# Contribución a Proyecto Open Source 2
# Mejoras para NWiki, una wiki para Moodle

Gonzalo Serrano Revuelta

20 de junio de 2008

*A mis padres, Carlos y Teresa.*
*A mi madrina, Elisa.*
*Al resto de mi familia y a mis amigos.*

# Índice general

# Capítulo 1

# Presentación y objetivos

## 1.1. Prólogo

A la hora de afrontar una tarea importante y larga como es un Proyecto de Final de Carrera (PFC) de una Ingeniería hay que tener muy claras las cosas antes de empezar si se quiere llevar a buen puerto desde el principio hasta el final.

Han habido varios factores que han influido en mi decisión de seleccionar un proyecto u otro. Lo que quería era:

- Un proyecto que me motivara.
- Un proyecto relacionado con la programación web.
- Un proyecto cuyo director tuviera buenas referencias.
- Un proyecto que fuera *software libre*.
- Un proyecto útil.

### Un proyecto motivador

La razón más importante de todas. Si uno se va a pasar varios (muchos, en mi caso) meses, el PFC ha de ser una proyecto que sirva para aprender cosas nuevas constantemente, que permita desarrollar características interesantes desde el punto de vista técnico y del que uno se sienta orgulloso de haber aprendido cosas interesantes al haberlo terminado.

### Un proyecto relacionado con la programación web

Tenía claro que una de las ramas de la informática que me interesaba más profundizar era en el desarrollo web. Ha sido un área poco practicada durante mis estudios como Ingeniero Técnico en Informática de Sistemas en la Universitat de Les Illes Balears pero que sí tuve oportunidad de ejercer durante 6 meses trabajando en la empresa UPCNet antes de empezar la Ingeniería Superior. En ésta última pude estudiar diferentes áreas del desarrollo web en la asignatura *Projecte de Xarxes de Computadors*, que no hizo más que confirmar que era un área que me interesaba, y mucho.

También relacionado con ésto es el mundo laboral que hay después de una ingeniería informática, donde la oferta de puestos de trabajo relacionados con el desarrollo web es cada año mayor y es en Internet donde se encuentran muchas de las oportunidades de negocio del mundo de la informática y las telecomunicaciones.

### Un proyecto cuyo director tuviera buenas referencias

Estuve barajando una lista de proyectos y antes de decidirme busqué opiniones sobre los directores de cada uno de ellos, siendo éstas nulas, positivas y negativas. Los proyectos en los cuales ha participado Marc Alier [1] [2], mi director de proyecto, son accesibles via Internet ahora mediante el portal web

[1] Página personal: http://www.lsi.upc.edu/ malier/
[2] Blog: http://orangoodling.blogspot.com

DFWikiLabs [3] y allí se puede leer largo y tendido sobre en qué consisten sus proyectos, por lo que tenía información de primerísima mano. Las entradas de su *blog* y su página personal también me ayudaron a hacerme una idea del perfil del director, ya que no había tenido la oportunidad de tenerle como profesor.

### Un proyecto que fuera *software libre*

La importancia de *software libre* ha sido una pieza clave a lo largo de mi carrera universitaria; su filosofía ha influenciado en mi ética profesional y sus programas, protocolos y estándares me han ayudado a resolver problemas durante toda la carrera. Además han mantenido una motivación en mi que no hubiera tenido si me hubiera centrado exclusivamente en el uso de programas privativos.

Es por eso pienso que me siento un poco en deuda con la comunidad que hay detrás del mundo del *software libre* y a la cual me hacía ilusión devolver algo del tiempo ganado gracias a ella.

Una de las frases típicas o *slogan* de la comunidad de desarrolladores de *free software* es la siguiente:

> *No pienses qué puede hacer el software libre por ti, sino qué puedes hacer tú por el software libre.*

Por eso lo que tenía claro es que me interesaba un proyecto final de carrera basado en *software libre* y que además fuera *software libre*.

### Un proyecto útil

Uno siempre se siente orgulloso del trabajo hecho si se cree que se ha hecho bien, pero más orgulloso se siente si ése trabajo sirve para otras personas.

[3] http://dfwikilabs.org

El proyecto elegido me ha dado la posibilidad de que mi trabajo vaya a ser utilizado por muchos usuarios y que no se quede en un cajón después de haberlo terminado.

## 1.2. Descripción general del proyecto

El PFC *Contribución a Proyecto Open Source 2* consiste en una serie de ampliaciones funcionales al programa NWiki, desarrollado por diversos proyectistas de la Universidad Politécnica de Catalunya (UPC) en la Facultad de Informática de Barcelona (FIB) y dirigidos por Marc Alier.

NWiki es una wiki, aplicación web que principalmente posibilita la edición de un documento colaborativamente a través de una red local o de Internet, programada como módulo de Moodle.

