíciles de visualizar por lo abstractos que suelen ser. Para que los comprendan de manera efectiva es necesario recurrir al estudio de implementaciones reales de código, ya que el código rellena el vacío entre la abstracción teórica y la implementación.

El procedimiento general que se utiliza en la presentación del código es el siguiente: primero, se explican los conceptos de la misma forma que en otros cursos; segundo, se prepara una transparencia con el mismo ejemplo sobre el que superpone el código de Linux; por último, se explica utilizando un navegador de código, por ejemplo, nosotros solemos usar <http://lxr.linux.no>.

El primer paso es necesario pues deseamos partir una visión general e independiente del sistema operativo del concepto que queremos estudiar. Aquí no deseamos que los detalles técnicos y de implementación empañen el concepto.

En un segundo pretende ver de forma conjunta el ejemplo teórico y la implementación en Linux. En este paso a partir de un Esquema como el de la Figura 1, se superponen extractos del código que pretendemos ilustrar. Esto permite al estudiante ubicar el código en el esquema general del sistema, y seguir los detalles de la posterior navegación por el código sin perderse.

El tercer paso, que explica con más detalle el código, esta también enfocado a ver el código real y a familiarizar al alumno en la navegación a través de código. De esta forma se pretende fomentar la curiosidad del alumno y que el mismo se anime a visitar y comprender estos y otros fragmentos de código.

2. Ejemplos

A continuación, vamos a mostrar algunos de los ejemplos utilizados en la asignatura para ilustrar la implementación de diferentes conceptos de distinto nivel y complejidad. La presentación de los ejemplos se hace siguiendo el orden de presentación de los diferentes elementos a los alumnos.

En ellos, dado que se pretende ilustrar un concepto específico, se procura resaltar aquellos aspectos relacionados con el concepto en cuestión sin fijarse demasiado en otros detalles que siendo también interesantes no aportan nada a la discusión en curso.

Los ejemplos elegidos muestran el mecanismo de realización de una llamada al sistema, la implementación de tiempo compartido, y la estructura de una operación de entrada/salida de teclado.

2.1. Mecanismo para realiza las llamadas al sistema

Un ejemplo típico de concepto visto en teoría que no tiene una contrapartida práctica es el mecanismo para realizar una llamada al sistema en una arquitectura con funcionamiento dual del procesador.

En este punto, partimos de la explicación de como se realiza una llamada al sistema tal y como podemos encontrar en cualquier libro de texto. A partir de aquí, se pretende ilustrar cada uno de los pasos del mecanismo, empezando por los ajustes en modo kernel y siguiendo con la invocación de la llamada desde espacio de usuario. Comenzamos viendo como primero el sistema operativo asigna el vector *0x80* al manejador de llamadas al sistema, que podemos encontrar en el archivo fuente *linux/include/asm/hw_irq.h*:

```
#define SYSCALL_VECTOR 0x80
```

a continuación vemos como se asigna a este vector el manejador de llamadas al sistema, tal como aparece en el archivo *linux/arch/386/traps.c*

```
set_system_gate (SYSCALL_VECTOR, &system_call);
```

donde la función *set_system_gate* copia en la entrada de la IDT (Interrupt Descriptor Table) el correspondiente manejador de la llamada:

```
static void __init set_system_gate(unsigned int n, void
                                *addr)
```

```
{
    _set_gate(idt_table+n,15,3,addr,__KERNEL_CS);
```

```
static inline void _set_gate(void *adr, unsigned type,
unsigned long func, unsigned dpl, unsigned ist)
```

```

{
    struct gate_struct s;
    s.offset_low = PTR_LOW(func);
    s.segment = __KERNEL_CS;
    s.ist = ist;
    s.p = 1;
    s.dpl = dpl;
    s.zero0 = 0;
    s.zero1 = 0;
    s.type = type;
    s.offset_middle = PTR_MIDDLE(func);
    s.offset_high = PTR_HIGH(func);
    /* does not need to be atomic because it is only
done once at setup time */
    memcpy(adr, &s, 16);
}

```

En el archivo *linux/arch/i386/entry.S* encontramos los puntos de acceso al kernel marcados con la declaración *ENTRY()*. En nuestro caso nos interesa la entrada (en el ejemplo del apartado 2.2, veremos el punto de salida) al kernel por la realización de una llamada al sistema, como muestra el código:

ENTRY(system_call)

```

    pushl %eax                # save orig_eax
    SAVE_ALL
    GET_CURRENT(%ebx)
    testb $0x02,tsk_ptrace(%ebx) # PT_TRACESYS
    jne tracesys
    cmpl $(NR_syscalls),%eax
    jae badsys
    call *SYMBOL_NAME(sys_call_table)(,%eax,4)
    movl %eax,EAX(%esp)      # save the return value

```

donde lo primero que hacemos es *SAVE_ALL* que salva los registros del procesador para luego retornar a modo usuario con *RESTORE_ALL*. La invocación de la llamada al sistema específica que estamos realizando se realiza en *call *SYMBOL_NAME(sys_call_table)(,%eax,4)* donde *%eax* contiene el identificador de la llamada que realizamos y se utiliza como índice en la tabla de llamada al sistema que es simplemente una matriz de punteros a función.

Este archivo nos permite además ilustrar todos los puntos de entrada al kernel como por ejemplo por interrupciones, excepciones, etc.

Todo lo visto hasta ahora se realiza en el kernel, en la parte de usuario lo que debemos hacer es cargar los registros del procesador con el número de llamada que deseamos invocar y los argumentos de esta, e invocamos a la interrupción 0x80. Les mostramos como pueden hacerlo directamente, o bien, en la forma usual, a través de una función de la biblioteca.

Para la invocación directa solo es necesario realizar un programa en ensamblador. El programa siguiente muestra un fragmento del programa “Hola, Mundo”:

```
.LC0:
    .string "hola, mundo\n"
    . . .
main: movl $len,%edx
      movl $.LC0, %ecx
      movl $1, %ebx
      movl $4, %eax
      int $0x80
    . . .
```

La invocación de la llamada al sistema desde una función de la biblioteca es algo más difícil de ver pues el código fuente de la biblioteca es independiente de la arquitectura. Por ello, para ilustrarlo, compilamos el programa y los analizamos con `objdump` que nos permite desensamblar el código generado, concretamente:

```
% gcc -o syscall syscall.c -static
% objdump -D syscall
. . .
08060e20 <__libc_write>:
8060e20: 53                               push %ebx
8060e21: 8b 54 24 10                      mov 0x10(%esp,1),%edx
8060e25: 8b 4c 24 0c                      mov 0xc(%esp,1),%ecx
8060e29: 8b 5c 24 08                      mov 0x8(%esp,1),%ebx
8060e2d: b8 04 00 00 00                  mov $0x4,%eax
8060e32: cd 80                            int $0x80
8060e34: 5b                               pop %ebx
    . . .
```

2.2. Implementación de tiempo compartido

Este segundo ejemplo parte de la implementación de un sistema operativo multiprogramado visto en el segundo tema de teoría. A partir de él, podemos ver como implementar un sistema de tiempo compartido. Para hacerlo, se generan interrupciones periódicas para quitar el control al proceso en ejecución y dárselo a otro proceso preparado para ejecutarse. Aquí es donde entra en juego el manejador de la interrupción, que lleva la cuenta del tiempo consumido por el proceso actual y le quita el control cuando supera el límite establecido.

Esto se muestra en la rutina de interrupción de reloj (archivo `linux/kernel/timer.c`), que se muestra a continuación para la versión 2.4 del kernel:

```
void update_process_times(int user_tick)
{
    struct task_struct *p = current;
    int cpu = smp_processor_id(), system = user_tick ^ 1;
    update_one_process(p, user_tick, system, cpu);
    if (p->pid) {
        if (--p->counter <= 0) {
            p->counter = 0;
            /* SCHED_FIFO is priority preemption, so this is
```

```

* not the place to decide whether to reschedule a
* SCHED_FIFO task or not - Bhavesh Davda
*/
        if (p->policy != SCHED_FIFO) {
            p->need_resched = 1;

```

Vemos como es necesario replanificar (*need_resched=1*) cuando el proceso ha consumido la cantidad de tiempo asignada (*--p->counter<=0*).

Este ejemplo nos permite ilustrar otro aspecto más complejo de entender relacionado con la no-apropiatividad del kernel 2.4. Si analizamos el fragmento de código anterior, hay algo que inicialmente no nos concuerda con lo que comentábamos antes sobre cambiar de proceso cuando se produce una interrupción. En el código anterior, cuando el tiempo asignado al proceso ha concluido deberíamos invocar al planificador para cambiar de proceso, y en lugar de ello ponemos el indicador *need_resched=1*.

En este punto, planteamos a nuestros estudiantes la siguiente cuestión ¿cómo es posible realizar el cambio de proceso, que es necesario realiza en modo kernel, cuando el kernel es no apropiativo? La respuesta es fácil cuando nos damos cuenta que la pregunta tiene una pequeña trampa: cuando decimos que un kernel es no-apropiativo, queremos decir que lo es en general menos en un punto donde si lo es, es decir, no apropiio un proceso en modo kernel salvo el instante en que esta a punto de retornar a modo kernel, en cuyo caso dado que las estructuras del kernel estan en un modo seguro es posible apropiar al proceso actual. Esto lo podemos ver en punto de retorno de modo kernel a usuario del archivo *entry.S* que citamos anteriormente. Como se muestra, si en algun instante se ha activado en indicar *need_resched*, se invoca al planificador en lugar de retornar a modo usuario:

```

ENTRY(ret_from_sys_call)
    cli          # need_resched and signals atomic test
    cmpl $0,need_resched(%ebx)
    jne reschedule
    cmpl $0,sigpending(%ebx)
    jne signal_return
restore_all:
    RESTORE_ALL

```

2.3. Entrada/salida de teclado

Este ejemplo es más amplio porque abarca diferentes componenes y archivos de Linux. Antes de ver el código, los alumnos guiados por el profesor ha construido un esquema como el que aparece en la Figura 1 tras realizar varios ejercicios. En él, se puede ver como un proceso que realiza una operación de lectura de teclado, invico a la llamada al sistema *read*. Esto transfiere el control al sistema operativo que ejecuta el código de la llamada *sys_read()* que bloqueará al proceso si no hay caracteres que leer del búfer. Por otro lado, la rutina de interrupción de teclado en la encargada de

leer la tecla pulsada, transformarla en el correspondiente carácter, insertarla en el búfer de teclado, y despertar a los procesos que esperan este evento.

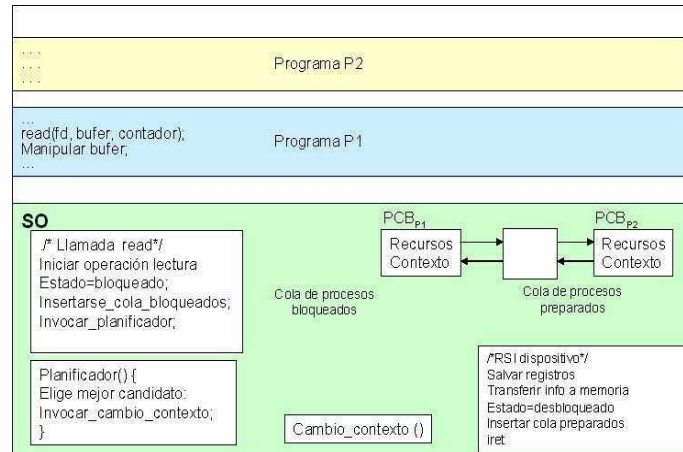


Figura 1.- Ejemplo genérico de una operación de entrada/salida

Una vez estudiado el ejemplo a nivel genérico, procedemos a verlo en el caso de Linux. Para ello, mostramos un fragmento del archivo `sys_read()` que contiene el código de la llamada al sistema para leer de un archivo. Dado el sistema de independencia del dispositivo utilizado por Linux, esta función se instancia en la función `tty_read()` del archivo `linux/drivers/char/tty_io.c`, de la cual le mostramos el siguiente fragmento:

```
static ssize_t read_chan(struct tty_struct *tty, struct
file          *file, unsigned char *buf, size_t nr)
{
    . . .
    add_wait_queue(&tty->read_wait, &wait);
    set_bit(TTY_DONT_FLIP, &tty->flags);
    while (nr) {
        . . .
    }
}
```

Aquí podemos ver como el proceso se bloquea a la espera de que se produzca la escritura de teclado.

La entrada se producirá cuando se pulse una tecla. Por ello, el siguiente paso es mostrar la rutina de servicio de interrupción de teclado, que podemos ver en el archivo `drivers/char/pc_keybd.c` (aquí también podemos ilustrar como en Linux se encapsulan todas las dependencias del hardware en archivos específicos).

Lo primero es mostrar la asignación de los puertos del teclado, dado que estos son visibles, por ejemplo, visualizando el archivo `/proc/ioports`, y que encontramos en el archivo `include/linux/pc_keyb.h`:

```
#define KBD_STATUS_REG 0x64 /* Status register (R) */
```

```
#define KBD_CNTL_REG 0x64 /*Controller command register
(W)*/
#define KBD_DATA_REG 0x60 /* Keyboard data register (R/W)
*/
```

La lectura de esos puertos del código de rastreo de la tecla pulsada se hace a través de las operaciones *inb* que es dependiente de la plataforma y que podemos encontrar en el archivo *linux/include/asm-x8664/keyboard.h*:

```
#define kbd_read_input() inb(KBD_DATA_REG)
#define kbd_read_status() inb(KBD_STATUS_REG)
#define kbd_write_output(val) outb(val, KBD_DATA_REG)
#define kbd_write_command(val) outb(val, KBD_CNTL_REG)
```

Finalmente, tras insertar el carácter leído en el búfer, se procede a despertar al proceso que espera una pulsación de tecla. Esto se ilustra en el fragmento de código del archivo *include/linux/pc_keyb.h*:

```
static void keyboard_interrupt(int irq, void *dev_id, struct
pt_regs *regs)
{
...
    spin_lock_irq(&kbd_controller_lock);
    handle_kbd_event();
    spin_unlock_irq(&kbd_controller_lock);
}

static unsigned char handle_kbd_event(void)
{
    unsigned char status = kbd_read_status();
    unsigned int work = 10000;
    while (--work > 0) && (status & KBD_STAT_OBF) {
        unsigned char scancode;
        scancode = kbd_read_input();

```

Del archivo *linux/drivers/char/keyboard.c*

```
void handle_scancode(unsigned char scancode, int down)
200 {
...
221     if ((raw_mode = (kbd->kbdmode == VC_RAW))) {
222         put_queue(scancode | up_flag);

void put_queue(int ch)
{
    wake_up(&keypress_wait);

```

3. Conclusiones

Para estudiar la utilidad de estos ejemplos, realizamos una pequeña encuesta entre nuestros alumnos sobre la conveniencia o no de estos ejemplos, como ocurre en otros

casos [8]. Del total de 31 alumnos que asisten con regularidad a clase, 29 alumnos responden que el material puesto de ejemplo les ha sido útil para comprender los conceptos mostrados, y sólo 2 han mostrado su desacuerdo al no entender los ejemplos.

También, pregunte a los mismos alumnos sobre algún aspecto negativo sobre el material. En este caso, 11 respondieron que lo que les costaba ver eran aquellos ejemplos, o partes de ejemplos, donde se incluyen fragmentos de ensamblador. Los mismos alumnos indican que no hubiesen entendido los ejemplos si no hubiese sido por la explicación del profesor. Creemos que esto último se debe en parte a que los alumnos de segundo curso de carrera imparten en paralelo con Sistemas Operativos I, la asignatura de Estructura de Computadores que es donde ven ensamblador. Respecto a la comprensión de los fragmentos en C no ha habido ninguna indicación de que les sean incomprensibles.

Por ello, creemos que no nos equivocamos al indicar que los ejemplos diseñados han cumplido su función. Nos consta que los alumnos han estudiado los ejemplos. Nos falta por constatar si realmente han animado a nuestros estudiantes a seguir profundizando en la comprensión del kernel. Este último punto, pretendemos analizarlo en la asignatura de Sistemas Operativos II, que se imparte en el siguiente cuatrimestre, y que dedica una buena fracción de tiempo a estudiar el kernel de Linux.

4. References

1. A.S. Tanenbaum, A.S. Woodhull, *Sistemas Operativos: Diseño e implementación*, Prentice-Hall, 1998
2. W. Christopher, S. Procter, y T. Anderson, *The Nachos Instructional Operating System*, Proceeding 1993 USENIX Winter Technical Conference, 1993
3. G. Nutt, *Operating System (2nd ed)*, Addison Wesley, 2002
4. X. Amatriain, *El programari lliure a l'educació: guia per a la seva justificació i implementació*, III Jornades de Programari Lliure. Manresa, escola politècnica superior d'enginyeria, UPC, Juliol 2004. En <http://portal.jornadespl.org/biblioteca/iii-jornades/ponencies/34214-10-062004.pdf/view>
5. GD. P. Bovet, M. Cesati, *A Real Bottom-Up Operating System Course*, *Operating System Reviews* 35(1), pp. 48-60, ACM Press, January 2001
6. A. Silberschatz, G. Gagne, P.B. Galvin, *Operating System Concepts*, John Wiley & Sons, 2004
7. W. Stallings, *Operating Systems: Internals and Design Principles (5 ed)*, Prentice Hall, 2005
8. D. Carrington, S.K. Kim, *Teaching Software Design with Open Source Software*, 33RD ASEE/IEEE Frontiers in Education Conference, Vol. 3, pgs: SIC 9-14, Nov. 2003

Accessibility Requirements for educational packages in dotLRN

Olga Revilla Muñoz¹,

¹ Itakora.com
<http://www.itakora.com>
itakora@gmail.com

Abstract. dotLRN open source developers community is making a great effort to improve the user interface of this learning management system and to adapt it to the accessibility requirements proposed by the W3C Web Accessibility Initiative. This paper contributes to this task by analyzing the current accessibility status of three fundamental packages for dotLRN future: LORS, Assessment and IMS-LD and providing appropriate recommendations. These packages provide the educational standards support in the platform. More specifically, they implement SCORM, IMS-CP, IMS-QTI, IMS-LD and IMS-MD specifications.