Moodle es, según los propios autores del programa, a *Free, Open Source Course Management System for Online Learning*. Esto es, se trata de una aplicación web libre de *e-learning* que sirve para gestionar cursos para el aprendizaje a través de la red.

Más sobre estos conceptos en los siguientes apartados.

## 1.3. Conceptos

En ésta sección explico más detalladamente en qué consiste el software libre, el sistema educacional *e-learning*, la plataforma Moodle, la tecnología Wiki y el proyecto NWiki.

### 1.3.1. Software Libre

Según la Fundación por el Software Libre o *Free Software Fundation* [4], el *software libre* o *free software*...

> *... se refiere a la libertad de los usuarios para ejecutar, distribuir, estudiar, cambiar y mejorar el software.*

Más concretamente, se refiere a cuatro libertades de los usuarios del software:

- Libertad 0: La libertad de usar el programa, con cualquier propósito.
- Libertad 1: La libertad de estudiar cómo funciona el programa, y adaptarlos a tus necesidades.
- Libertad 2: La libertad de distribuir copias, con lo que puedes ayudar a tu vecino.
- Libertad 3: La libertad de mejorar el programa y hacer públicas las mejoras a los demás, de manera que toda la comunidad se beneficie.

Para publicar un proyecto como software libre es necesario hacerlo bajo una licencia compatible con éstas libertades. Existen muchísimas licencias [5] que son software libre. Quizás la más famosa y extendida es la licencia GNU GPL, que comento en el siguiente apartado.

#### El proyecto GNU y la licencia GPL

El proyecto GNU [6] (acrónimo recursivo de *GNU is Not UNIX*) fue creado en 1983 por Richard Stallman con la finalidad de crear un sistema operativo totalmente libre. Para asegurar que el software GNU permaneciera libre, el proyecto debía ser publicado bajo una licencia diseñada para garantizar esos derechos al tiempo que evitase restricciones posteriores de los mismos.

[4] http://www.fsf.org
[5] http://www.gnu.org/licenses/license-list.es.html
[6] http://gnu.org

La idea, que se conoce como *copyleft* o *copia permitida* (en oposición al *copyright*) se plasmó en la licencia GNU *General Public License* [7].

Figura 1.1: El Ñu es la mascota oficial del proyecto GNU.

La importancia del software libre es muy importante en el mundo de la informática y de las telecomunicaciones. Por ejemplo, la mayoría de servicios que sustentan la infraestructura de Internet son libres. Los ordenadores servidores tienen instalados sistemas operativos como GNU/Linux, servidores web como Apache , servidores de correo, servidores FTP, programas de filtro *antispam*, de resolución de nombres (DNS), etc etc, están publicados con una licencia libre. También existen miles de programas libres para ordenadores de sobremesa que son usados por millones de personas, sobretodo distribuciones de GNU/Linux con entornos de escritorio avanzados como KDE [8] o GNOME [9].

El software libre es una alternativa real al software privativo y existe un negocio muy importante a su alrededor. Empresas multinacionales como Google, IBM , Sun , Nokia , Novell o Red Hat [10] ganan e invierten decenas e incluso miles de millones de euros gracias al software libre.

Desde el ámbito académico cada vez más universidades apuestan por el software libre, y éste PFC no es una excepción.

[7] http://gnu.org/licenses/gpl.txt
[8] http://kde.org
[9] http://gnome.org
[10] http://www.redhat.com

#### El software libre en éste PFC

Éste PFC es y se ha basa en software libre.

Es software libre porque el código para implementar las mejoras en NWiki se publica bajo la licencia GPL versión 2, concretamente en la web de DFWikiLabs [11]. Además NWiki no tendría sentido sin su integración en Moodle [12], que también está publicado bajo la licencia GNU GPL v2 en y se puede descargar desde el sitio web oficial de Moodle [13].

Se basa en software libre porque ha sido realizado sobre una distribución de GNU/Linux (Kubuntu Feisty [14]) sobre el entorno de escritorio KDE y el editor Vim [15]. La documentación ha sido escrita en Vim también en el lenguaje LaTeX.

#### Mapa conceptual del software libre

A continuación se muestra un diagrama que explica varios elementos y conceptos que interactúan alrededor del software libre.

### 1.3.2. E-learning

El *e-learning*, *electronic learning* o *aprendizaje virtual* es el término utilizado para referirse a la manera de aprender donde el instructor o profesor y el estudiante están separados por espacio (limitaciones geográficas) o tiempo donde se cubre esa separación a través del uso de tecnologías *online*. Esto ha supuesto una revolución en los sistemas educacionales tradicionales, cambiando el concepto de *educación a distancia*.