Keywords: Accesibility, W3C, WAI, dotLRN, UAAG 1.0, WCAG 1.0, LORS, SCORM, IMS-QTI, IMS-LD, Evaluation.

1. Introduction

dotLRNⁱ is an open source e-learning platform based on the OpenACSⁱⁱ framework, which is intended for developing web based scalable applications. dotLRN is currently used by half a million users in higher education, government, non-profit, K-12, etcⁱⁱⁱ. Regarding its functionality, it is strongly compliant with educational standards for courses delivery (IMS-LD^{iv}, IMS-CP^v, IMS-MD^{vi}, IMS-QTI^{vii} and SCORM^{viii}). In particular, these standards are supported in the following dotLRN packages^{ix}: LORS (IMS-CP, IMS-MD and SCORM standards), Assessment (IMS-QTI standard) and IMS-LD.

The usage of educational standards in e-learning platforms allows the reusability of contents by the authors among different courses. However, in order to provide an inclusive support for learners with disabilities, it would be desirable to deliver those contents in the appropriate way to cope with the special needs that students may have. dotLRN is currently the only learning management system that covers IMS-LD and it supports a high level of accessibility features. However, the educational packages have not already been fully included in the last accessibility review^x.

In this paper, I analyze the three educational packages provided by dotLRN (i.e. LORS, Assessment and IMS-LD) from the accessibility point of view considering the following two guidelines from the W3C Web Accessibility Initiative: the 'Web

Content Accessibility Guidelines 1.0' (WCAG 1.0) and the 'User Agent Accessibility Guidelines 1.0' (UAAG 1.0). As a result, I provide some requirements for the improvement of these developments. These recommendations are contributed to the dotLRN open source community.

2. Analysis scope

This study does not cover the platform installation, the course creation, the course administration nor the course contents but the application usage from the learners' point of view. This analysis focuses on the structure and way of presenting the contents offered by the three educational packages: LORS, Assesments and IMS-LD.

LORS package consists on three pages, a frameset (lors.htm) which contains two other pages, one with the menu for the tree of contents defined by SCORM (lors_menu.htm) and the other which shows the course contents (lors_body.htm).

IMS-LD package consists on four pages, a frameset (IMS-LD.htm) which contains three other pages: one with the menu for the tree of contents defined by IMS-LD (imsld_tree.htm), the other which shows the course contents (imsld_activity-frame.htm) and a third frame that loads a complementary page (depending on the course at hand) that is not being analyzed here.

Assesment package consists on four pages, one with the form to present the questions (assesment.html), the page returned when the form is submitted, to show that the assessment has been submitted (assesment_return.html), the result of the answers (assesment_results.html) and the sessions log (assesment_sessions.html). Unlike LORS and IMS-LD packages, Assesment package is embedded within dotLRN user interface. For this reason, to avoid any interferences of dotLRN templates in this analysis, I have not taken into account the code surrounding the package, but just the code included in <DIV id=portal> tag, appart from the css and js related to the package.

I would like to remark that the above names for the HTML pages do not exist as such in the application. I have renamed them to identify them along the evaluation.

3. Methodology

dotLRN is a web server application, which is used by the users through web-browsers. For this reason, the analysis has to be done from the application point of view. Therefore, UAAG^{xi} should be applied. Moreover, there is also content generated by dotLRN application, fully independent from the course or questionnaire at hand. This content has to be analyzed with the WCAG^{xii}. Therefore, both UAAG and WCAG have been applied to analyze the accessibility of the educational packages.

In order to perform the evaluation according to the WCAG, first a automatic validation has been carried out with TAW3^{xiii} validation tool. This evaluation has been complemented with an

heuristic manual revision with Opera browser. This manual validation has been done with and without javascript, and with the functionality for screen reader where it is required by the checkpoints .

Regarding the UAAG evaluation, the analysis has been done with the heuristic method, since there is not any tool that make this evaluation in a automatic way. On the other hand, the assistant tool proposed by the W3C^{xiv} has been ruled out since it was not working at the time of the evaluation. Furthermore, it was not intended to adequate the study to accomplish any conformance profile since this study is a first approximation of UAAG to this packages. Once dotLRN community developers have made this packages comply with these guidelines, an official conformance profile can be achieve.

This analysis focuses just in WCAG and UAAG Priority 1 checkpoints to guide dotLRN developers in order to carry out the needed improvements to achieve conformance level A. WCAG and UAAG are complementary guidelines. Sometimes it may happen that they overlap. For instance, regarding Content Type labels “Visual Text” and “Image”, as well as in Events Label. For this reason, the analysis has been done for both, with the appropriate point of view.

4. Accessibility evaluation

This section includes the accessibility evaluation performed for LORS, Assessment and IMS-LD packages in dotLRN applying both WCAG and UAAG. A Web content development must satisfy priority 1. Otherwise, one or more groups of persons will find it impossible to access information in the document. Satisfying it is a basic requirement for some groups to be able to use Web documents.

4.1 WCAG evaluation

Next the errors detected after the evaluation according to WCAG is presented in a tabular format to facilitate the transmission of information to dotLRN community developers.

Table 1. LORS package WCAG 1.0 evaluation results

Page	Detected errors
lors.htm	2 errors, both related to 12.1 checkpoint (frames do not have a title attribute to allow the browser its identification and navigation). Moreover, although it is not specified in level 1 checkpoints, it has to be pointed out that when LORS is accessed, it takes the user to a page out of the context of the course space. This may cause disorientation in the learner.
lors_menu.htm	This page is not shown if scripts are deactivated (checkpoint 6.3), and there is not an alternative <NOSCRIPT> for this case. Consequently, checkpoint 8.1a gives an error, too.
lors_body.htm	This page shows errors if scripts are deactivated (checkpoint 6.3), and there is not an alternative <NOSCRIPT> for this case. Consequently, checkpoint 8.1a gives an error, too.

Table 2. IMS-LD package WCAG 1.0 evaluation results

Page	Detected errors
IMS-LD.htm	3 errors related to 12.1 checkpoint (frames do not have a title attribute to allow the browser its identification and navigation). As in LORS package, when IMS-LD is accessed, it takes the user to a page out of the context of the course space. This may cause disorientation in the learner.
Imslld_tree.htm	This page is shown correctly if scripts are deactivated (checkpoint 6.3), but there is not an alternative <NOSCRIPT> for this case. Consequently, checkpoint 8.1a gives an error, too.
imslld _activity-frame.htm	If javascript is deactivated, some functionality is lost. However this functionality is not fundamental for the working of the application, such as minimizing the menu frame. Moreover, there is an <iframe> tag which contains the course material. This iframe tag does not have its corresponding alternative (checkpoint 6.2). This implies that a browser that does not support iframes cannot show the course contents, i.e., the most important part of the application.

Table 3. Assessment package WCAG 1.0 evaluation results

Page	Detected errors
assesment.html	The page levels are ordered in the wrong order (checkpoint 6.1). For this reason, the user may be confused when browsing the application with style sheets deactivated.
assesment_return.html	The page is not ordered in levels (checkpoint 6.1). For this reason, the user may be confused when browsing the application with style sheets deactivated. Moreover, I suggest avoid using the BLOCKQUOTE tag.
assesment_results.html	The data tables where results are shown are not identified with headings (checkpoint 5.1). Moreover, checkpoint 5.2 is not achieved either, since there are nested tables which are not clearly identified. The page is not ordered in levels (checkpoint 6.1).
assesment_sessions.html	The page levels are ordered in the wrong order (checkpoint 6.1).

4.2 UAAG evaluation

Next the errors detected after the evaluation according to UAAG are presented in a tabular format to facilitate the transmission of information to dotLRN community developers. If the user agent does not satisfy priority 1, one or more groups of users with disabilities will find it impossible to access the Web. Satisfying it is a basic requirement for enabling some people to access the Web. The scope for this analysis includes: a) Input modalities, b) Output modalities, c) Size and color of non-text content, d) Background image interference and e) User control of every user interface component.

The following legends have been used:

- FAILS: It does not comply with the checkpoint and provision
- PASSES: It complies with the checkpoint and provision

- N/A: This checkpoint and / or provision is not applicable for this application.
- BROWSER DEPENDANT (BD): The browser satisfies the checkpoint.

Table 4. UAAG 1.0 evaluation results for LORS, Assessment and IMS LD packages

Checkpoints	Provisions	LORS	Assmnt.	IMS-LD
<u>1.1</u> Full keyboard access	Users can operate, through keyboard input alone, any user agent functionality available through the user interface.	PASSES	PASSES	PASSES
<u>1.2</u> Activate event handlers	Users can activate, through keyboard input alone, all input device event handlers that are explicitly associated with the element designated by the content focus.	PASSES	PASSES	PASSES
	Users must be able to activate as a group all event handlers of the same input device event type.	N/A	N/A	N/A
<u>1.3</u> Provide text messages	Every message that is a non-text element and is part of the user agent user interface has a text equivalent.	FAILS	PASSES	FAILS
<u>2.1</u> Render content according to specification	Content renders according to format to specification	FAILS	FAILS	FAILS
<u>2.2</u> Provide text view		N/A	N/A	N/A
<u>2.3</u> Render conditional content	Allow configuration to provide access to each piece of unrendered conditional content.	FAILS	PASSES	FAILS
	When a specification does not explain how to provide access to this content, do so as follows...(4a) allow the user to follow a link to C from the context of D.	PASSES	N/A	PASSES
<u>2.4</u> Allow time-independent interaction		N/A	N/A	N/A
<u>2.5</u> Make captions, transcripts, audio descriptions available		N/A	N/A	N/A
<u>2.6</u> Respect synchronization cues		N/A	N/A	N/A
<u>3.1</u> Toggle background images		N/A	N/A	N/A
<u>3.2</u> Toggle audio, video, animated images		N/A	N/A	N/A
<u>3.3</u> Toggle animated or blinking text		N/A	N/A	N/A
<u>3.4</u> Toggle scripts		N/A	N/A	N/A
<u>3.5</u> Toggle automatic content retrieval	The user agent only retrieves content on explicit user request.	PASSES	PASSES	PASSES

Checkpoints	Provisions	LORS	Assmnt.	IMS-LD
4.1 Configure scale	text Allow global configuration of the scale of visually rendered text content. Preserve distinctions in the size of rendered text as the user increases or decreases the scale.	BD	PASSES	BD
	Provide a configuration option to override rendered text sizes specified by the author or user agent defaults.	BD	N/A	BD
	Offer a range of text sizes to the user (...)	BD	N/A	BD
4.2 Configure font family	font Allow global configuration of the font family of all visually rendered text content.	BD	BD	BD
	Provide a configuration option to override font families specified by the author or by user agent defaults.	BD	BD	BD
	Offer a range of font families to the user ...	BD	BD	BD
4.3 Configure colors	text Allow global configuration of the foreground and background color of all visually rendered text content.	BD	BD	BD
	Provide a configuration option to override foreground and background colors specified by the author or user agent defaults.	BD	BD	BD
	Offer a range of colors to the user...	BD	BD	BD
4.4 Slow multimedia		N/A	N/A	N/A
4.5 Start, stop, pause, and navigate multimedia		N/A	N/A	N/A
4.6 Do not obscure captions		N/A	N/A	N/A
4.7 Global volume control		N/A	N/A	N/A
4.8 Independent volume control		N/A	N/A	N/A
4.9 Configure synthesized speech rate		BD	BD	BD
4.10 Configure synthesized speech volume		BD	BD	BD
4.11 Configure synthesized speech characteristics		BD	BD	BD
4.14 Choose style sheets	style Allow the user to choose from and apply alternative author style sheets.	BD	BD	BD
	Allow the user to choose from and apply at least one user style sheet.	BD	BD	BD
	Allow the user to turn off author and user style sheets.	BD	BD	BD
6.1 Programmatic access to HTML/XML infocet		N/A	N/A	N/A
6.2 DOM access to HTML/XML content		N/A	N/A	N/A
6.3 Programmatic access to non-HTML/XML content		N/A	N/A	N/A
6.4 Programmatic access to information about rendered content		B/D	B/D	B/D
6.5 Programmatic operation of user agent user interface		B/D	B/D	B/D
6.6 Programmatic notification of changes		B/D	B/D	B/D
6.7 Conventional keyboard APIs		PASSES	PASSES	PASSES
6.8 API character encodings		PASSES	PASSES	PASSES
7.1 Respect focus and selection conventions		PASSES	PASSES	PASSES
7.2 Respect input configuration conventions		PASSES	PASSES	PASSES

Checkpoints	Provisions	LORS	Assmnt.	IMS-LD
<u>8.1</u> Implement accessibility features		FAILS	FAILS	FAILS
<u>9.1</u> Provide content focus	Provide at least one content focus for each viewport (including frames) where enabled elements are part of the rendered content.	PASSES	PASSES	PASSES
	Allow the user to make the content focus of each viewport the current focus.	PASSES	PASSES	PASSES
<u>9.2</u> Provide user interface focus		PASSES	PASSES	PASSES
<u>9.3</u> Move content focus	Allow the user to move the content focus to any enabled element in the viewport.	PASSES	PASSES	PASSES
	Allow configuration so that the content focus of a viewport only changes on explicit user request.	PASSES	PASSES	PASSES
	If the author has not specified a navigation order...	N/A	N/A	N/A
<u>9.4</u> Restore viewport history	For user agents that implement a viewport history mechanism, for each state in a viewport's browsing history, maintain information about the point of regard, content focus, and selection.	B/D	B/D	B/D
	When the user returns to any state in the viewport history (e.g., via the "back button"), restore the saved values for the point of regard, content focus, and selection.	FAILS	FAILS	FAILS
<u>10.1</u> Associate table cells and headers	For graphical user agents that render tables, for each table cell, allow the user to view associated header information.	N/A	FAILS	N/A

Checkpoints	Provisions	LORS	Assmnt.	IMS-LD
<u>10.2</u> Highlight selection, content focus, enabled elements, visited links	Allow global configuration to highlight the following four classes of information in each viewport: the selection, content focus, enabled elements, and recently visited links.	B/D	B/D	B/D
	For graphical user interfaces, allow at least one configuration where the highlight mechanisms for the four classes of information: differ from each other, and do not rely on rendered text foreground and background colors alone.	B/D	B/D	B/D
	For graphical user interfaces if a highlight mechanism involves text size, font family, rendered text foreground and background colors, or text decorations, offer at least the following range of values...	B/D	B/D	B/D
	4. Highlight enabled elements according to the granularity specified in the format.	B/D	B/D	B/D
<u>10.6</u> Highlight current viewport	Highlight the viewport with the current focus (including any frame that takes current focus).	PASSES	PASSES	PASSES
	For graphical viewports, as part of satisfying provision one of this checkpoint, provide at least one highlight mechanism that does not rely on rendered text foreground and background colors alone	PASSES	PASSES	PASSES
	If the techniques used to satisfy provision one of this checkpoint involve rendered text size, font family, rendered text foreground and background colors, or text decorations, allow global configuration and offer same ranges of values required by provision three of <u>checkpoint 10.2</u> .	FAILS	FAILS	FAILS
<u>11.1</u> Provide information to the user about current user preferences for input configurations.	N/A	N/A	N/A	
<u>12.1</u> Provide accessible documentation (level Double-A WCAG 1.0)	FAILS	FAILS	FAILS	
<u>12.2</u> Provide documentation of accessibility features	FAILS	FAILS	FAILS	
<u>12.3</u> Provide documentation of default bindings	N/A	N/A	N/A	

The following table summarizes the above results.

Table 5. Summary of checkpoint status for the educational packages

	LORS	ASSESSMENTS	IMS-LD
FAILS	8	7	8
PASSES	15	17	15
BD	23	20	23
N/A	21	23	21

5. Conclusions and recommendations to dotLRN community

This paper presents the accessibility evaluation performed for LORS, Assessment and IMS-LD packages in dotLRN applying both WCAG and UAAG. The analysis performed intends to help dotLRN community produce software that is expected to be more flexible, manageable, extensible, and beneficial to all users.

In my opinion, future developments should be focused on those areas that require more accessibility improvements in each package, which are as follows for each of the guidelines analyzed in this paper:

WCAG 1.0

- LORS
 - Promote the independence of javascript code
 - Platform integration
 - Small changes in code
- ASSESSMENT
 - Improve session data rendering. This improvement implies a deep code change.
- IMS-LD
 - Promote the independence of javascript code
 - Platform integration
 - Small changes in code

UAAG 1.0

The following graphics summarize the accessibility evaluation performed. Data are obtained from Table 5.

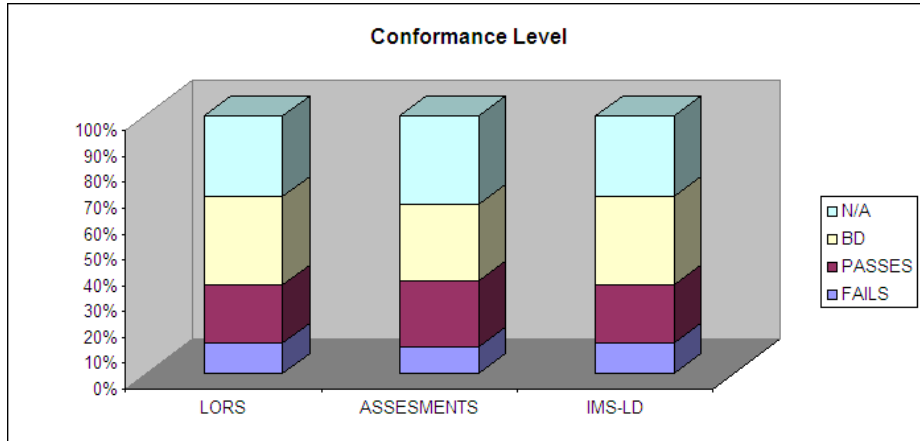


Figure 1. Conformance level.

Figure 1 shows that the 3 packages have provided similar results for UAAG evaluation. This implies that the design made follows the same approach regarding accessibility in the 3 packages.

Figures 2, 3 and 4 shows that, as far as dotLRN is a web-server application and the user interacts with it through a web browser, it counts on the browsers for many accessibility features. This does not mean a failure in the accessibility compliance, but a benefit derived from being a web-based application. Each figure corresponds to each educational package analyzed.

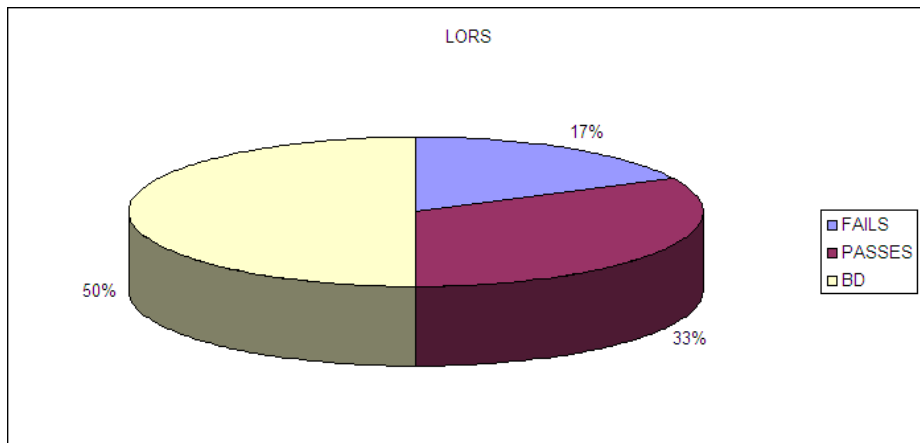


Figure 2. LORS package results

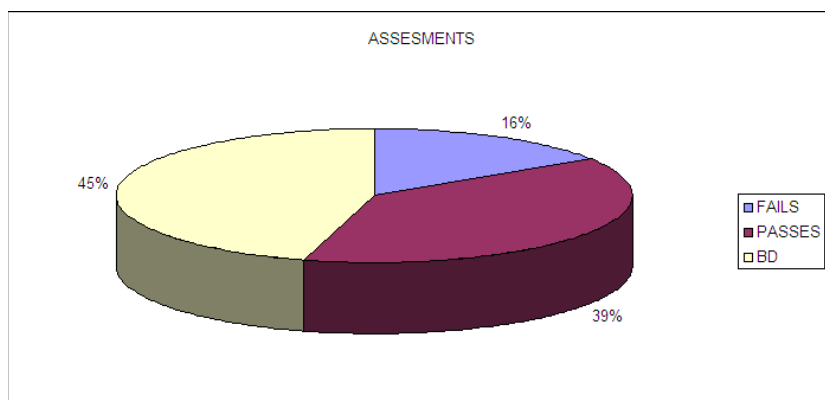


Figure 3. Assessment package results

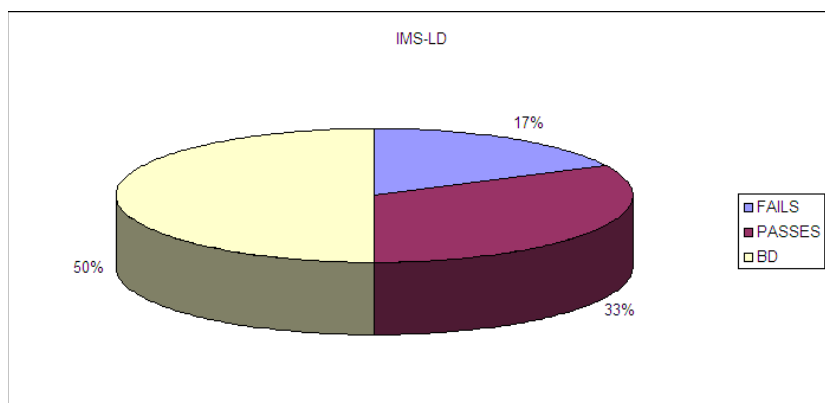


Figure 4. IMS-LD package results

Data for optimism

The percentage of features that pass checkpoints is very high in the three packages (around 75% considering both browser dependent and passed), which demonstrates a remarkable developing effort from dotLRN community. Nevertheless, I would suggest continuing working on the checkpoints to reach a W3C WAI level A of accessibility for the packages analyzed.

6. References

See footnotes at the end of the paper.

- i. Known as **.LRN**: <http://dotlrn.org>
- ii. Open Architecture Community System: <http://openacs.org>
- iii. dotLRN users: <http://dotlrn.org/users/>
- iv. Instructional Management Systems – Learning Design:
<http://www.imsglobal.org/learningdesign>
- v. Instructional Management Systems – Content Packaging:
<http://www.imsglobal.org/content/packaging>
- vi. Instructional Management Systems – Metadata:
<http://www.imsglobal.org/metadata>
- vii. Instructional Management Systems – Question & Test Interoperability:
<http://www.imsglobal.org/question>
- viii. Sharable Content Object Reference Model:
<http://www.adlnet.gov/scorm/index.cfm>
- ix. Educational standard support in dotLRN:
http://openacs.org/xowiki/Educational_Standards
- x. Accessibility status in dotLRN: <http://openacs.org/xowiki/Accessibility>
- xi. The User Agent Accessibility Guidelines (UAAG) documents explain how to make user agents accessible to people with disabilities, particularly to increase accessibility to Web content. User agents include Web browsers, media players, and assistive technologies, which are software that some people with disabilities use in interacting with computers. User Agent Accessibility Guidelines Overview
<http://www.w3.org/WAI/intro/uaag.php>
- xii. Web "content" generally refers to the information in a Web page or Web application, including text, images, forms, sounds, and such. Web Content Accessibility Guidelines Overview
<http://www.w3.org/WAI/intro/wcag.php>
- xiii. TAW (Web Accessibility Test) is a tool for the analysis of Web sites, based on the W3C - Web Content Accessibility Guidelines 1.0 (WCAG 1.0)
<http://www.tawdis.net/taw3/cms/es>
- xiv. UAAG 1.0 Evaluation Form Generator
<http://www.w3.org/WAI/UA/2002/08/eval>

phpRADmin project

Administración de FreeRADIUS vía web

Toni de la Fuente Díaz

Fundación I+D del Software Libre. Granada.
toni@blyx.com

Resumen. phpRADmin es una herramienta escrita en PHP para administrar FreeRADIUS vía web con MySQL como base de datos. Permite gestionar la seguridad de una red en diferentes niveles de una forma sencilla e intuitiva.

Keywords: RADIUS, PKI, EAP, WPA, TLS, TTLS, CA, security, phpRADmin.

1. Objetivos:

Es creado para ocupar un hueco que existe actualmente en el mundo del Software Libre para la gestión, configuración y administración global de FreeRADIUS y las redes basadas en AAA. Generalmente este es un proceso complicado de modo que intentamos hacerlo un poco más sencillo con phpRADmin.

2. Definición:

phpRADmin es una herramienta escrita en PHP para administrar FreeRADIUS vía Web con MySQL como backend. Esta interface permite al administrador configurar, buscar, crear y editar usuarios en una base de datos SQL (MySQL), crear grupos, almacenar información de la actividad de los usuarios, hacer consultas avanzadas sobre la base de datos, comprobar el funcionamiento del servidor, generar estadísticas de uso, gestión (crear, actualizar, revocar) certificados PKI para EAP (EAP-TLS, PEAP, etc), gestionar el diccionario de datos y monitorización del sistema.

Es una forma de administrar FreeRADIUS junto a WPA, EAP (802.1X), PPP, PPPoE, Captive Portal, Sistemas VoIP y otros métodos de autenticación.

phpRADmin incorpora gráficos, auditoría de uso, perfiles de administración en tres niveles y posibilidad de exportar la base de datos vía web para facilitar la migración. phpRADmin está preparado para funcionar con Chillispot y WISP.

Datos del proyecto:

- Licencia: GPL
- Lenguaje de programación: PHP, Perl y shell script.
- Plataformas soportadas: Linux, FreeBSD.
- Idiomas soportados: Inglés, Español (fácil para traducir a otros idiomas).

phpRADmin está disponible en paquete instalable y en versión LIVE CD/USB basada en Fedora Core 3 (linux-live.org).

El proyecto phpRADmin debe su nombre en honor a las famosas y útiles herramientas para administrar MySQL y LDAP, vía web llamadas phpMyAdmin y phpLDAPadmin.

En la actualidad no hay ninguna herramienta de Software Libre para la gestión global y configuración de redes basadas en RADIUS con las últimas características que éste ofrece, por ejemplo protocolos de la familia EAP.

Herramientas análogas:

Libres:

- Dialup Admin (incorporado en phpRADmin)
- pRadius
- ARA - Asn Radius Admin

Propietarias:

- Juniper: JunOS Stell-Belted Admin
- OSC: Radiator RAdmin
- Airspan (Radionet): Network Controller
- Witelcom: WISE – RIMS
- etc.

3. Conceptos:

A continuación se describen varios de los conceptos necesarios para comprender el objetivo de este documento.

Protocolo RADIUS:

RADIUS es acrónimo en inglés de Remote Access Dialin User Service, es un protocolo de autenticación, autorización y gestión de clientes/usuarios para aplicaciones de acceso a la red o movilidad IP. Éste ha vuelto a cobrar importancia en las telecomunicaciones dada la rápida implantación de protocolos como 802.1x y

todas sus variantes. Permitiendo, de este modo, configurar redes pequeñas, medianas y grandes (hasta millones de usuarios) para dotarlas de mayor seguridad y movilidad.

AAA:

Authentication: Verifica la identidad con credenciales (user/password, certificados, tokens, etc.). *Authorization*: Derechos y privilegios para acceder a los recursos tras la autenticación. *Accounting* (gestión de cuentas): Gestiona los privilegios. ACL. Log, monitorización, informes y capacidad de facturación.

FreeRADIUS:

Implementación libre más conocida y usada del protocolo RADIUS (Remote Authentication Dial In User Service), este protocolo es usado para autenticar, autorizar y gestionar el acceso a redes por parte de usuarios y dispositivos. Muy usado para redes de acceso a Internet como las ofrecidas por ISP (Proveedores de Servicios de Internet), ya sea por cable o por algún medio inalámbrico. Hay un nuevo protocolo que intenta mejorar y aumentar las funcionalidades de RADIUS, este protocolo se llama DIAMETER.

PKI:

Infraestructura de clave pública (Public Key Infraestructura). Es una combinación de hardware y software, políticas y procedimientos de seguridad que permiten la ejecución con garantías de operaciones criptográficas como el cifrado, la firma digital o el no repudio de transacciones electrónicas. El término PKI se utiliza para referirse tanto a la autoridad de certificación y al resto de componentes, como para referirse, de manera más amplia y a veces confusa, al uso de algoritmos de clave pública en comunicaciones electrónicas. Este último significado es incorrecto, ya que no se requieren métodos específicos de PKI para usar algoritmos de clave pública.

802.1x:

Estándar del IEEE para control de admisión a red basado en puertos. Permite la autenticación de dispositivos conectados a un puerto LAN. Se implementó para evitar el acceso directo a una red por el simple hecho de enchufar un dispositivo a un concentrador (hub) o conmutador (switch).

EAP:

Extensible Authentication Protocol, es uno de los elementos básicos del protocolo 802.1x y desarrollado para ampliar la seguridad y funcionalidades del protocolo PPP (Point to Point Protocol). PPP usa "usuario" y "contraseña" de modo que con EAP se pueden utilizar otros métodos más específicos y sofisticados para realizar autenticación. PPPoE es el estándar que permite encapsular PPP sobre tramas ethernet.

Captive Portal o Portal Cautivo:

Es un programa o máquina de una red informática que vigila el tráfico HTTP y fuerza a los usuarios a pasar por una página especial si quieren navegar por Internet

de forma normal. A veces esto se hace para pedir una autenticación válida, o para informar de las condiciones de uso de un servicio wireless (que es donde más se encuentran). Chillispot es uno de los programas de portal cautivo más conocidos. NoCat Auth fue de los primeros desarrollos que implementaban esta idea.

4. Componentes:

phpRADmin es un puzzle que contiene muchas piezas. Para evitar reinventar la rueda se han utilizado otras herramientas y además se ha escrito desde cero parte del código con php y shell script. Las herramientas que se han adaptado y modificado son:

- Oreon: se ha utilizado parte de su código como base para toda la aplicación.
- Dialup admin: para gestión de usuarios y clientes.
- Phpki: para la gestión de los certificados.
- Phpconfig: para la edición de archivos vía web.

5. Instalación:

En el archivo INSTALL del paquete de instalación se encuentra la documentación extendida sobre todos los aspectos de la instalación siguiendo unos sencillos pasos.

phpRADmin funcionará correctamente en cualquier entorno LAMP (Linux Apache MySQL y PHP). Adicionalmente se deberán instalar los paquetes de software que se citan a continuación:

- FreeRADIUS (con soporte MySQL).
- Radiusclient.
- Net-SNMP y Net-SNMP utils.
- Módulos de PHP: php-gd, php-snmp.
- Módulos de Perl: DateManip, MD5, perl-rrdtool.
- Freetype.
- RRDtool y RRDtool utils.
- Sudo.

Tras la instalación del software necesario deberemos configurar FreeRADIUS para usar MySQL como repositorio de usuarios para autenticación (archivos radiusd.conf y sql.conf).

Configuración de php para editar algunos archivos desde phpRADmin:

```
memory_limit = 16M
```


Tras configurar la base de datos y colocar los archivos necesarios podremos abrir un navegador y seguir con la instalación a través del asistente que encontraremos en <http://servidor/phpradmin>

6. Capturas:

En la figura 1 podemos ver la página principal donde se encuentran los gráficos de usuarios conectados, datos de los usuarios en la base de datos, estado de los clientes y estado de la base de datos. En la parte superior encontramos el menú con las diferentes opciones principales:

Home: Página inicial, concretamente la que muestra la figura 1.

Users: Página de gestión de los usuarios, grupos, certificados y PKI (ver figura 2).

Clients/NAS: Página de gestión de los dispositivos que utilizaremos para autenticar a los usuarios podremos añadir, modificar y eliminar (ver figura 3).

Reporting: podemos observar estadísticas generales del servicio, de usuarios, gráficas de usuarios, clientes y base de datos con el histórico de las gráficas. También podremos ver los logs del servidor RADIUS (ver figura 4).

Options: En esta sección podemos realizar todas las configuraciones necesarias para ajustar y administrar phpRADmin y para configurar FreeRADIUS (ver figura 5).

Billing: actualmente en desarrollo, albergará la sección para administrar las facturas, planes y costes.

Adicionalmente, en la parte superior derecha cada página podemos ver el usuario conectado, fecha, hora y opción para salir del entorno de administración. Y en el pie de página vemos el tiempo que ha tardado el servidor en generar la página mostrada además de los créditos e información básica del proyecto.