[11] http://dfwikilabs.org
[12] http://moodle.org
[13] http://download.moodle.com
[14] http://www.kubuntu.org
[15] http://www.vim.org

En *e-learning* Internet se convierte en la infraestructura básica para desarrollar los procesos de enseñanza-aprendizaje no presenciales, combinando servicios síncronos y asíncronos, lo que ha dado lugar a un modelo cada vez más valorado, no como sustituto de la formación presencial tradicional, sino más como un complemento que se ha de adaptar según las necesidades y nivel de madurez del público receptor de esta formación. Sin embargo, las aproximaciones mixtas, que combinan actividades formativas presenciales y no presenciales (o soluciones *blended*), toman cada vez más fuerza y se posicionan como una importante alternativa.

### Ventajas e inconvenientes

Algunas ventajas del *e-learning* frente al aprendizaje tradicional son las siguientes:

- Independencia geográfica: el profesor y el alumno pueden acceder al sistema de aprendizaje desde cualquier ordenador con conexión a Internet.
- Independencia temporal: el profesor y el alumno pueden acceder al sistema en la franja horaria que les sea más conveniente.
- Eficiencia temporal de aprendizaje: el alumno puede decidir su propio ritmo de aprendizaje.
- Aumento de la interacción profesor-alumno: se eliminan las barreras psicológicas derivadas de los encuentros cara a cara del alumno con el profesor.
- Aumento de la interacción alumno-alumno: sistemas como los foros o *chat* en tiempo real integrados en esos sistemas permiten a los alumnos comunicarse ente ellos de manera más eficaz.
- Seguimiento del alumno mucho más eficiente y detallado: éste tipo de sistemas permiten recoger estadísticas de los accesos, trabajos y manera

de trabajar de los alumnos casi instantáneamente, por lo que el profesor cuenta con muchos más factores para valorar a los alumnos.

- Disposición de recursos *online* y multimedia (documentos digitales, audio, vídeo).
- Coste: al ser las aulas virtuales y los documentos digitales, se ahorra muchísimo dinero en costes.

Pero no todo son ventajas en los sistemas de *e-learning*:

- Es necesario contar con un acceso a Internet de banda ancha para poder acceder a los servicios a una velocidad adecuada.
- Hay materias en los que directamente es imposible que sus contenidos sean dados de manera virtual y se hace necesario el uso de clases magistrales.
- Es necesario un dominio previo del uso de ordenadores y de sistemas web que determinados espectros de la población (léase personas de una cierta edad) actualmente no tienen y que les supondría un esfuerzo extra.

### Componentes

Los elementos que componen una sistema de *e-learning* son básicamente tres:

1. Tecnología: se trata de plataformas o campus virtuales, es el conjunto de *software* y *hardware* que se encuentra del lado del servidor.
2. Contenidos: se refiere a la información que se almacena en esas plataformas y que sirve de base para ofrecer los recursos a los alumnos y profesores.
3. Servicios: consisten en una serie de elementos que posibilitan acciones para la gestión, comunicación, evaluación... de los contenidos a través de la tecnología.

### Tipos de plataformas de *e-learning*

Existen diferentes tipos de programas que sirven como plataformas de *e-learning*.

Estos programas tienen en común que se basan en aplicaciones web, por lo que los alumnos y profesores interactúan con el sistema a través de un navegador web. pero se diferencian en cuanto al objetivo y funcionalidades que ofrecen, siendo la diferenciación algo difusa en algunos casos.

Algunos términos que se utilizan para definir dichos programas son:

- LMS o *Learning Management System*: se trata de sistemas que se centran más en gestión de cursos, profesores y alumnos y en la enseñanza profesor-alumno mediante el seguimiento de las tareas que cumplen éstos últimos en cada curso. También permiten asignar notas a las tareas que realizan los alumnos. Son conocidos también como CMS (referido a *Course Management System* o VLE (*Virtual Learning Environment*).

- LCMS o *Learning Content Management System*: son sistemas que tiene el origen en CMS (*Content Management System* y se centran más en la parte de publicación de contenidos y recursos educativos *online* en repositorios que pueden ser utilizados directamente o o de manera independiente de los cursos dentro del sistema.

Éste proyecto se centra en la parte de LMS ya que el sistema base es uno de ellos. Aún así, existen otros muchos LMS en el mercado, tanto libres como privativos. Algunos de ellos, quizás los más usados, son los siguientes:

- Blackboard [16]: uno de los sistemas más usados, se trata de un software privativo perteneciente a la compañía Blackboard Inc. Dicha empresa tiene la patente estadounidense *Internet-based education support system and methods* [17] la cual ha sido llevada a tribunales por otros LMS

[16] http://www.blackboard.com
[17] http://patft.uspto.gov/netacgi/nph-Parser?patentnumber=6988138

acusados de violar dicha patente.