Figura 1:

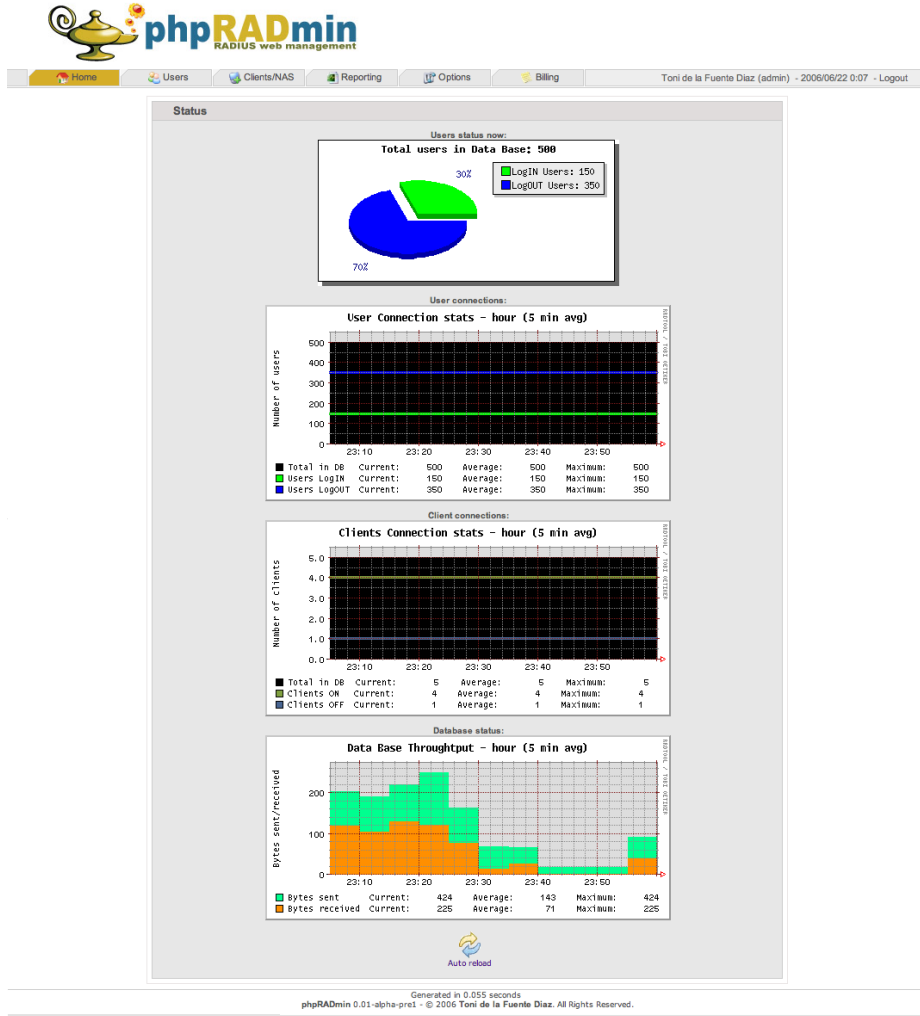



Figura 2:

The screenshot displays the phpRADmin web management interface. The top navigation bar includes links for Home, Users, Clients/NAS, Reporting, Options, and Billing. The current user is identified as 'Toni de la Fuente Diaz (admin)' with a timestamp of '2006/06/22 0:11' and a 'Logout' option. The left sidebar contains a menu with sections for Users, Groups, and Certificates. The main content area is titled 'User Preferences for new user' and contains a form with the following fields and options:

- Username:
- Password:
- Group:
- Name (First Name Surname):
- Mail:
- Department:
- Home Phone:
- Work Phone:
- Mobile Phone:
- Auth-Type:
- Simultaneous Use:
- Protocol:
- IP Address:
- IP Netmask:
- Framed-MTU:
- Compression Used:
- Service Type:
- Session Timeout:
- Idle Timeout:
- Port Limit:
- Lock Message:
- Daily Limit (secs):
- Weekly Limit (secs):
- User Login Period (UUCP Format):
- User Expiration Date:
- ChilliSpot Max Input Octets:
- ChilliSpot Max Output Octets:
- ChilliSpot Max Total Octets:

At the bottom of the form, there are buttons for 'Create' and 'Show User', and a link for 'Operator's Help'. The footer of the page indicates the page was generated in 0.035 seconds and is part of phpRADmin 0.01-alpha-pre1, © 2006 Toni de la Fuente Diaz. All Rights Reserved.

Figura 3:



Home Users Clients/NAS Reporting Options Billing Toni de la Fuente Diaz (admin) - 2008/06/22 0:17 - Logout

Clients Manage Apply

IMPORTANT NOTES:
Advanced configuration. Do not modify if you do not know what are you doing.
Remember to Unlink and Apply when you make changes here.

NAS Administration

NAS List: 192.168.1.30

NAS Name:

NAS Short Name:

NAS Type: usriper

NAS Ports Number:

NAS Secret:


NAS SNMP community:

NAS Description:

Change NAS/Client Info Perform Action

Add NAS/Client

Figura 4:



Home Users Clients/NAS Reporting Options Billing Toni de la Fuente Diaz (admin) - 2008/06/22 0:23 - Logout

Reporting Online users Accounting Global Stats User Stats

Graphs User graphs Client graphs DB graphs

Logs RADIUS logs System logs

System Logs

Last 30 lines for System Log

```

Jun 21 23:20:02 moon crond(pam_unk[4346]: session closed for user root
Jun 21 23:25:01 moon crond(pam_unk[4392]: session opened for user root by (uid=0)
Jun 21 23:25:02 moon crond(pam_unk[4392]: session closed for user root
Jun 21 23:30:01 moon crond(pam_unk[4464]: session opened for user root by (uid=0)
Jun 21 23:30:03 moon crond(pam_unk[4464]: session closed for user root
Jun 21 23:31:27 moon proftpd[4056]: moon.blyk.com [192.168.1.250][192.168.1.250] - FTP no transfer timeout, disconnected
Jun 21 23:31:27 moon proftpd[4056]: session closed for user root
Jun 21 23:31:27 moon proftpd[4056]: moon.blyk.com [192.168.1.250][192.168.1.250] - FTP session closed.
Jun 21 23:35:01 moon crond(pam_unk[4507]: session opened for user root by (uid=0)
Jun 21 23:35:02 moon crond(pam_unk[4507]: session closed for user root
Jun 21 23:40:01 moon crond(pam_unk[4553]: session opened for user root by (uid=0)
Jun 21 23:40:02 moon crond(pam_unk[4553]: session closed for user root
Jun 21 23:45:01 moon crond(pam_unk[4589]: session opened for user root by (uid=0)
Jun 21 23:45:02 moon crond(pam_unk[4589]: session closed for user root
Jun 21 23:50:01 moon crond(pam_unk[4625]: session opened for user root by (uid=0)
Jun 21 23:50:03 moon crond(pam_unk[4625]: session closed for user root
Jun 21 23:55:01 moon crond(pam_unk[4661]: session opened for user root by (uid=0)
Jun 21 23:55:02 moon crond(pam_unk[4661]: session closed for user root
Jun 22 00:00:01 moon crond(pam_unk[4730]: session opened for user root by (uid=0)
Jun 22 00:00:02 moon crond(pam_unk[4730]: session closed for user root
Jun 22 00:01:01 moon crond(pam_unk[4780]: session opened for user root by (uid=0)
Jun 22 00:01:01 moon crond(pam_unk[4780]: session closed for user root
Jun 22 00:05:01 moon crond(pam_unk[4952]: session opened for user root by (uid=0)
Jun 22 00:05:02 moon crond(pam_unk[4952]: session closed for user root
Jun 22 00:10:01 moon crond(pam_unk[4995]: session opened for user root by (uid=0)
Jun 22 00:10:03 moon crond(pam_unk[4995]: session closed for user root
Jun 22 00:15:01 moon crond(pam_unk[5039]: session opened for user root by (uid=0)
Jun 22 00:15:02 moon crond(pam_unk[5039]: session closed for user root
Jun 22 00:20:01 moon crond(pam_unk[5105]: session opened for user root by (uid=0)
Jun 22 00:20:02 moon crond(pam_unk[5105]: session closed for user root

```

Generated in 0.16 seconds
phpRADmin 0.01-alpha-pret - © 2006 Toni de la Fuente Diaz. All Rights Reserved.

Figura 5:

The screenshot displays the phpRADmin web management interface. At the top, there is a navigation bar with tabs for Home, Users, Clients/NAS, Reporting, Options (selected), and Billing. The user is identified as Toni de la Fuente Diaz (admin) on 2006/06/22 0:23, with a Logout option.

The main content area is titled "General Options" and contains the following configuration fields:

- phpRADmin installation folder: `/usr/local/phpRADmin/`
- RADIUS binary files folder: `/usr/sbin/`
- RADIUS config files folder: `/etc/raddb/`
- RADIUS Dictionary Path: `/usr/share/freeradius/`
- RADIUS startup script: `/etc/init.d/radiusd`
- Interface refresh: `60` seconds
- RRDTool binary path: `/usr/bin/rrdtool`
- Sudo binary path: `/usr/bin/sudo`
- System log file path: `/var/log/messages`
- FreeRADIUS radius.log file path: `/var/log/radius/radius.log`
- Session expiration time: `unlimited` minutes

Below the fields, there is a "Color of the errors" section with a legend:

- The directory or file do not exist
- The file is not executable
- The directory or file is not writable
- The directory is not readable

A "Save" button is located at the bottom of the configuration area. At the very bottom of the page, it says "Generated in 0.048 seconds" and "phpRADmin 0.01-alpha-pre1 - © 2006 Toni de la Fuente Diaz. All Rights Reserved."

7. Referencias:

1. Página oficial del proyecto: <http://www.phpradmin.org>
2. Wikipedia, La enciclopedia libre. <http://es.wikipedia.org>
3. FreeRADIUS web oficial: <http://www.freeradius.org>
4. RADIUS Securing Public Access to Private Resources, Jonathan Hassell, October 2002. Ed. O'reilly
5. AAA and Network Security for Mobile Access: Radius, Diameter, EAP, PKI and IP Mobility. Madjid Nakhjiri (Author), Mahsa Nakhjiri (Author). 2006. Ed. Wiley
6. Chillispot <http://www.chillispot.org>
7. NoCat Auth <http://www.nocat.net>
8. Oreon project <http://www.oreon-project.org>
9. Dialup Admin <http://www.freeradius.org/dialupadmin.html>

Proyecto RESTAD: Herramientas de Código Libre para la Traducción y Postedición de Documentos

Luis Villarejo Muñoz,
Joaquim Moré López,
Mercè Vázquez García

Universitat Oberta de Catalunya
{lvillarejo, mvazquezga, jmore}@uoc.edu

Resumen. En este artículo presentamos tres herramientas de código libre en desarrollo por el Servicio Lingüístico de la Universitat Oberta de Catalunya (UOC), con la financiación del proyecto RESTAD (Recursos de Soporte a la Traducción Automática Aplicados a la Docencia). El objetivo del proyecto era fomentar la creación de herramientas y recursos de ayuda a la traducción y revisión de documentos, que fueran de código abierto, para las universidades implicadas en este proyecto. La primera herramienta (Stem-LES) es un extractor automático de terminología que incorpora la búsqueda automática de equivalentes de traducción. La segunda herramienta (Frog Translator) es un sistema de traducción asistida que se ha desarrollado añadiendo funcionalidades al sistema de código libre llamado ForeignDesk. Y, finalmente, la tercera herramienta, CREN (del catalán, Cerca i Revisió d'Enllaços), tiene como función detectar los enlaces inactivos de un documento o página web y buscar la nueva dirección de las páginas referidas por el enlace inactivo.

Palabras clave: extracción de terminología, sistema de traducción asistida, traducción automática, proyectos de traducción, postedición automática

1. Introducción

En este artículo presentamos tres herramientas de código libre, vinculadas con la traducción y revisión de documentos, que se han desarrollado en el Servicio Lingüístico de la Universitat Oberta de Catalunya (UOC), bajo el patrocinio del proyecto RESTAD¹ (Recursos de Soporte a la Traducción Automática Aplicados a la Docencia). Este proyecto nace de las necesidades que tienen los servicios lingüísticos de las universidades para realizar la gestión de la oferta docente[1]. Ha sido financiado por el Departamento de Educación y Universidades de la Generalitat de Cataluña y en él han participado, además del Servicio Lingüístico de la UOC, los

¹ Se puede acceder a una descripción más detallada del proyecto en el espacio del Servicio Lingüístico de la UOC en [2].

servicios lingüísticos de la Universitat Autònoma de Barcelona, la Universitat de Girona y la Universitat Politècnica de Catalunya.

El objetivo general del proyecto era desarrollar recursos que facilitaran y mejoraran la traducción de documentos docentes y documentos académicos administrativos del catalán al castellano o al inglés. Además de la traducción, se tuvo en cuenta la revisión de los documentos en cualquiera de estas lenguas. Ante la cantidad ingente de documentos docentes y administrativos es necesario introducir procesos automáticos en el flujo de traducción y revisión. Las universidades implicadas ya disponían de un sistema de traducción automática para los pares de lenguas catalán-castellano, castellano-catalán y también recurrían a herramientas de traducción asistida para las traducciones al castellano o al inglés. Sin embargo, estas herramientas de traducción asistida carecían de algunas opciones fundamentales para obtener un buen rendimiento del programa, por este motivo se desarrolló la herramienta de traducción asistida Frog Translator, una evolución de la herramienta ForeignDesk[5]. Asimismo, de lo que tampoco no disponían las universidades implicadas en el proyecto era de herramientas de revisión que ayudaran a agilizar trabajos que deben realizarse con regularidad pero que son lentos si se llevan a cabo manualmente, como verificar si los enlaces de un documento publicado hace un tiempo continúan siendo activos y ante un enlace inactivo, buscar la nueva ubicación de la página web referida en el enlace antiguo. Para ello se desarrolló la herramienta Cren, que pretende automatizar esta tarea en el mayor grado posible.

Además, para llevar a cabo la introducción de procesos automáticos de ayuda a la traducción asegurando la calidad del contenido se aprovechó la información lingüística que ya disponían los servicios lingüísticos de las universidades implicadas. Gran parte de esta información no está declarada explícitamente en fuentes de referencia, sino que está presente de forma implícita en los mismos documentos generados. Por ejemplo, aunque no exista un glosario de un dominio temático concreto, la terminología empleada con los equivalentes de traducción en castellano o en inglés de dicho dominio puede ser extraída a partir de los documentos docentes y administrativos puestos en paralelo con las traducciones correspondientes en las dos lenguas. En este contexto situamos la herramienta StemLES de extracción de terminología y los equivalentes de traducción, que presentaremos más adelante. Gracias a esta herramienta es posible crear recursos que mejoran la calidad lingüística de las traducciones, porque sirven tanto para actualizar el léxico del sistema de traducción automática como para alimentar las bases de datos terminológicas en un sistema de traducción asistida. Los recursos terminológicos generados son compartidos por las universidades implicadas, por lo que se crean en un formato estándar para facilitar el intercambio.

Aunque el proyecto se vincula directamente al ámbito universitario, nuestro objetivo principal es abrir este espacio de intercambio de recursos de ayuda a la traducción y la postedición a cualquier traductor o autor, sea cual sea el colectivo al que pertenezca. Además, pretendemos que el usuario tenga un alto grado de autonomía.

Por esta razón, las herramientas que presentamos son de código libre, lo que permite, por ejemplo, que el sistema de traducción asistida que presentaremos (Frog Translator) sea de libre acceso para cualquier usuario, sea éste un traductor profesional o el autor del documento que tiene que traducir, aunque no sea traductor profesional (docente, administrativo, etc.). Creemos que éste es un enfoque novedoso en el ámbito del software de código libre para traductores [3].

2. Descripción de las Herramientas

En este apartado presentamos las tres herramientas que hemos desarrollado en el Servicio Lingüístico de la UOC: Stem-LES, Frog Translator y Cren.

2.1. Stem-LES

Stem-LES (*Lexical Extraction Suite*) es un programa gratuito de extracción léxica de tipo estadístico, de libre distribución y de código abierto que ayuda a crear listas léxicas útiles para la traducción automática y asistida de una forma rápida y eficiente. Seguidamente presentamos las funcionalidades de esta herramienta.

a) Extracción Automática de Léxico Relevante

Stem-LES utiliza una técnica estadística de extracción automática de léxico relevante para la traducción automática y la traducción asistida. Esta técnica se basa en el cálculo de todos los *n-gramas* de palabras (normalmente desde $n = 2$ hasta una n determinada por el usuario, por defecto 3). Es decir, se calculan todas las combinaciones de dos palabras, de tres palabras, etc. Entre estas combinaciones se localizan las unidades léxicas relevantes que hay en el texto, pero también se localizan otras muchas combinaciones no relevantes.

Para reducir notablemente el número de candidatos no relevantes es imprescindible filtrar los resultados con una lista de palabras vacías (*stop words*). Las palabras vacías son las que habitualmente no se encuentran en la posición extrema (la primera o la última palabra) de una entrada léxica relevante. Normalmente las listas de palabras vacías son listas de palabras funcionales. Los resultados del proceso de extracción dependerán de la calidad de esta lista, por lo que el usuario podrá mantener y mejorar la lista simplemente editándola en un editor de textos.

Una vez eliminados todos los *n-gramas* que empiezan o terminan por una palabra vacía, ya disponemos de una lista de candidatos. A partir de aquí, hay que revisarla, puesto que no todos estos candidatos serán realmente relevantes. Mientras se hace la revisión de los candidatos, es conveniente ir añadiendo nuevas palabras a la lista de palabras vacías.

b) Búsqueda Automática de Equivalentes de Traducción

El programa Stem-LES detecta con métodos estadísticos la traducción más probable de una determinada unidad léxica a partir de un corpus paralelo. Este programa determina qué palabra o conjunto de palabras de un segmento traducido corresponde a la traducción de la unidad léxica en cuestión. Hay que tener en cuenta que se trata de un proceso estadístico y no siempre acierta en la selección; por este motivo, el programa muestra más de un posible candidato para que el usuario pueda escoger el que considere correcto. Stem-LES está desarrollado íntegramente en Perl y, por lo tanto, es un programa multiplataforma. Se distribuyen tanto los fuentes en Perl como la versión ejecutable para Windows. Stem-LES está disponible en la página web de SourceForge [3].