- WebCT [18]: recientemente adquirido por Blackboard, es un LMS privativo muy utilizado en universidades españolas considerado uno de los más complejos de utilizar por parte de los usuarios.
- Claroline [19]: se trata de un LMS libre bastante extendido: oficialmente ha sido implantado en 1182 organizaciones en 93 países.
- Sakai Project [20]: se trata de un sistema libre publicado hace relativamente poco (año 2005) que actualmente está instalado en 250 organizaciones, y donde la UPC participa en su desarrollo.
- Moodle: se trata de un LMS libre muy utilizado y sobre el cual se basa éste PFC, por lo cual va a ser comentado más detalladamente.

### 1.3.3. Moodle

#### Introducción

Como se ha dicho anteriormente, Moodle es un LMS libre, y además uno de los más usados y de los que crec más rápidamente [21]: actualmente hay casi 45.000 organizaciones y 20 millones de usuarios que lo usan. La UPC no es una excepción y son varias las asignaturas las que ofrecen sus contenidos a través de ésta plataforma desde enero de 2005 [22].

#### Historia

Moodle fue creado por Martin Dougiamas, quien fue administrador de WebCT en la Universidad Tecnológica de Curtin. La primera versión de Moodle se

[18] http://www.webct.com/webct
[19] http://www.claroline.net
[20] http://sakaiproject.org
[21] http://moodle.org/stats
[22] http://www.upc.es/catala/la-upc/govern/bupc/hemeroteca/2005/b75/22-3-2005.pdf


Figura 1.2: Logo oficial del proyecto Moodle.

lanzó en agosto de 2002, y se han ido publicando varias versiones con mejoras desde entonces, siendo la versión estable a día de hoy Moodle 1.9. Martin sigue liderando el proyecto desde entonces.

Martin basó su diseño para Moodle en las ideas del constructivismo en pedagogía que afirman que el conocimiento se construye en la mente del estudiante en lugar de ser transmitido sin cambios a partir de libros o enseñanzas y en el aprendizaje colaborativo. Un profesor que opera desde este punto de vista crea un ambiente centrado en el estudiante que le ayuda a construir ese conocimiento con base en sus habilidades y conocimientos propios en lugar de simplemente publicar y transmitir la información que se considera que los estudiantes deben conocer.

### Características

Las características más importantes de Moodle son las siguientes:

- Es software libre y gratuito.
- Su instalación es sencilla y asistida, y su uso es sencillo también gracias a su interfaz ligera y compatible.
- Es seguro: soporta *cookies* cifradas, validación de formularios, tiene mecanismos de prevención de ataques de inyección de código y XSS (*Cross-site Scripting*), etc.
- Es modulable: la flexibilidad de su arquitectura y su API (*Application Programming Interface* permiten escribir módulos externos de manera sencilla.

- Es escalable: soporta miles de usuarios y cursos. Moodle.org tiene más de 320.000 usuarios y hay un sitio con más de 19.000 cursos.

- Está escrito en PHP [23] y soporta sistemas gestores de bases de datos (SGBD) libres como MySQL [24] o PostreSQL [25].

### Principales módulos de Moodle

Como se ha dicho, la arquitectura de Moodle permite separar las funcionalidades en módulos o *plugins* independientes. Algunos de los más importantes son:

- *Chat*: permite a los participantes mantener conversaciones online en tiempo real.

- *Choice*: permite hacer pequeñas encuestas formulando una pregunta y dando múltiples repuestas. Cada participante elige una respuesta que considere adecuada.

- *Database*: permite visualizar información en forma de tablas especiales para la web.

- *Forum*: típico foro donde se permite a los participantes crear hilos de discusión y mensajes de respuesta a esos hilos.

- *Glossary*: permite crear y mantener listas de definiciones, como un diccionario.

- *Lesson*: permite añadir contenido de manera flexible en formato de páginas consecutivas. Para cada página el profesor puede añadir una serie de preguntas a las que hay que responder antes de pasar a la siguiente página.

[23] http://php.net
[24] http://www.mysql.com
[25] http://www.postgresql.org

- *Quiz*: permite diseñar y crear cuestionarios de una manera flexible basándose en un conjunto de tipos de preguntas.
- *Survey*: módulo de encuestas más general.
- *Wiki*: módulo que permite la edición colaborativa de documentos *online* mediante un lenguaje de marcado especial.

Éste PFC trata sobre mejoras a un módulo Wiki externo para Moodle llamado NWiki, por lo que se va a comentar éste tipo de sistemas más ampliamente en los siguientes apartados.

### 1.3.4. Wiki

Una wiki (del hawaiano *wikiwiki*, que significa *rápido*) es una aplicación web cuyo propósito es posibilitar la edición de los contenidos de las páginas wiki por parte de los usuarios de una manera colaborativa, rápida y sencilla y a través de un navegador web.

El origen de los sistemas wiki se encuentra en el sitio web WikiWikiWeb [26] creado por Ward Cunningham en 1995 para poder administrar el repositorio de patrones de diseño de la ciudad de Portland, en Estados Unidos. WikiWikiWeb sigue siendo un sitio web muy utilizado actualmente.