2.2. Frog Translator

La herramienta Frog Translator, que se distribuirá mediante SourceForge, es un entorno de traducción asistida desarrollado a partir del paquete Foreign Desk [4]. Éste último es un sistema de traducción asistida desarrollado y liberado, mediante su distribución bajo la licencia de software BSD, por la empresa Lion Bridge. Esta licencia de software, otorgada principalmente para los sistemas BSD (del inglés, *Berkeley Software Distribution*) y que pertenece al grupo de licencias de software libre, tiene menos restricciones en comparación con otras licencias como la GPL (del inglés, *General Public License*) y está muy cercana al dominio público; además, contrariamente a la GPL, permite el uso del código fuente en software no libre. Con el objetivo de facilitar la tarea de los traductores con el uso de Foreign Desk, realizamos un análisis para identificar posibles mejoras a incorporar en la herramienta. A continuación, pasamos a describir estas mejoras.

a) Base de Datos Terminológica

Cuando un usuario traduce un documento utilizando una aplicación de traducción asistida, tiene la opción de contar con la ayuda de una base de datos terminológica que contiene los nombres de las lenguas en las que se está trabajando. Este tipo de base de datos viene incorporada en Foreign Desk mediante la aplicación Termbase[6]. De esta forma, cada vez que se traduce un segmento, la base de datos terminológica es consultada y muestra los términos coincidentes en la ventana de terminología. Las mejoras incorporadas a la base de datos terminológica han sido las siguientes:

- Incorporación de un algoritmo de búsqueda no exacta: Foreign Desk no recuperaba palabras si no coincidían exactamente con la forma almacenada en la base de datos. En cambio, el algoritmo de búsqueda no exacta implementado no utiliza la flexión morfológica, de modo que la búsqueda pasa a ser independiente del idioma de trabajo.

- Gestión terminológica basada en el formato UTF-8: esto permite una correcta recuperación y visualización de la terminología.
- Incorporación de una segmentación adecuada para la mejora del almacenaje de la terminología: el sistema almacenaba las palabras junto con los signos de puntuación, lo que provocaba que la terminología no fuese correctamente recuperada.

b) Importación de Documentos

Cuando se crea un nuevo proyecto de traducción mediante el asistente de creación de proyectos FDPA (del inglés, Foreign Desk Project Assistant), cuyas funciones explicamos más adelante, es necesario especificar en qué formato están los documentos con los que se desea trabajar. Dado que el asistente FDPA no permitía crear proyectos con documentos en formato Word, ni en formato Open Office Writer, ni en formato Xliff y que los dos primeros son de uso común entre nuestros traductores y que el último es necesario como formato de intercambio, nos planteamos este desarrollo para incorporarlos en la aplicación. Asimismo, el asistente FDPA no permitía incorporar nuevos documentos al proyecto una vez que éste había sido creado, lo que limitaba bastante la tarea de traducción. En este sentido, también llevamos a cabo el desarrollo para superar esta limitación.

c) Incorporación de Traducción Automática

Hemos incorporado un servicio de traducción automática vía web catalán-castellano y castellano-catalán que utiliza el servicio web de traducción automática de la Generalitat de Catalunya. En este sentido, el usuario tiene la posibilidad de activar o no activar la opción que le presenta la propuesta de traducción automática del servicio web, es decir, si activa esta opción, entonces es cuando se pone en marcha este servicio vía web, y en pantalla se muestra la propuesta de traducción. A partir de aquí, el usuario puede modificar esta propuesta o escribir una nueva solución de traducción.

d) Adaptación para el Funcionamiento en Lengua Catalana

A la hora de crear un nuevo proyecto de traducción se debe utilizar el asistente FDPA, que genera automáticamente la estructura de directorios y memorias de traducción correspondientes a un proyecto de Foreign Desk. Este asistente consiste en una serie de interfaces gráficas donde se introduce la información asociada al nuevo proyecto (nombre del proyecto, directorio de trabajo, lengua origen, lengua destino, etc.). Anteriormente, el FDPA no contemplaba la lengua catalana y no permitía crear proyectos de traducción que la incluyera. Una vez hechas las modificaciones pertinentes, es posible crear proyectos con la lengua catalana como lengua origen o destino.

2.3. Cren

La herramienta Cren (del catalán, Cerca i Revisió d'Enllaços), distribuida bajo licencia GNU/GPL y mediante la página web del proyecto RESTAD, revisa automáticamente los enlaces a páginas web de uno o más documentos *doc* o *html*, incluso toda una página web. Cuando un enlace está inactivo, Cren busca la nueva ubicación de la página web a la que el enlace inactivo hace referencia. La búsqueda de la nueva ubicación se realiza según dos estrategias:

- Si el enlace hace referencia a un documento con la extensión *html*, intenta acceder a este documento con la extensión *htm* o bien, si el documento referido tiene la extensión *htm*, intenta acceder con la extensión *html*.
- Si la primera estrategia no ha servido para localizar la nueva ubicación del documento, realiza una búsqueda automática de páginas web de la red que tengan como título el mismo texto anclado del enlace o bien un título que contenga este texto.

La búsqueda automática se realiza mediante llamadas a la API del buscador Google. El parámetro de entrada es la cadena de palabras a buscar, que es el texto anclado del enlace. Entre los parámetros de la respuesta extraemos el título de la página web que contiene el texto anclado; de este modo, es posible comprobar si el título es igual o contiene el texto anclado. A continuación presentamos un ejemplo de localización de la nueva ubicación de una página web siguiendo la segunda estrategia. En un documento *html* en el que aparece un listado de editoriales que publican libros en catalán aparece el enlace Alfonduco Edicions con la siguiente ubicación: <http://www.menorcaweb.net/alfonduco/>. Cren detectó que este enlace estaba inactivo, por lo que, tras fracasar la primera estrategia, buscó en la red una página web que tuviera como título “Alfonduco Edicions” y la encontró en esta ubicación: <http://www.alfonducoedicions.com/>. Teniendo en cuenta que es posible que el usuario no encuentre en esta página la información referida, Cren también busca la ubicación de páginas web cuyos títulos contienen el texto anclado. Por ejemplo, guarda <http://www.alfonducoedicions.com/poesia/> o bien <http://www.alfonducoedicions.com/conte/>, etc. como enlaces alternativos.

El resultado de la revisión es un fichero *html* en el cual, para cada enlace inactivo, se presenta la siguiente información:

- El texto anclado.
- El enlace inactivo.
- La lista de enlaces sugeridos. El usuario puede hacer clic en un enlace y acceder a la página referida para comprobar que es la página que está buscando. La lista está ordenada según el grado de coincidencia entre el texto anclado y el título de la página, siendo la primera opción la página cuyo título coincide exactamente con el texto anclado.

- El contexto en el que aparece, en el caso de que se encuentre en un párrafo. De este modo indicamos al usuario dónde se encuentra el enlace que tiene que actualizar para acceder a él de forma ágil. También sirve para elegir un enlace sugerido alternativo, en el caso de que el usuario considere que el enlace presentado como primera opción no se ajusta al contexto en el que tendría que aparecer en la versión actualizada.

Como prueba piloto se analizaron todos los enlaces de un espacio digital denominado Lletra [7], dedicado a la literatura catalana. El total de enlaces a revisar fue de 14.000, aproximadamente. Cren detectó 2.408 enlaces inactivos, de los cuales 1.967 presentó enlaces alternativos. El tiempo empleado para realizar la revisión fue de unas cuatro horas.

3. Conclusiones y Trabajo Futuro

Las tres herramientas presentadas en este artículo representan un paso adelante a la hora de facilitar la automatización de algunos de los procesos que se llevan a cabo a la hora de revisar y traducir documentos. El desarrollo de las herramientas ha partido de dos situaciones diferenciadas. Tanto en el caso de Cren como en el caso de Stem-LES, las herramientas se construyeron desde cero para satisfacer necesidades concretas del entorno de trabajo del Servicio Lingüístico; mientras que, para el desarrollo de Frog Translator, se mejoraron y ampliaron las funcionalidades de una herramienta de código libre existente. Las tres herramientas son distribuidas bajo licencias de código libre, de manera que se facilite su ampliación y distribución más allá de las fronteras del proyecto RESTAD y pueda satisfacer así las necesidades de cualquier entidad o particular que esté interesado en ellas.

De las tres herramientas, Stem-LES y Frog Translator continúan en desarrollo, mientras que CREN se da por terminado por parte del Servicio Lingüístico de la UOC. Respecto a Stem-LES, para mejorar el proceso de extracción, se integrarán filtros basados en técnicas estadísticas [8] sobre frecuencias relativas de palabras extraídas de corpus de lengua general. Y respecto a Frog Translator, se trabajará para incorporar traducción automática entre un mayor número de pares de lenguas utilizando las interfaces web de Google y BabelFish.

4. Agradecimientos

Los autores desean expresar su agradecimiento a las personas que han colaborado en el desarrollo de las herramientas aquí presentadas: Antoni Oliver González, Jordi Atserías Batalla y Albert Romero Sanchez. Asimismo, a las distintas partes implicadas en el proyecto RESTAD, que ha sido desarrollado gracias al convenio de colaboración (con número de registro 2281) entre el Departamento de Universidades,

Investigación y Sociedad de la Información (DURSI) de la Generalitat de Catalunya y la Universitat Oberta de Catalunya (UOC), la Universitat Autònoma de Barcelona (UAB), la Universitat Politècnica de Catalunya (UPC) y la Universitat de Girona (UdG).

Referencias

1. Climent, S., Moré, J., Oliver, A., Salvatierra, M., Sánchez, I., Vázquez, M.: Technologies de la traducció per a la gestió de la doble oferta docent en català i castellà de la UOC. En: Zeitschrift für Katalanistik, Vol. 18. Deutscher Katalanistenverband, Freiburg (2005) 31-57. ISSN 0932-2221
2. Proyecto RESTAD: www.uoc.edu/serveilinguistic/home/restad/restad.html
3. Ubicación de Stem-LES: <http://sourceforge.net/projects/stem-les>.
4. McKay, C.: Open Source Update 2005: A guide to free and open source software for translators. Lulu editora digital independiente. <http://www.lulu.com/content/178586>
5. Foreign Desk: <http://sourceforge.net/projects/foreigndesk/>
6. Termbase: www.fask.uni-mainz.de/user/srini/termbase.html
7. Web de Lletra: www.lletra.com
8. Peñas, A., et. al.: Corpus-based terminology extraction applied to information access. En: Proceedings of the Corpus Linguistics conference (2001), 458-465. ISBN 1 86220 107 2

El software libre en la docencia de multimedia

Carlos Casado Martínez
Antoni Marín Amatller
Roser Beneito Montagut
César Pablo Corcoles Briongos
Ferran Giménez Prado
Laura Porta Simó

Universitat Oberta de Catalunya
{ccasado, amarina, rbeneito, ccorcoles, fgimenezp, lportasi}@uoc.edu

Abstract. Este artículo pretende explorar cómo se utiliza el software libre en la docencia universitaria, concretamente en los estudios de Multimedia de la UOC.

El Graduado en Multimedia y el master en Creación y producción multimedia, ambos ubicados dentro de los estudios que ofrece la Universitat Oberta de Catalunya, en su objetivo de formar profesionales altamente cualificados y conectados con el mundo de la empresa y la industria multimedia necesitan ofrecer una formación transdisciplinar que contemple un perfil híbrido con capacidades de gestión, de producción gráfica digital y unos conocimientos técnicos sólidos. A ello cabe añadir la especificidad de los propios estudios, que viene dada por tratarse de unos estudios on-line.

La poca tradición universitaria en el área de la multimedia hace que en nuestra labor docente nos movamos en terrenos poco explorados y que se nos planteen constantemente retos provocados por el continuo desarrollo de las tecnologías de la información y la comunicación. En pocos años hemos pasado de un internet básicamente textual a una web donde la interactividad y el multimedia se han convertido en habituales. Es por ello también que cada día se hace más necesaria una formación capaz de abordar la gestión, la producción y la programación de contenidos en internet.

En este marco contextual nos interesa de un modo particular reflexionar sobre el software libre como elemento clave para conseguir que nuestros estudiantes acaben con un perfil profesional adecuado para su inserción en el mundo laboral. La utilización de software libre dentro de nuestros estudios se realiza de diferentes maneras.

- Por una parte como eje fundamental de la parte práctica de algunas asignaturas.
- En otros casos, cuando el software propietario tiene mucha aceptación en el mundo empresarial, el software libre se presenta como una alternativa válida, pero manteniendo el software propietario como la base de práctica.
- Finalmente, nos provee de herramientas de trabajo colaborativo que consideramos imprescindibles dentro del ámbito de la multimedia, dado que muchos de los proyectos a desarrollar tienen que ser realizados por un equipo de profesionales.

En resumen, haremos un recorrido por el software libre que utilizamos en nuestros estudios y un análisis de las capacidades profesionalizadoras que se abordan con su uso.

1. Introducción

El software libre va adquiriendo un papel más importante cada día que pasa en todos los ámbitos. Muestra de ello es la preocupación que desde diversos sectores se tiene sobre la utilización del mismo, y que implica a áreas de conocimiento aparentemente tan ajenas como la legislativa y la cultural, pasando por la tecnológica. Otro dato remarcable es el hecho de que un tanto por ciento muy elevado de la infraestructura en la que se basa internet es libre.

En este contexto, también la docencia implicada en las tecnologías de información y comunicación tiene en el software libre un punto de atención, como demuestra la aparición de postgrados y masteres en software libre o de foros de debate como este congreso. De este modo esta ponencia la vamos a dedicar a analizar el uso del software libre en los estudios Multimedia de la Universitat Oberta de Catalunya, es decir, en el Graduado Multimedia y el Master de Creación y Producción Multimedia.

En una primera aproximación consideramos definir multimedia como un sistema compuesto por múltiples lenguajes y que a su vez utiliza más de un modo de comunicación para la presentación de la información - texto, imagen, animación, sonido y vídeo - y, aun más, puede utilizar (utiliza) diversos medios de comunicación - internet, televisión, telefonía móvil, PDAs, etc.

La multimedia está basada en la hipermedia. Las capacidades, muchas veces potenciales, que ofrecen los sistemas hipermedia para la cultura visual hacen que sea necesaria una nueva forma de pensamiento y funcionamiento, así como un nuevo modelo operacional. Ahora, necesitamos cambios en nuestro modo de ver, de informarnos, de leer y de gestionar y producir información. En los estudios tradicionales no existe un área de conocimiento específica que se dedique al ámbito de la multimedia, con una visión transdisciplinar que contemple un perfil híbrido con capacidades de gestión, de producción audiovisual digital y unos conocimientos técnicos sólidos.

Así, estos estudios nacen ante la necesidad de formar profesionales altamente cualificados y conectados con el mundo de la empresa y la industria multimedia, con una formación multidisciplinar, y con grandes capacidades para el trabajo en grupo. Este perfil curricular estará formado básicamente a partir de la triangulación de tres capacidades: capacidad de gestión, de producción audiovisual digital y capacidad tecnológica.

Al campo de la multimedia se han ido incorporando especialistas que o bien están dentro del ámbito de la informática o nada tienen que ver con ella. Encontramos profesionales que parten de la literatura, del arte, de la música, de la ingeniería, de las ciencias, de la comunicación, que se acercan seducidos por las grandes posibilidades que ofrece la hipermedia. Pero lo complicado es encontrar especialistas que no tengan a priori un acercamiento desde otras áreas de conocimiento sino que se hayan formado en y para la multimedia, y que en su perfil incorporen competencias que pueden parecer antagónicas, como por ejemplo guionización audiovisual (más cercana a la literatura y por ello al ámbito de las humanidades), programación (informática, es decir, pertenece al ámbito de lo tecnológico), gestión de proyectos (marketing, publicidad, business, ámbito empresarial) y diseño visual.

Por ello, en primer lugar, al situarnos en relación con la empresa aparece una cuestión que nos preocupa y nos replanteamos constantemente: ¿Qué tipo de profesional queremos formar? ¿Qué herramientas utilizamos para ello? ¿Qué software

tienen que manejar los estudiantes para que una vez egresados puedan optar al mundo laboral o mejorar sus posibilidades? ¿Qué papel tiene el software libre en el perfil curricular de nuestros estudiantes?

Lo que sí tenemos claro a priori es que las herramientas a utilizar en el área de conocimiento de la multimedia tienen que permitir cubrir las necesidades que surgen de la gestión de la información, del diseño visual y de la tecnología. Y en este sentido el software libre ocupa un espacio determinante en la formación de nuestros estudiantes, con respecto a varios sentidos que analizaremos más adelante.

En segundo lugar, situándonos en el espacio académico, la universidad Oberta de Catalunya apuesta claramente por el software libre, como se desprende de las afirmaciones del Vicerrector de Tecnología de la UOC, Llorenç Valverde: "El software libre es una apuesta institucional, pero también responde a una petición de la comunidad"; y también de los proyectos relacionados con el software libre que hay en marcha, así como del hecho de ofrecer un master oficial en Software libre que pertenece a nuestros estudios, los Estudios de Informática, Multimedia y Telecomunicaciones.

2. Utilización de wiki como soporte a las asignaturas del graduado en multimedia

La docencia en entornos virtuales tiene necesidades de gestión de la información que muestran determinadas particularidades. Entre ellas cabe destacar la necesidad de contar con repositorios para esa información fácilmente administrables y actualizables. Las llamadas herramientas 'web 2.0' disponibles para tal función (blogs, wikis, herramientas 'groupware' y otros sistemas de administración de la información), muchas de las cuales son software libre, ofrecen soluciones diversas con diferentes grados de sofisticación y dificultad de uso. En el caso del Graduado en Multimedia se ha optado, en diversas asignaturas, por el uso de MediaWiki para solventar algunas de esas necesidades de gestión de la información.