El mayor exponente de un sistema wiki es la Wikipedia, la enciclopedia *online*, que cuenta con más de diez millones artículos.

#### Características

Las principales características de un sistema wiki son:

- Es un conjunto de una o más páginas web editables por los usuarios.

[26] http://c2.com/cgi/wiki

- La edición de una página wiki no se realiza con el lenguaje de marcado X/HTML sino que se ha creado un lenguaje de marcado más sencillo (*wiki markup language*) que es traducido por el programa a X/HTML para una posterior visualización correcta en el navegador.

- Existen diferentes lenguajes wiki de marcado o sintaxis wiki, siendo los más famosos los soportados por MediaWiki [27] (el *software* sobre el que funciona la Wikipedia) y WikiCreole [28], una sintaxis creada para estandarizar las diferentes sintaxis wiki existentes.

- Contienen un sistema de control de versiones integrado que facilita la posibilidad de revertir los cambios hechos por los usuarios. Además la política de ser editable por cualquier usuario a veces provoca vandalismos (corromper los contenidos de páginas adrede) que han de ser fácilmente revertibles.

- También suelen contar con otras características, como una página de discusión asociada a cada página wiki para comentar posibles cambios, un sistema de *diff* para visualizar las diferencias entre páginas, sistemas de bloqueo de páginas, sistema de visualización de cambios recientes, etc.

### 1.3.5. NWiki

NWiki es una wiki implementada como módulo de Moodle y que reemplaza la wiki que viene de serie en Moodle (*ewiki*).

#### Historia

NWiki nació en 2005 como un proyecto final de carrera llamado DFWiki, en honor a sus creadores David Castro y Ferràn Recio, dos estudiantes de la

[27] http://www.mediawiki.org
[28] http://www.wikicreole.org

FIB y dirigidos por Marc Alier.

Originalmente el proyecto a implementar era WikiBook para Moodle, una adaptación del formato wiki para generar un contenido similar al de un libro. La precariedad de *ewiki* y la dificultad de integrar sistemas como MediaWiki en Moodle hizo a Marc Alier contactar con Martin Dougiamas, que le comunicó que si era capaz de programar un módulo wiki mejor que *ewiki* entonces sustituiría *ewiki* por ése nuevo módulo nuevo.

A partir de ese momento han habido diferentes PFC relacionados con implementación de mejoras en DFWiki y otros proyectos relacionados, dando lugar a al equipo de trabajo DFWikiTeam y al portal web DFWikiLabs [29].

En 2006 y después de un análisis del código de DFWiki por parte de los desarrolladores principales de Moodle, Martin Dougiamas propuso una serie de cambios para adaptar la wiki a los estándares de calidad de Moodle. A partir de ahí el proyecto pasó a llamarse NWiki.

A principios de 2008 y después una votación dentro de la comunidad Moodle se decidió que NWiki iba a pasar a ser la wiki oficial de la próxima versión de Moodle, la 2.0.

### Características de NWiki

NWiki cuenta con una serie de funcionalidades que la diferencian de otras wikis convencionales:

- Se instala y se integra sobre Moodle, sobreescribiendo a la antigua *ewiki*. Además permite actualizar las páginas de *ewiki* a NWiki.
- Soporta el sistema de cursos y usuarios (administradores, profesores, estudiantes) de Moodle, de tal manera que una wiki NWiki puede ser es una actividad de un curso de la wiki.

[29] http://www.dfwikilabs.org/

- Soporta wikis para estudiantes juntos, separados, para grupos de estudiantes, etc típicos de las actividades Moodle.
- Soporta internacionalización (i18n). Está traducida al inglés al 100y también al castellano y catalán, aunque no por completo.
- Utiliza la API de Moodle para el acceso a cursos, usuarios, BBDD etc.
- Añade bloques Moodle para la administración de la wiki, bloques de estadísticas (páginas más editadas, más vistas...).
- Más características...

NWiki actualmente soporta una serie de características aquí no indicadas. Éstas funcionalidades son las que se han hecho durante el proyecto final de carrera que se presenta en éste documento. Las funcionalidades nuevas se comentan a continuación como parte de los objetivos del PFC.

## 1.4. Metodología

### 1.4.1. Descripción general

Éste PFC ha sido un proyecto muy dinámico en el que las funcionalidades a implementar han cambiado con el paso del tiempo según el *feedback* del director Marc Alier.

A la hora de implementar las funcionalidades, la metodología a seguir ha sido una mezcla entre el modelo de desarrollo en cascada (análisis, diseño, codificación, prueba, implentanción) y el modelo en espiral (determinar objetivos, análisis de riesgo, desarrollo + verificación + prueba, planificación siguiente).

Los pasos seguidos para cada tarea realizada han sido las siguientes:

1. Reunión con Marc Alier y Jordi Piguillem en la facultad, donde se

analizaba el tipo de funcionalidad a implementar, sus objetivos y un diseño inicial en alto nivel.

2. Análisis del código de NWiki y de Moodle involucrados en la implementación.

3. Breve diseño más específico de la funcionalidad.

4. Implementación parcial de la funcionalidad y demostración a Marc.

5. Refinamiento.

6. Publicación de la funcionalidad a la comunidad Moodle.

7. Mantenimiento de la funcionalidad (*bugfixing* y otros cambios). Volver a 1.

### 1.4.2. Reuniones

Durante el transcurso del PFC han habido múltiples reuniones en la facultad con el director Marc Alier i/o con el programador jefe a cargo de NWiki Jordi Piguillem. En las reuniones se comentaba el ritmo de trabajo, las últimas funcionalidades implementadas, cambios en el diseño de funcionalidades, etc.

Normalmente el ritmo ha sido de una reunión por semana los lunes por la mañana, aunque han habido semanas donde no ha habido reunión por determinadas circunstancias y otras semanas donde han habido más de una a la semana.

### 1.4.3. Blog

Todos los proyectistas de Marc han creado un *blog* o bitácora electrónica donde comentar progresivamente el trabajo hecho durante la realización del PFC.

Después de analizar varias plataformas me decanté por WordPress, ya que ofrecen, además un *software* muy completo y libre, un espacio web en su portal http://wordpress.com.

La dirección del blog es http://parabol.wordpress.com y todas las entradas relacionadas con el PFC pueden leerse en el Anexo I de éste mismo documento.

## 1.5. Objectivos

Los objetivos del proyecto corresponden con la implementación de las nuevas funcionalidades requeridas por el director del proyecto, cuyo resumen es el siguiente:

- Edición y visualización de secciones de páginas wiki de manera independiente.
- Sistema de bloqueos para evitar problemas de concurrencia al editar páginas y secciones de página.
- Mejoras en el parser NWiki:
  - Resolver problemas de seguridad del tipo *SQL injection* o inyección de código SQL.
  - Mejoras en la sintaxis Creole.
- *Backport* a Moodle 1.8 de las funcionalidades implementadas en NWiki para Moodle 1.9.
- Arreglar una serie de errores en el subproyecto *Wikibook*.
- Implementación de un sistema de etiquetado o *tagging* para páginas wiki, así como de un bloque Moodle para mostrar las etiquetas en forma de nube de tags.

- Terminar el sistema de notas o *grades* de NWiki creado por otro proyectista.
    - *Bugfixing* de las funcionalidades existentes.
    - Implementación de nuevas funcionalidades.
    - Diseño e implementación de una nueva interfaz.
- Cambios en la interfaz de la funcionalidad *historia* de NWiki.

La explicación de cada funcionalidad se encuentra en el el Capítulo 3: *Funcionalidades* en la página 45.

# Capítulo 2

# Tecnologías y herramientas

Las tecnologías y herramientas con las que se ha llevado a cabo éste proyecto han sido aquellas con las cuales trabajan Moodle y NWiki. Casi todas las tecnologías y herramientas (excepto dos) utilizadas en éste proyecto son estándares y/o *software libre*.

Tanto Moodle como NWiki están basados en un conjunto de tecnologías y herramientas llamadas LAMP (Linux, Apache, MySQL, PHP). Forman conjuntamente una plataforma tecnológica básica que es la solución profesional libre más utilizada en programación web.

Se comentan dichas tecnologías y herramientas, juntamente con otras utilizadas en éste proyecto, a continuación.

## 2.1. Tecnologías

Aquí se comentan una serie de lenguajes y protocolos utilizados.

### 2.1.1. HTTP

HTTP o *HyperText Tranfer Protocol* es, como indica su nombre, un protocolo para la transmisión de hipertexto, y es utilizado para el funcionamiento de Internet.

HTTP fue desarrollado por el consorcio W3C y la IETF, colaboración que culminó en 1999 con la publicación de una serie de RFC o *Request For Comments*, siendo el más importante de ellos el RFC 2616 que especifica la versión 1.1 de HTTP.

HTTP define la sintaxis y la semántica que utilizan los elementos software de la arquitectura web (clientes, servidores, *proxies*) para comunicarse. Es un protocolo orientado a transacciones y sigue el esquema petición-respuesta entre un cliente y un servidor. Al cliente que efectúa la petición (un navegador o un spider) se lo conoce como *user agent* (agente del usuario). A la información transmitida se la llama recurso y se la identifica mediante un URL o *Localizador Uniforme de Recursos*. Los recursos pueden ser archivos, el resultado de la ejecución de un programa, una consulta a una base de datos, la traducción automática de un documento, etc.