Antes de entrar en mayores detalles convendría explicar brevemente qué son las aulas virtuales del entorno UOC, el espacio al que acceden las diferentes figuras del proceso educativo, que en este caso son principalmente los estudiantes y el 'consultor', el responsable del día a día de la docencia en las aulas. El aula virtual en sí es un conjunto de espacios para la comunicación, planificación, evaluación y presentación de recursos para el aprendizaje. Así, en el espacio de comunicación encontramos un 'tablón del profesor', en el que el consultor publica las informaciones convenientes para el progreso de la asignatura, un 'foro' al que tienen acceso libre tanto el consultor como los estudiantes para la interacción pública y, si se considera conveniente, espacios adicionales para celebrar debates, por ejemplo. Naturalmente, consultor y estudiantes cuentan con la opción de comunicación privada a través del correo electrónico. El espacio de planificación contiene un calendario detallado de las actividades a realizar a lo largo del semestre y el plan docente que da las pautas de funcionamiento. El espacio de evaluación contiene buzones privados a los que todos

¹ Sangrà, M. (2006). "El programari lliure entra a les aules". Mo n UOC, nu m. 25, setembre de 2006. http://www.uoc.edu/prensa/entrevistas/masters_oficiales.html (última consulta 18/02/07)

² http://www.uoc.edu/masters/oficials/estudis/master_oficial_programari_lliuere_estudis.htm

los estudiantes pueden enviar sus actividades evaluables y al que sólo el consultor tiene acceso. Finalmente, en el espacio de recursos el estudiante cuenta con enlaces a los materiales didácticos de la asignatura y material de soporte (estos enlaces se determinan al inicio de semestre), una biblioteca virtual y un espacio de disco compartido para todos los miembros del aula en el que se pueden compartir todo tipo de archivos.

Con el uso de un wiki damos a esos agentes la capacidad, más o menos libre, de utilizar una herramienta colaborativa de publicación y edición de contenidos. El espacio del wiki, además, al estar separado de los espacios convencionales del aula, trasciende los límites temporales del semestre, y la información que se acumula en él se conserva entre semestres, algo que no pasa con los espacios de comunicación, que se clausuran a la finalización de cada curso.

Debe notarse que el uso de una herramienta de wiki no supone necesariamente que se suscriba una filosofía de libre edición de contenidos sin control. Veremos a continuación que el uso de la aplicación y los niveles de acceso permitidos a los usuarios son diferentes dependiendo de los objetivos de cada asignatura en que se usa.

El consultor de la asignatura de Redes y comunicaciones informáticas reúne a lo largo del semestre numerosos artículos, enlaces e informaciones varias relacionadas con el temario y que considera interesante que los estudiantes lean. Como ya hemos comentado, la información publicada en los foros no se traslada de aula en aula al final de cada semestre, por lo que el material recopilado un semestre no se usaba en el siguiente, al menos sin un incómodo trabajo de recopilación previo, con lo que los nuevos estudiantes perdían parte de información que habían recibido los anteriores. Para evitar esta pérdida de información se decidió crear un wiki donde el consultor pudiese ir haciendo una recopilación de enlaces interesantes, de manera que pudiesen persistir a lo largo del tiempo hasta quedar anticuados. El wiki se organizó de manera que los enlaces a noticias o artículos que por su contenido habían dejado de tener interés (o este se había reducido mucho), se pudiesen mantener en un espacio aparte. Además, se organizó la información por temas, diferenciando enlaces a webs de enlaces a artículos o blogs, se pusieron resúmenes de aquello que iba a aparecer tras un enlace,... En general se consiguió un repositorio de enlaces de interés, organizado, con una vigencia superior a la del semestre docente sin representar un exceso de trabajo para el consultor.

Además, se invitó a los estudiantes a participar, abriendo un espacio para que ellos mismos pudiesen añadir enlaces interesantes al wiki. El resultado es que, usando un conjunto de herramientas libres (mediawiki³ sobre LAMP⁴) se ha conseguido que estudiantes y consultor participen conjuntamente en un repositorio de contenidos interesantes para la asignatura.

En la asignatura de Realidad Virtual también se había decidido utilizar el wiki, pero de manera diferente. El objetivo del wiki es doble: por un lado se pretende que, como en la asignatura de Redes, sea un repositorio de información relacionada con la asignatura y el campo de conocimiento asociado, pero también de la información oficial de la asignatura, y por el otro que sirva de espacio de publicación de los trabajos prácticos de los estudiantes, dado que en la filosofía de la asignatura el hecho de compartir el código generado entre iguales es esencial. Pero esa filosofía debe funcionar en paralelo con los requisitos de un repositorio oficial y que, por tanto, sólo

³ <http://www.mediawiki.org/wiki/MediaWiki>

⁴ LAMP es Linux-Apache-MySQL-PHP (o Perl o Python) - <http://es.wikipedia.org/wiki/LAMP>

debe poder modificar el consultor de la asignatura u otro personal autorizado. para ello se decidió usar el sistema de autenticación de usuarios de Mediawiki. Dado que gestionar las altas y bajas de estudiantes otorgando a cada uno ellos permisos diferentes sería excesivamente engorroso, se optó por una solución de compromiso que permitiese el doble funcionamiento con el mínimo esfuerzo administrativo: el usuario por defecto no tiene acceso de edición a ninguna página aparte de su propia página de usuario, que le es suficiente para la publicación de sus contenidos, mientras que el usuario correspondiente al consultor de la asignatura tiene permisos de administrador y tiene acceso a toda la estructura del wiki (disponiendo, además, de la potestad de aumentar los privilegios de estudiantes específicos a discreción, como es natural).

3. Utilización de la plataforma lamp para la asignatura de bases de datos del graduado multimedia

En la asignatura de Bases de datos multimedia los estudiantes realizan dos prácticas que consisten en la creación de pequeñas aplicaciones web que hacen uso de bases de datos para acceder a contenidos multimedia.

El objetivo de la asignatura es que los estudiantes aprendan a usar bases de datos y a crear aplicaciones dinámicas, para lo que existen en el mercado numerosas alternativas. Al diseñar la asignatura se planteó con que herramientas se realizaban las prácticas y se tuvieron en cuenta las de Microsoft (entorno de desarrollo .NET), diferentes gestores de bases de datos y diferentes lenguajes de programación. Finalmente se seleccionó la plataforma LAMP por varios motivos:

- Está ampliamente extendida y reconocida. Si bien hay algunas críticas en cuanto a sus posibilidades de escalabilidad y a su seguridad, para hacer aplicaciones no excesivamente complejas resulta posiblemente la mejor elección.
- Es fácilmente instalable en un entorno local para la realización de pruebas. Además, teniendo en cuenta que muchos estudiantes usan MS Windows, no hay ningún problema en instalar Apache-MySQL-PHP bajo dicho sistema operativo.
- Es gratuito i de libre distribución, con lo cual puede distribuirse a los estudiantes una copia sin problemas.
- Existen soluciones integradas que permiten, con una instalación única, instalar todos los componentes (menos Linux, aunque, instalando Linux, puedes instalar todos los componentes directamente).

El paso del tiempo ha demostrado lo acertado de esta decisión. Se ofreció a los estudiantes la posibilidad de hacer un curso de un mes en el que se usaba la plataforma .NET y el resultado fue descorazonador por las dificultades que encontraron para instalar el entorno en local, o para encontrar un servidor gratuito que les permitiera usarlo a aquellos que no fueron capaces de instalarlo en su equipo.

4. Software libre para el tratamiento gráfico

4.1. Utilización de gimp como software complementario a la asignatura de fundamentos de imagen y fotografía digital

La asignatura de Fundamentos de fotografía e imagen digital es de libre elección dentro del plan de estudios del Graduado. Presenta un recorrido por la realización, edición y publicación de fotografías enfocado a la mejora de competencias del profesional del multimedia en relación a la fotografía digital. Este enfoque implica que se potencian los aspectos del trabajo del fotógrafo que más se relacionan con trabajos para soportes electrónicos (web, televisión, soportes ópticos,...) y no se abordan con el mismo peso específicos aspectos como la realización fotográfica para impresión o para trabajos que requieran plató o infraestructuras técnicas complejas. También es de señalar que el hecho que se trate de unos estudios totalmente virtuales lleva a potenciar los aspectos que como la edición y retoque son perfectamente compatibles con la virtualidad.

Dentro de estos parámetros es de destacar la importancia de aproximar los procedimientos de trabajo de la asignatura a los propios del entorno profesional. En este sentido es claro que el software que más amplia difusión y penetración cuenta en los entornos de trabajo es Photoshop. Éste es pues el programa básico de trabajo en nuestros estudios y pretendemos que el estudiante adquiera amplias competencias en su manejo.

Sentada esta prioridad se planteó hace ya algunos semestres el uso del software libre como elemento complementario en la asignatura y en este sentido se optó por Gimp. Un primer objetivo en el que se basa su uso responde al hecho de evitar una relación unilateral entre un procedimiento y un software en concreto. Por poner un ejemplo, comentar que si únicamente de trabajan los procesos de edición con un programa concreto (Photoshop en el caso que nos ocupa) el estudiante puede llegar a la conclusión errónea que procedimientos como el ajuste por niveles, por curvas o las máscaras de capa son exclusivos de la opción con la que trabaja cuando en realidad se trata de procedimientos comunes a la gran mayoría de softwares de edición. Observar los paralelismos entre Photoshop y Gimp, por ejemplo, induce a pensar en los procedimientos de trabajo en una forma más abstracta o generalista, desligada en fin de un programa concreto y asociada al retoque o edición digital en general.

Pero éste objetivo podría conseguirse también trabajando con diversas alternativas de software propietario. La utilización del software libre permite aquí un valor añadido que también interesa destacar. A la potencia de un programa como Gimp se une la filosofía del desarrollo colaborativo, del trabajo en común. En la macroesfera de la colaboración en la web, Gimp es resultado de la suma de muchas individualidades y en este sentido, animar a los estudiantes a participar en el proyecto refuerza este espíritu de trabajo en común. Por otra parte en el propio planteamiento de las asignaturas se contempla siempre una actividad comunitaria cuya razón de ser pedagógica pasa por los planteamientos que entroncan con la idea que una de las mejores vías para lograr un aprendizaje significativo pasa por la socialización de los

procesos de aprendizaje. En este sentido la filosofía de Gimp entronca perfectamente con el planteamiento.

Entrando en el terreno de la publicación de las fotografías en el aula, algo imprescindible si se pretende socializar el aprendizaje, se pasa a un segundo plano en la utilización de programas de software libre. Aquí, ya sin alternativa mejor en el terreno del software propietario, se plantea el uso de galerías fotográficas que elaboran cada uno de los estudiantes y mediante las cuáles comparten sus trabajos en el aula. Pese a que lo situamos en este apartado, por estar ligado a las asignaturas de tratamiento de imagen, estas son herramientas de gestión de la información. En este sentido, se crea un espacio al que tienen acceso los integrantes del aula en el que la actividad principal es la de publicación de prácticas e imágenes y la publicación asimismo de comentarios de comentarios y críticas, tanto por parte del consultor como del resto de los compañeros. El software utilizado es Gallery.

Finalmente un último aspecto a destacar. Anteriormente se ha comentado que en el entorno profesional de trabajo primaba el uso de software propietario como Photoshop. Si bien esto es cierto en la gran mayoría de empresas de producción audiovisual, se está abriendo también una nueva realidad laboral en la que la opción del software libre incrementa su presencia. Así son diversas las instituciones públicas que apuestan por alternativas de software libre como Gimp, tanto en campañas de la llamada alfabetización digital de sectores de la población que corren el peligro de quedar fuera de las TIC, como en la realización de las propias publicaciones. En este posible marco laboral, es necesario también contemplar de forma adecuada la formación de los estudiantes del Graduado.

4.2. Utilización de Blender como software complementario a la asignatura de animación 2D y 3D

De un modo similar a lo que ocurre con el binomio Photoshop-Gimp, en el constituido por 3D Studio MAX y Blender existe uno como programa ampliamente reconocido en el entorno profesional, y otro que se sitúa como elemento destacado en el conjunto de aplicaciones de software libre en la producción de gráficos 3D y en la animación por ordenador. Los criterios de uso en las asignaturas son también aquí similares. Mientras uno constituye un paradigma para la capacitación profesional de los estudiantes, el otro permite desarrollar, en base a un uso paralelo, estrategias educativas que posibiliten centrar la atención del estudiante en los procesos fundamentales de la animación. Ver que éstos no dependen tanto de un programa en concreto sino que por el contrario están presentes en los diversos software disponibles en el mercado. También por otra parte, la participación de los estudiantes en comunidades abiertas y en foros de discusión, algo que el entorno de Blender facilita, fomenta el espíritu del trabajo colaborativo y posibilita diseñar intervenciones pedagógicas en este sentido.

En el caso de Blender, el trabajo con el programa se lleva a cabo en talleres puntuales que se plantean en los meses de febrero y setiembre, periodos en los que la actividad en las aulas es inexistente. La razón del trabajo en talleres se encuentra motivada por la alta carga lectiva que presenta la asignatura de Animación 2D y 3D.

4.3. Utilización de Jahshaka como software complementario para la edición de video en el master de creación y producción multimedia

El Master se plantea, con respecto al Graduado en Multimedia, con un enfoque más conceptual de los contenidos. No tiene unos objetivos profesionalizadores ligados a programas de software comercial como en el grado y por tanto el enfoque práctico en este caso opta más claramente por programas de software libre.

Aparte del uso de programas ya descritos anteriormente como Gimp o Blender, actualmente se está planteando la introducción de Jahshaka como editor y compositor de vídeo. Inicialmente se había trabajado con programas gratuitos pero propietarios - Avid Free DV - pero en la actualidad se considera necesario optar por un programa como Jahshaka.

5. Conclusiones

Tras este análisis podemos concluir diciendo que en nuestros estudios el software libre no se utiliza por unas razones únicas y homogeneizantes, sino que la elección del mismo está en función de unas competencias a adquirir y en relación a contenidos concretos de cada una de las asignaturas.

En un primer momento hemos constatado como el software libre nos ofrece muy buenas herramientas de gestión de la información, que permiten el trabajo en grupo, el almacenamiento de la información y su reutilización. Nosotros hemos optado por MediaWiki para la gestión de información y Gallery cuando la información es visual o gráfica.

También constatamos que en algunos casos el software libre es una alternativa competitiva al software de pago. En el caso de Apache - MySQL - PHP y de cara a la educación, es incluso mejor por la sencillez a la hora de instalar las diferentes aplicaciones y la posibilidad de usarlas sobre diferentes sistemas operativos, a diferencia de la mayoría de entornos de software propietario que requieren un determinado SO para funcionar.

Para la educación on-line es muy importante disponer de herramientas que nos permitan compartir información, trabajar en grupo y compartir ideas, por ello estas aplicaciones son de gran ayuda para reforzar el aprendizaje del aula virtual. Aun así, si atendemos al desarrollo de este tipo de recursos, entendidos como recursos

democráticos y como cultura abierta⁵, nos sirven para la enseñanza-aprendizaje de contenidos transversales.

Por otro lado, en muchos casos, el hecho de que el software libre permita compartir el programa y sus mejoras con más gente, y mejorarlo adaptándolo a necesidades específicas⁶ hace que el trabajo sea mucho más creativo. Es decir, consideramos que el software libre potencia la creatividad en un sentido más amplio. Trabajar para producir en un lugar que al mismo tiempo permite ser modificado, (re)creado y manipulado, nos parece una opción con un nivel superior de creatividad.

Para terminar hay un último aspecto que cabe destacar. Ya hemos comentado desde un principio que en el entorno profesional y empresarial, se prima el uso del software propietario. Para constatar esto sólo tenemos que mirar los requisitos que se solicitan en las ofertas de trabajo: prácticamente siempre se pide el control de programas propietarios. Aun así, si esta es la situación real, creemos que la gran mayoría de empresas de producción audiovisual, se está abriendo también a la alternativa que ofrece el software libre, que incrementa su presencia día a día. También es una alternativa más que considerable para aquellos productores freelance, que no disponen de presupuestos tan fuertes para estar actualizados en software propietario. También son diversas las instituciones públicas que últimamente están apostando por el software libre para diferentes proyectos de gestión, de difusión y de alfabetización digital. Por ello, aparece un marco laboral, para el que es necesario tener a personas formadas.

En definitiva, consideramos necesario para nuestros estudios combinar el software libre con el software de autor, por las cuestiones que hemos analizado y que tienen más que ver con la realidad del mundo laboral y de la demanda empresarial que con planteamientos ideológicos, que nos harían optar, sin lugar a dudas, por el software libre. Lo que sí afirmamos sin reparos es que el software libre en el ámbito de la multimedia es una alternativa válida al software propietario.

⁵ "...recursos democràtics com els wiki o les llistes de correu obertes. L'enviament de propostes és lliure. A Espanya el primer hackmeeting es va celebrar a Barcelona l'any 2000, i després de diferents destinacions aquest any s'ha celebrat la seva setena edició a Mataró (Barcelona) amb el títol "Experimenta la teva llibertat". L'última frase del manifest 120 del Hackmeeting 2006 il·lustra a la perfecció l'essència d'aquestes trobades " ...no consentis que el capitalisme s'apropriï i et vengui el que sempre ens va pertanyer, la CULTURA i la TECNOLOGIA... ". Els hacklabs han dut a terme projectes directament vinculats a la promoció de la cultura lliure com per exemple els copisteris 121 (centres de còpia de material copyleft), el desenvolupament del sistema operatiu X-evian (basat en Debian - GNU/Linux) o Alephandria 122." Manuel Castells Oliván; Meritxell Roca, "El software lliure a Catalunya i a Espanya. Informe d'investigació". http://portal.uoc.edu/west/servlet/west.servlets publica.Documento?id_proyecto=1214&id_idioma=a&id_item=1215 (última consulta 18/02/07)

⁶ "... la forma más simple de hacer que un programa sea libre es ponerlo en el dominio público, sin derechos reservados. Esto permite compartir el programa y sus mejoras con la gente, si así lo desean; pero permite a gente no cooperativa convertir el programa en software privativo... copyleft dice que cualquiera que redistribuye el software, con o sin cambios, debe dar la libertad de copiarlo y modificarlo más. Copyleft garantiza que cada usuario tiene libertad...". Free Software Foundation - GPL, <http://www.gnu.org/copyleft/copyleft.es.html> (última visita 15/02/07)

6. Referencias

1. Entrevista a los directores de los tres máteres oficiales de la UOC. Publicada en el boletí n de la UOC, Wok!, nú m. 12, septiembre de 2006.
http://www.uoc.edu/prensa/entrevistas/masters_oficiales.html
2. Roca Sales, M. y Castells, M. (2007) Drets de propietat intel•lectual i Internet a Espanya. Materials per a un debat informat. IN3, UOC. Barcelona.
3. Sangrà, M. (2006). "El programari lliure entra a les aules". Mo n UOC, nú m. 25, setembre de 2006
4. Universitat Oberta de Catalunya. <http://www.uoc.edu>
5. <http://es.wikipedia.org>

Experiencias en el Desarrollo de una Taxonomía de Estándares Abiertos

Gerardo Aburruzaga García, Alejandro Álvarez Ayllón, Antonio García Domínguez, J. Carlos González Cerezo, Manuel Palomo Duarte y J. Rafael Rodríguez Galván

Oficina del Software Libre, Universidad de Cádiz
{gerardo.aburruzaga}@uca.es
<http://softwarelibre.uca.es>

Resumen A lo largo de los últimos años, han ido surgiendo más y más estándares para cubrir la necesidad de interoperabilidad entre diversas organizaciones a lo largo del tiempo, evitando tecnologías cerradas que limiten las opciones de éstas organizaciones.

Sin embargo, no basta simplemente con que el estándar sea público para que dicha tecnología se halle abierta a todos: se requieren estándares que sean libremente implementables y utilizables: estándares

En este trabajo, expondremos la labor realizada y la casuística generada en la OSLUCA entro de su participación en la elaboración de un Documento de Interoperabilidad para la Junta de Andalucía. Esta participación ha consistido básicamente en la elaboración de una taxonomía en la que se han clasificado varios cientos de especificaciones, para cada una de los cuales se han estudiado cinco criterios relacionados con la posible clasificación como estándar abierto de cada una de estas especificaciones. Basándose en la taxonomía realizadas y en una serie de reglas previamente definidas en función de las necesidades de la administración pública, fue posible clasificar a cada una de estas especificaciones.

Palabras clave: Estándares Abiertos, Taxonomía, Formatos, Especificaciones, Clasificación

1. Introducción

La Oficina de Software Libre de la Universidad de Cádiz (UCA) ha participado, en colaboración con un equipo de trabajo externo a la UCA, en la elaboración de un Marco de Interoperabilidad para la Junta de Andalucía. El hecho de que las administraciones públicas utilicen estándares abiertos para operar internamente y en sus relaciones con ciudadanos, empresas y otras instituciones tiene una importancia vital, como se comentará en este apartado. Con posterioridad (sección 3), analizaremos en detalle la experiencia llevada a cabo en la Universidad de Cádiz.

1.1. Definición de estándar

En los últimos años, el continuo avance de las nuevas tecnologías ha hecho que el tratamiento digital de la información se haya convertido en una realidad indiscutible a cualquier nivel, ya sea en el almacenamiento como en la transmisión.

Esto se ha hecho especialmente significativo con el auge y extensiva implantación de Internet, con lo que el tráfico de datos y de información ha aumentado espectacularmente.

Pero esta expansión no ha afectado sólo al sector privado. La Administración Pública también ha realizado una progresiva adaptación a los medios electrónicos. Ejemplo de ello es la Ley de Administración Electrónica, con la que el gobierno español permite a realizar cualquier trámite de forma electrónica.

Obviamente para posibilitar el entendimiento de todas las partes interesadas es necesario establecer unas normas comunes o estándares que determinen la forma de interactuar o de transmitir la información.

La definición del Diccionario de la Real Academia Española es bastante escueta, limitándose al aspecto de un estándar como una *especificación* de algo:

Que sirve como tipo, modelo, norma, patrón o referencia.

Esta definición se puede completar con la clasificación de [1], que distingue, desde un punto de vista técnico, entre dos tipos de estándares (que se comentarán en los siguientes apartados) para aclarar la diferencia entre una simple *especificación* y un verdadero *estándar*.

En este trabajo enlazaremos el concepto de estándares abiertos con el del software libre, que la existencia de implementaciones de referencia de dichos estándares abiertos que tengan licencia libre garantiza la publicidad de la especificación, siendo convenientes el uno para el otro.

Estándar de facto Un estándar de facto, como indica el nombre, no es un estándar auténtico, sino simplemente una *especificación*, que frecuentemente no es pública y que es ampliamente utilizada y que es propiedad de una empresa.

En el caso de que ésta sea pública, las implementaciones de terceras partes habrán de revisarse cada vez que el propietario de dicha especificación decida cambiarla, sin poder participar en dicho proceso de cambio.

Pero si la especificación no es pública, la situación es mucho más desfavorable, la infraestructura e información de sus clientes estarán atados a los productos de dicha empresa, disminuyendo su libertad e impidiendo la libre competencia.

Estándar de comunidad Los estándares de comunidad son creados por más de una entidad, ya sean personas físicas o jurídicas, y con frecuencia pueden considerarse realmente estándares, aunque su validez dependerá del proceso exacto seguido. El caso ideal es aquel en que, de forma transparente y democrática, se formen a partir del consenso de estas entidades, permitiéndoles tratar el problema en cuestión desde un mayor número de puntos de vista e intereses.

Entre estos estándares, se encuentran los estándares *de jure*, aprobados o desarrollados por una organización con competencias especiales para ello. Estas organizaciones, como la ISO (International Standards Organization), el ANSI (American National Standards Institute) o la ITU (International Telecommunications Union), disfrutan de cierta credibilidad y fiabilidad, siendo en algunos países muy difícil el uso de estándares que no hayan tenido el visto bueno de estas organizaciones.

Otros grupos, como el W3C (World Wide Web Consortium) u OASIS (Organization for the Advancement of Structured Information Standards), forman consorcios y, a veces, envían los estándares que desarrollan a organizaciones formales de estandarización como las anteriores para que sean aprobados.

Estándares abiertos Un *estándar abierto* consiste en una serie de especificaciones públicas cuya propiedad intelectual se ofrece de forma libre de regalías y sin restricciones en cuanto a su uso y reutilización. Idealmente, el estándar es mantenido por una entidad neutral y sin ánimo de lucro y cuenta con implementaciones de referencia con licencia libre.

La definición más conocida es la propuesta por la comisión IDABC de la Comunidad Europea [3]:

“Las siguientes son las mínimas características que una especificación y sus documentos de apoyo deben tener para ser denominados estándares abiertos:

1. El estándar ha sido adoptado y es mantenido por una entidad sin ánimo de lucro, y su sucesivo desarrollo tiene lugar sobre la base de un proceso de decisión abierto a todas las partes interesadas (consenso o decisión por mayoría, etc.).
2. El estándar se ha publicado y el documento con la especificación del mismo se encuentra disponible de forma gratuita o bien por un precio simbólico. Se debe permitir a cualquiera su copia, distribución y uso sin cargo o con un precio simbólico;
3. La propiedad intelectual -por ejemplo, posibles patentes presentes del estándar (o de alguna de sus partes) se ofrece de forma irrevocable libre de regalías (royalty-free basis);
4. No hay restricciones en cuanto a la reutilización del estándar.”

Esta definición es la base de los criterios que se han estudiado (sección 3.3) en la clasificación taxonómica de estándares llevada a cabo en la Oficina de Software Libre de la Universidad de Cádiz en el año 2006 contrastaremos los criterios seguidos en este proyecto tomando como base la descripción que aquí nos ocupa con los de otros autores.

1.2. Impacto de los estándares abiertos

El uso de estándares abiertos conlleva un impacto positivo en los sectores privado y público, a través de la explotación de las características mencionadas anteriormente (que se reflejan en la definición de la Unión Europea) y de la existencia de implementaciones de referencia con licencia libre.

Impacto en el sector privado Consideramos, en primer lugar, que la innovación y la libre competencia son los dos mecanismos a través de los cuales los consumidores salen beneficiados, ya obtengan bienes finales o productos intermedios, y la economía puede mantener un desarrollo constante.

Los estándares abiertos protegen la innovación y la libre competencia ofreciendo especificaciones abiertas y con mínima o ninguna restricción abiertas con el del software a su uso. De la misma forma, los derechos de los consumidores a cambiar a la solución que mejor se ajuste a sus necesidades se pueden garantizar a través de un estándar de comunidad obtenido mediante un proceso abierto basado en el consenso entre todas las partes interesadas, que no dependa únicamente de los intereses de una organización específica.

Contratos leoninos y altos costes (a veces ni siquiera previsibles) derivados de la propiedad intelectual limitan la posibilidad de innovar a las grandes compañías con amplios portafolios de patentes e impiden el verdadero aprovechamiento de las posibilidades que ofrecen las tecnologías de hoy en día.

Adicionalmente, la existencia de implementaciones de referencia con licencia libre implican la reducción de costes de desarrollo de soluciones basadas en estos estándares y permiten agilizar su adopción, manteniendo niveles considerables de calidad a través de la colaboración abierta entre todos los desarrolladores de diversas compañías interesadas en dicha tecnología. Cualquier nueva iniciativa basada en dicho estándar puede simplemente reutilizar componentes ya probados para su nueva solución, posiblemente mejorándolos según sus necesidades.

Por supuesto, este tipo de implementaciones cuenta con todos los puntos fuerte del software libre: nuevos modelos de negocio, mantenimiento constante e indefinido (no necesariamente por los desarrolladores originales), personalización, etc.

Un ejemplo paradigmático de la estrecha relación entre estándares abiertos e implementaciones de referencia es la propia Internet, que se halla basada en su mayor parte por software libre: servidores web (Apache), de correo (Sendmail), DNS (BIND), sistemas operativos de servidor (GNU/Linux), etc.

Sin embargo, no es posible considerar el aprovechamiento de software libre como base para una solución si el estándar en que se basa no es abierto, o si el software no se halla protegido ante un posible “cierre” a un modelo propietario a través del uso de una licencia con cláusulas de mantenimiento de un copyleft fuerte, en que todo añadido o modificación a la base inicial debe ser también publicado.

Impacto en el sector público El uso de estándares abiertos, independientemente de que se realice bajo software libre o no, impide la aparición de prácticas monopolísticas como las que se han ido sucediendo últimamente por algunos de los líderes de la industria, evitando encerrar a cualquier organización en las soluciones de un vendedor en particular.

Esto es tanto más importante para una organización de la escala y complejidad de un gobierno, donde la flexibilidad y autonomía son algo tremendamente importantes: resulta mucho más factible crear un “sistema de sistemas” integra-

dos de forma flexible entre sí que un único sistema monolítico que actúe como una caja negra. Tampoco debería forzarse a los ciudadanos a usar tecnologías cerradas que restrinjan sus libertades. Y se debe evitar que un gobierno dependa de la voluntad del vendedor para mantener su producto una vez deje de ser rentable.

Además, para un gobierno, en que es necesario el almacenamiento a muy largo plazo de grandes cantidades de información (registros médicos o policiales, por ejemplo), el uso de formatos abiertos permite su uso independientemente de si el implementador original sigue dando soporte para dicho formato o no.

Y por supuesto, como en el sector privado, la existencia de implementaciones de referencia libres conllevan una serie de ventajas: entre las cuales no es despreciable la reducción del coste en licencias, permitiendo redistribuir mejor el presupuesto público en mayores cantidades de hardware y soporte técnico.

Varios países y organizaciones han realizado estudios previos sobre interoperabilidad, por lo que, para garantizar la fiabilidad, se han tomado tres de ellos como base de nuestro estudio.

2. Iniciativas previas

Diversos países y organizaciones han realizado estudios previos sobre interoperabilidad, que han sido analizados en detalle por parte del equipo de trabajo externo a la universidad de Cádiz y empleados para garantizar la fiabilidad de nuestro estudio. Entre ellos, se encuentran los siguientes:

eGIF El *e-Government Interoperability Framework* [4] es el más antiguo de la unión europea, previo incluso al Marco de Interoperabilidad Europeo [3]. Además, es de obligado cumplimiento para cualquier organismo del gobierno británico.

La selección de especificaciones se hace en función de la interoperabilidad, la extensión en el mercado, la escalabilidad, apertura y estandarización internacional.

La última revisión data del año 2005.

SAGA El documento SAGA [5] (traducido al español como *Estándares y Arquitecturas para Aplicaciones de Gobierno Electrónico*) se encuadra dentro de la iniciativa de gobierno electrónico del gobierno alemán. Además de los criterios del eGIF añade como variables a tener en cuenta el coste, la flexibilidad, el rendimiento, tolerancia a errores y vigencia, entre otros.

En función de estos parámetros divide las especificaciones en obligatorias, recomendadas y bajo observación. Por otra parte las agrupa en tres listas: blanca, estándares aún por clasificar; gris, recomendados en el pasado pero que han quedado excluidos; y negra, donde se encuentran los obsoletos o rechazados.

ePING Por último, mencionaremos el ePING [6], marco de interoperabilidad del gobierno brasileño. Establece las normas para compartir datos digitales y las especificaciones técnicas a emplear en la comunicación entre sistemas y entre diferentes organismos gubernamentales.

3. Trabajo Realizado

Como se ha comentado, en el año 2006, un equipo de trabajo de seis personas participó, en el seno de la Oficina de Software Libre de la Universidad de Cádiz, en la elaboración de un Marco de Interoperabilidad para la Junta de Andalucía, en colaboración con un equipo de trabajo formado por personas de otras universidades andaluzas y de otras entidades.

El fruto de este trabajo es un Documento de Interoperabilidad para la Junta de Andalucía que cuenta con una clasificación taxonómica de varios cientos de especificaciones, para cada una de las cuales se han estudiado cinco criterios relacionados con la posible clasificación como estándar abierto de cada una de estas especificaciones, y que ha sido desarrollado por el equipo de la UCA.

3.1. Objetivos, herramientas y procesos

El objetivo general que fue planteado era la elaboración de una taxonomía que debía consistir en el estudio de un gran número de especificaciones, protocolos e interfaces, investigando para cada uno de ellos su posible cumplimiento de cada uno de los cinco criterios que se detallan en la sección 3.3. La intención era el contar con una base de datos objetivos lo suficientemente amplia como para que pudiera ser utilizada con posterioridad para la clasificación de estas especificaciones (o al menos de todas aquellas que fueran necesarias para la administración pública objeto del estudio) dentro de una serie de estados que definieran el ciclo de vida de especificaciones y que determinaran su posible recomendación como estándar para la interoperabilidad.

Para ello, se optó por una dinámica de trabajo abierta, elaborando una serie de tablas en un wiki alojado en los servidores web de la Universidad de Cádiz. El utilizar este tipo de herramientas permitía una orientación del trabajo muy dinámica, reforzando los procesos colaborativos y primando la libertad de ubicación, la revisión cruzada y otras estrategias motivadoras.

Para ello, se dividió el trabajo en diversas tablas, agrupadas en distintos niveles:

3.2. Niveles de interoperabilidad

Los formatos y especificaciones estudiadas estaban clasificados en diversos niveles basados en el *Marco de Interoperabilidad Europeo*[3], guía que recomienda la Unión Europea que se emplee como base para el resto de marcos de interoperabilidad realizados por poderes públicos.

Este marco divide el problema en cuatro dominios conceptuales:

Interoperabilidad básica La denominada interoperabilidad básica constituye el nivel primario que asegura una capacidad mínima de intercambio de información entre las entidades implicadas. Este nivel de interoperabilidad presupone que el formato de los mensajes admitidos así como el significado de los mismos es idéntico entre las administraciones.

Interoperabilidad técnica El siguiente nivel de interoperabilidad, técnica, define mecanismos de adaptación sencilla entre procesos administrativos que, de otro modo, serían incompatibles entre sí.

A modo de ejemplo podría considerarse un servicio simple de traducción de términos entre los idiomas oficiales de la Unión Europea que permita traducir el término “licencia de conducir” por sus correspondientes en otros idiomas oficiales como “driving license” en Inglés.

Interoperabilidad semántica El nivel anterior define mecanismos simples de adaptación de la información entre participantes, no obstante existen escenarios más complejos que requieren conocer el contexto general en el que se produce el intercambio de información. Dicho contexto se conoce como la semántica asociada al intercambio de la información.

A modo de ejemplo para el caso propuesto anteriormente, un servicio simple de traducción de términos no podría resolver la ambigüedad existente entre las distintas categorías de conductores en función del país de emisión de la licencia de conducir. Es necesario disponer del contexto, en este caso país de emisión del permiso y país en el cual se utiliza, para resolver el tipo de vehículo que un conductor puede conducir.

Interoperabilidad organizativa El nivel último de interoperabilidad analiza como llevar a cabo la coordinación de procesos administrativos de alto nivel como la realización de transacciones financieras entre administraciones independientes, cumplimiento de procedimientos administrativos, etc.

Así mismo cada nivel se subdivide en distintas tecnologías según casos de uso específicos, como codificación de video, codificación de audio, escritura de documentos de texto, etc.