HTTP es un protocolo sin estado, es decir, que no guarda ninguna información sobre conexiones anteriores. El desarrollo de aplicaciones web necesita frecuentemente mantener estado. Para esto se usan las *cookies*, que es información que un servidor puede almacenar en el sistema cliente. Esto le permite a las aplicaciones web instituir la noción de *sesión*, y también permite rastrear usuarios ya que las *cookies* pueden guardarse en el cliente por tiempo indeterminado.

### 2.1.2. HTML, XML y XHTML

#### HTML

HTML o *HypetText Markup Language* es un lenguaje de marcas que deriva del lenguaje SGML y es el lenguaje principal para la construcción de páginas web.

Es usado para describir la estructura y el contenido en forma de texto, así como para complementar el texto con objetos tales como imágenes. HTML se escribe en forma de etiquetas o marcas, rodeadas por corchetes angulares (<, >). HTML también puede describir, hasta un cierto punto, la apariencia de un documento, y puede incluir un script (por ejemplo Javascript), el cual puede afectar el comportamiento de navegadores web y otros procesadores de HTML.

Originalmente HTML fue un lenguaje diseñado para compartir información científica entre científicos de todo el mundo. Era puramente un lenguaje estructural, en el que no había forma de describir la apariencia de las páginas ni la posibilidad de poner, por ejemplo, un texto en negrita o cursiva.

Más adelante se añadieron numerosas opciones para formatear y presentar gráficos y texto. Se desarrollaron las ampliaciones de HTML para conseguir la presentación deseada, pero siempre desde perspectivas de diferentes desarrolladores que terminaron en una serie de implementaciones del lenguaje no estándares que funcionaban en diferentes navegadores web.

Todo ello propició la aparición de un consorcio que controlase la evolución del lenguaje HTML, el W3C o *World Wide Web Consortium*. Dicha evolución tuvo un momento clave: cuando se decidió separar el contenido de las páginas HTML de su diseño. Así, a partir de la versión 4 de HTML se recomendó otro mecanismo para controlar la visualización del contenido del HTML, y ese mecanismo fueron las hojas de estilo (CSS o *Cascading Style Sheets*).

### XML

XML o *Extensible Markup Language* es un metalenguaje extensible de etiquetas desarrollado por el World Wide Web Consortium (W3C). Es una simplificación y adaptación del SGML y permite definir la gramática de lenguajes específicos (de la misma manera que HTML es a su vez un lenguaje definido por SGML).

Por lo tanto XML no es realmente un lenguaje en particular, sino una manera de definir lenguajes para diferentes necesidades con el propósito de intercambiar información estructurada entre diferentes plataformas. Se puede usar XML en bases de datos, editores de texto, hojas de cálculo, etc. Algunos de estos lenguajes que usan XML para su definición son XHTML, SVG, MathML, etc.

XML es, por tanto, una tecnología sencilla que tiene a su alrededor otras tecnologías que la complementan y la utilizan, de manera que actualmente tiene un papel muy importante ya que posibilita la compatibilidad entre sistemas para compartir la información de una manera segura, fiable y sencilla.

### XHTML

XHTML o *eXtensible Hypertext Markup Language* es el lenguaje de marcado pensado para sustituir a HTML como estándar para las páginas web.

XHTML es la versión XML de HTML, por lo que tiene, básicamente, las mismas funcionalidades, pero cumple las especificaciones, más estrictas, de XML. Su objetivo es avanzar en el proyecto del World Wide Web Consortium de lograr una web semántica, donde la información, y la forma de presentarla estén claramente separadas. En este sentido, XHTML serviría únicamente para transmitir la información que contiene un documento, dejando para las hojas de estilo su aspecto y diseño en distintos medios (ordenadores, PDAs, teléfonos móviles, impresoras...) y JavaScript para su comportamiento.

### 2.1.3. PHP

PHP, acrónimo recursivo de *PHP: Hypertext Preprocessor* es un lenguaje de programación interpretado que se ejecuta del lado del servidor que fue diseñado para la creación de páginas web dinámicas. Es decir, el servidor genera mediante scripts PHP páginas X/HTML que el cliente visualiza mediante el navegador web.

PHP fue creado por Rasmus Lerdof en 1994 y actualmente la rama principal está desarrollada por The PHP Group. PHP puede ser desplegado en la mayoría de servidores web y sistemas operativos. Actualmente se encuentra instalado en más de 20 millones de sitios web y en 1 millón de servidores, aunque su uso ha declinado un poco desde 2005 por el aumento de popularidad de otras soluciones.