A continuación, dentro de cada tecnología, se estudiaron diferentes especificaciones y evaluaron el cumplimiento o no de los criterios planteados (y que detallaremos en la siguiente sección), para poder así aplicar la máquina de estados del punto 3.4 y determinar así si pueden ser recomendados realmente como *estándares abiertos*.

3.3. Criterios estudiados

Ideas previas Bruce Perens [7] define seis principios detrás de los estándares abiertos, y recomienda seis clases de prácticas para conseguir que los estándares sean abiertos:

Disponibilidad El caso ideal es que las especificaciones estuvieran disponibles de forma gratuita por Internet, o en su defecto a un coste simbólico, como el de un libro de texto universitario.

Además, su uso no debería de imponer restricciones ni sobre las licencias, con la precaución de protegerse de prácticas dañinas al estándar como la creación de extensiones propietarias cerradas, que puedan inutilizar el estándar.

Maximizar la posibilidad de elección Deben de permitirse tanto implementaciones a nivel comercial, académico como gubernamental, y en cualquier rango de precios, desde gratuito a más o menos caros.

Libre de regalías La implementación debe ser gratuita, sin regalías ni tasas algunas. La certificación puede implicar algún coste. Así, las patentes relacionadas con un estándar deben de estar acompañadas con licencias no discriminatorias libres de regalías, y deben de haber programas gratuitos de autocertificación, opcionalmente extendidos por programas más avanzados, posiblemente bajo coste adicional.

No discriminación No se debe favorecer a una implementación si no es por una mejor conformidad con el estándar. Esto implica que haya dos caminos para certificación: uno básico con coste cero, y posiblemente uno extendido con cierto coste adicional, a un nivel superior.

Extensiones o subconjuntos Una implementación de un estándar abierto puede ser extendida u ofrecerse sólo un subconjunto, imponiéndose restricciones sobre dichas extensiones (ver “Prácticas depredadoras”). Algunas organizaciones prohíben implementaciones no completas del estándar.

Prácticas depredadoras Se pueden emplear términos en la licencia que obliguen a crear referencias públicas y dar licencias gratuitas y no discriminatorias de propiedad intelectual sobre extensiones creadas, de forma que sea posible mantener la compatibilidad.

Ken Krechmer [2] considera unas condiciones que se ajustan a los seis principios de Perens, aunque añade dos más:

Un mundo Originalmente *One world*: Un estándar por cada necesidad para todo el mundo, sin diferencias.

Ayuda continua *On-going Support* en el original inglés: el estándar debe seguir activo hasta que el interés de los usuarios desaparezca, y no cuando disminuya el de los desarrolladores.

Criterios seguidos Teniendo en cuenta las ideas anteriores, así como la definición de la Unión Europea[3] y los intereses de la administración pública española, se se han estudiado cinco puntos que podrían ser de especial interés para clasificar cada una de las especificaciones:

1. Es adoptado y mantenido por una entidad sin ánimo de lucro abierta a todas las partes interesadas.
2. Está publicado y su especificación y documentación completas están disponibles de forma gratuita o a precio simbólico.
3. Su propiedad intelectual se ofrece de forma irrevocable libre de regalías y no está sujeta a patentes o contratos que restrinjan su uso y reutilización directa o indirectamente.

4. a) Existe al menos una implementación de referencia que implementa todas las funcionalidades de la especificación disponible para ser usada en cualquier propósito, y que puede ser copiada, estudiada, mejorada y distribuida libremente.
- b) Es posible realizar modificaciones y pueden ser redistribuidas siempre que se haga en los mismos términos y con las mismas condiciones expuestas. (Fuerza el copyleft)

Los puntos 2 y 3 se ajustan al primer y tercer punto de las condiciones impuestas por Perens, mientras que el criterio 4.b guarda relación con el sexto, ya que impide que se creen extensiones cerradas que dificulten la interoperabilidad.

Aplicación práctica de los criterios En la sección 3.3 se han introducido los criterios analizados para cada especificación. Por desgracia la teoría y la realidad no van parejas, por lo que muchos estándares estudiados (incluyendo algunos particularmente extendidos) cumplen sólo parcialmente alguno de los criterios.

A continuación se analizará cada criterio junto con parte de la casuística encontrada en el estudio de cada uno de ellos, en incluyendo ejemplos de formatos o protocolos que, a priori, parecería que deberían ser aceptados de forma indiscutible pero que, en un estudio detallado, se concluye que no cumplen algunos de estos requisitos.

Criterio 1: iniciativas privadas Varios formatos son iniciativas de empresas privadas y, sin embargo, sus especificaciones son públicas y, en algunos casos, han sido aceptados como estándares por organizaciones internacionales como la ISO.

Esto incluye formatos y protocolos tan estandarizados y abiertos a lo largo de la historia de la informática como puedan ser PVM, NFS y RPC, entre otros. Por lo que se decidió aceptarlos.

Ejemplos claros pueden ser NFS y RPC, sistemas de ficheros y de ejecución remota de procesos ampliamente extendidos y considerados como estándares, que fueron desarrollados originalmente por Sun Microsystems y publicados en distintos RFC.

El formato PDF fue creado por una empresa privada, Adobe, y recientemente ha sido aceptado como estándar ISO (ISO 19005-1:2005).

Curiosamente, organizaciones tan abiertas como la IETF incorporan mecanismos casi dictatoriales en sus especificaciones, con el director de área teniendo la última palabra sobre todas las decisiones.

Criterio 2: precios simbólicos Lógicamente una especificación tiene que estar publicada para que sea siquiera tenida en cuenta. Sin embargo el punto “precio simbólico” puede resultar conflictivo. ¿Qué cantidad se considera simbólica?

Por ejemplo, las especificaciones publicadas por la organización ISO no se pueden adquirir de forma gratuita, aunque sí lo es su uso.

Pese a esta ambigüedad se han dado por buenos (“simbólicos”) los precios establecidos, siguiendo el criterio de Perens del precio de un libro de texto universitario.

Un ejemplo: la parte 1 del estándar MPEG-4 de la ISO cuesta 16 CHF (10 euros), un precio asequible a cualquier organización que lo requiera. Las implementaciones de referencia de los estándares de la ITU-T son algo más caras, rondando los 100 CHF, aunque algunas especificaciones están disponibles gratuitamente.

En particular, aquí los mejores resultados los obtiene la IETF, que da todos sus documentos de forma totalmente gratuita, incluidos los borradores.

Criterio 3: patentes Este punto ha sido el más problemático por la naturaleza de las patentes. Es difícil saber si una especificación vulnera o no alguna patente, sobre todo teniendo en cuenta la abundancia de éstas. De hecho, en algunas especificaciones los autores no garantizan ni se hacen responsables de que se cometa alguna infracción implementando su formato o protocolo.

Podría ser poco importante habida cuenta de que en Europa, y, por inclusión, España, las patentes de software no tienen validez. Pero esta situación no es definitiva, por lo que en cualquier momento pueden ser aceptadas.

Esto provocaría que opciones dadas por válidas en la actualidad no estén libres de regalías en un futuro no muy lejano.

En definitiva, los autores de los estándares garantizan que no han empleado deliberadamente patentes sin licencia sin regalías y no discriminatorias, pero no que no se infrinja alguna que no lo sea.

En el mundo real, y particularmente en los Estados Unidos, donde el problema es más grave, los implementadores suelen unirse todos a consorcios para cubrirse las espaldas mutuamente y formar portafolios de patentes que puedan licenciarse en bloque. Además, las organizaciones dedican un período expresamente a buscar problemas que puedan surgir con la propiedad intelectual del estándar.

Toda esta concienciación surge a raíz de casos como los del estándar MP3 para codificación de audio, o el formato GIF de imágenes, en que diversas organizaciones reclamaron patentes que poseían tras extenderse el uso de dicho estándar, aprovechándose de dicha situación para su propio beneficio. Dichas patentes son conocidas como *patentes submarino*.

Otro problema común viene a raíz de las propias licencias de las especificaciones. En el caso de especificaciones como la del formato de Microsoft Word 98, Apple Quicktime o el muy conocido Adobe Flash, éstas impiden o restringen sobremanera crear directamente implementaciones libres: mientras que el primero directamente impide su uso en software libre, Apple Quicktime fuerza a que el desarrollador no dé soporte para otro formato de salida (para así poder forzar su DRM) y Adobe Flash no permite usar la especificación para crear editores, sólo reproductores.

De hecho, en el caso de tecnologías como OpenXML de Microsoft, no se han dado licencias de patentes, sino lo que en inglés se conoce como “covenant not to sue”, es decir, la promesa de no demandar a aquellos que las infrinjan. Es

más, en casos como el de la Microsoft XML Paper Specification, la licencia dice específicamente que “no se permite sublicenciar o transferir los derechos”, impidiendo efectivamente el uso de dicha especificación en el desarrollo de software libre.

Criterio 4a: ¿qué es una implementación “completa”? Aparentemente este punto está claro. Pero realmente formatos tan extendidos y aceptados como XHTML o CSS *no* tienen ninguna implementación que cumpla al 100 % la especificación original. Ha sido necesaria cierta tolerancia al respecto.

Algunas veces, sobre todo con tecnologías particularmente complejas como en el caso de la voz por IP, se dispone de un amplio abanico de componentes posibles. No está claro si habría que considerar como “completa” una implementación libre si incorpora todos los componentes de una solución típica comercial o sólo los puramente esenciales. De la misma forma, al no disponer de especialistas en cada estándar, a veces es difícil saber cuáles son precisamente estos componentes esenciales.

Criterio 4b: implementaciones bajo copyleft De nuevo una condición tal vez demasiado restrictiva. Una especificación cumple este criterio si impide, explícitamente, implementaciones o variaciones no libres. Realmente, en la práctica, la mayor parte de los estándares que cumplen el criterio 4.a cumplen el 4.b, pero no siempre. LDAP es un ejemplo. Cumple todos los puntos menos este, y, además, es la única opción para directorios.

Aunque aproximadamente el 70 % de todos los proyectos de software libre están bajo la GPL, que es de copyleft fuerte, muchos proyectos importantes están bajo licencias sin copyleft fuerte o ninguno en absoluto, como la MPL (Mozilla Public License, copyleft débil), MIT (equivalente a la BSD revisada, sin copyleft), BSD (Berkeley Software Distribution, sin copyleft) o APL (Apache Public License, sin copyleft), como el mismísimo Apache HTTPd Server, o el servidor de autenticación OpenLDAP.

A veces hay alternativas copyleft menos conocidas, como Cherokee en el caso del Apache HTTPd. Otras veces no tanto, como en el caso de Mozilla Firefox o Mozilla Seamonkey, difícilmente reemplazables a lo largo de todas las plataformas (aunque podrían usarse alternativas como los navegadores basados en el motor KHTML como Konqueror o Safari).

Por otra parte, se realizaron esfuerzos para asegurar que un proyecto realmente era completamente libre (tanto él como todas sus dependencias). Ciertos proyectos tienen conflictos legales en sus dependencias: por ejemplo, el conocido software de videoconferencia Ekiga incluye excepciones específicamente para evadir la incompatibilidad entre su licencia GPL y la licencia MPL de sus dependencias con OpenH323 (MPL) y OPAL (también MPL).

3.4. Estados

Con posterioridad al estudio taxonómico realizado en función de los cinco criterios cubiertos para cada especificación estudiada, éstas fueron clasificadas,

asignándoles un determinado estado dentro de un diagrama flujo que modelaba el ciclo de vida de formatos, interfaces y especificaciones y que se muestra en forma de diagrama de estados UML en la figura 1. Esta clasificación debería determinar las acciones a tener en cuenta para cada uno de ellas y su posible validación como estándar para la interoperabilidad en la administración pública.

Para la definición de los distintos estados se tuvieron en cuenta, además del ciclo de vida obvio de las especificaciones técnicas, las necesidades funcionales y técnicas de la administración pública intentado que toda funcionalidad o servicio quedara cubierta, al menos, por una especificación que pudiera considerarse como recomendada (aunque en algunos casos de funcionalidades o servicios no críticos, se podría aceptar que todas ellas estuvieran etiquetadas como “En abandono”). También se reservó la etiqueta “En consideración” para aquellos servicios que no hayan sido estudiado suficientemente o para los que no se pueda obtener conclusiones definitivas.

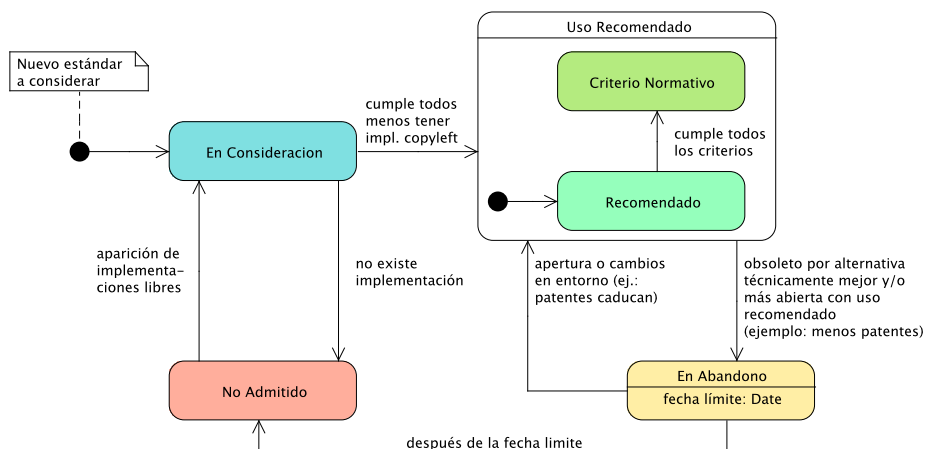


Figura 1. Ciclo de vida de una especificación

Los estados por los que puede pasar una especificación son:

Nuevo Cualquier especificación que no haya estado en “Uso Recomendado” con anterioridad.

En consideración Fase de estudio y evaluación. Sólo deben usarse en entornos de pruebas.

No admitido No cumple con las condiciones necesarias, por lo que se prohíbe su uso.

Recomendado Parte de “Uso Recomendado”. Su uso es voluntario.

Criterio normativo Parte de “Uso Recomendado”. Su uso es obligado.

En abandono Considerada inadecuada aunque aún se emplea.

Por último, se definieron una serie de reglas lógicas que, una vez aceptadas, permitieron utilizar nuestro estudio taxonómico para clasificar cada una de las

especificaciones en alguno de los estados anteriores. Un ejemplo (trivial) de tales reglas es:

“No admitido” = .not. “Existe implementación”

4. Conclusiones y trabajo futuro

En la Oficina de Software Libre de la Universidad de Cádiz se ha realizado un trabajo riguroso y exhaustivo, clasificando cientos de especificaciones (desde formatos de documentos gráficos hasta protocolos de e-learning o de Voz IP) según cinco criterios. Este estudio ha permitido que, una vez determinadas una serie de reglas lógicas que deberían acuar sobre dichos criterios, sea posible el realizar un estudio objetivo que conlleva la clasificación de cada una de las especificaciones en un diagrama de estados, en función de los cuales la administración pública puede discernir cuales de éstas puede considerar como un estándar para la interoperabilidad.

De cualquier manera, el estudio dista mucho de estar cerrado. En primer lugar, la propia naturaleza del diagrama de flujo implica que la clasificación variará con el tiempo y requerirá de la su revisión periódica.

Incluso, como ya se ha mencionado en 3.3, no podemos descartar que (desgraciadamente) en un futuro sean aprobaran las patentes de software en el ámbito europeo, lo que forzaría a una revisión más detallada de los estándares aprobados para evitar problemas legales.

Por último, además de los criterios estudiados existe mucha más información que también podría ser de interés para una taxonomía de estándares abiertos. Por ejemplo (como sugiere [2]) se podría incluir información acerca de las propias organizaciones de estándares. Y sería especialmente útil añadir una métrica de la calidad del propio estándar: un estándar, además de ser libre, debería de cumplir diversas características deseables, como no favorecer a ningún vendedor en particular y ser fácil de implementar. Una administración pública podría optar, en caso de existir varios estándares paralelos, por aquél que sea de mayor calidad.

Referencias

1. Bob Sutore: Open Standards versus Open Source. http://www-03.ibm.com/developerworks/blogs/page/BobSutor?entry=open_standards_vs_open_source1, Jan 2006.
2. Ken Krechmer: Open Standards Requirements. The International Journal of IT Standards and Standardization Research, Jan 2006, volume 4, number 1. También disponible en <http://www.csrstds.com/openstds.pdf>.
3. European Communities: European Interoperability Framework. <http://europa.eu.int/idabc/en/document/3473/5585>, Nov, 2004.
4. United Kingdom Cabinet Office, e-Government Unit: e-Government Interoperability Framework. <http://www.govtalk.gov.uk/schemasstandards/egif.asp>

5. Bundesministerium des Innern: Standards und Architekturen für E-Government-Anwendungen. http://www.alis-etsi.org/IMG/pdf/Germany_Fev_03.pdf, Feb 2003.
6. Ministério do Planejamento, Orçamento e Gestão (Brasil): ePING: Padrões de Interoperabilidade de Governo Eletrônico. http://www.governoeletronico.gov.br/governoeletronico/publicacao/down_anexo.wsp?tmp.arquivo=E15_677e-PING_v2.0_17112006_LINUX_FINAL.pdf, Nov 2006.
7. Bruce Perens: Open Standards, Principles and Practice. <http://perens.com/OpenStandards/Definition.html>, Feb, 2006.