El uso de PHP tiene muchas ventajas:

- Es un lenguaje multiplataforma.
- Capacidad de conexión con la mayoría de los manejadores de base de datos que se utilizan en la actualidad, destaca su conectividad con MySQL Capacidad de expandir su potencial utilizando la enorme cantidad de módulos (llamados *exts* o extensiones).
- Posee una amplia documentación en su página oficial, entre la cual se destaca que todas las funciones del sistema están explicadas y ejemplificadas en un único archivo de ayuda.
- Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- Permite las técnicas de Programación Orientada a Objetos.
- Tiene una incluida biblioteca nativa de funciones sumamente amplia.
- No requiere definición de tipos de variables.

- Tiene manejo de excepciones.

Pero PHP también tiene algunas desventajas:

- No posee una abstracción de base de datos estándar, sino bibliotecas especializadas para cada sistema gestor de bases de datos (a veces más de una para el mismo motor).
- Por su diseño dinámico no puede ser compilado y es muy difícil de optimizar.
- Por sus características promueve la creación de código desordenado y complejo de mantener. Está diseñado especialmente para un modo de hacer aplicaciones web que es ampliamente considerado problemático y obsoleto (mezclar el código con la creación de la página web).

### 2.1.4. JavaScript

JavaScript es un lenguaje de programación interpretado utilizado principalmente en páginas web, con una sintaxis semejante a la del lenguaje Java y el lenguaje C. Al igual que Java, JavaScript es un lenguaje orientado a objetos propiamente dicho, ya que dispone del mecanismo de herencia, si bien esta se realiza siguiendo el paradigma de programación basada en prototipos porque las nuevas clases se generan clonando las clases base (prototipos) y extendiendo su funcionalidad.

En el entorno de desarrollo, JavaScript se ejecuta del lado del cliente, es decir, en el navegador. Todos los navegadores modernos interpretan el código JavaScript integrado dentro de las páginas web. Para interactuar con una página web se provee al lenguaje JavaScript de una implementación del DOM o *Document Object Model*, que representa la estructura de un documento web en forma de objetos.

El lenguaje fue inventado por Brendan Eich en la empresa Netscape Communications, que es la que desarrolló los primeros navegadores web comerciales.

Apareció por primera vez en el producto de Netscape llamado Netscape Navigator 2.0 en 1995, y en 1997 fue adoptado como estándar ECMA (*European Computer Manufacturers Association*) con el nombre de ECMAScript.

En los dos últimos años el uso de JavaScript ha aumentado mucho gracias al concepto *Web 2.0* o *web semántica*, donde se usa juntamente con otras tecnologías como XML para ofrecer páginas web más ricas e interactivas, lo que se conoce como tecnología AJAX.

### 2.1.5. AJAX

AJAX es el acrónimo de *Asynchonous JavaScript And XML* y es una técnica de desarrollo web para crear aplicaciones interactivas o RIA (*Rich Internet Applications*). Éstas se ejecutan en el cliente, es decir, en el navegador de los usuarios y mantiene comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre la misma página sin necesidad de recargarla, lo que aumenta la interactividad, velocidad y usabilidad en la aplicación.

AJAX es una combinación de cuatro tecnologías ya existentes:

- X/HTML para estructurar el contenido de las páginas y hojas de estilos en cascada (CSS) para el diseño de las mismas.
- DOM o *Document Object Model* accedido con un lenguaje de scripting por parte del usuario, especialmente implementaciones ECMAScript como JavaScript y JScript, para mostrar e interactuar dinámicamente con la información presentada.
- El objeto XMLHttpRequest para intercambiar datos asincrónicamente con el servidor web.
- XML es el formato usado comúnmente para la transferencia de vuelta al servidor, aunque cualquier formato puede funcionar, incluyendo HTML pre-formateado, texto plano, JSON, etc.

### 2.1.6. MySQL

MySQL es un Sistema Gestor de Bases de Datos (SGDB) relacional, multihilo y multiusuario que utiliza el lenguaje SQL (*Structured Query Language*. MySQL ha sido creada por la empresa sueca MySQL AB, recientemente comprada por Sun Microsystems a principios de 2008. MySQL es software libre pero también tiene una versión comercial que ofrece soporte, asistencia técnica y documentación a los usuarios de dicha versión.

MySQL es uno de los SGBD más populares junto a PostgreSQL y Oracle, y forma parte del concepto LAMP como se ha dicho anteriormente. Esto es así por una serie de razones:

- Es que es un sistema fácil de instalar, integrado en muchísimas distribuciones de GNU/Linux pero que también está disponible en muchos otros sistemas operativos.
- Su módulo de consultas es muy rápido.
- Está soportado de forma nativa por el lenguaje PHP.
- Soporta características avanzadas como vistas, triggers transacciones, claves foráneas, replicación, búsqueda e indexación de campos de texto, plugins, herramientas de *clustering*, soporte de XML y un largo etcétera.

### 2.1.7. LaTeX

LaTeX(escrito LaTeX en texto plano) es un lenguaje libre de marcado para documentos y un sistema de preparación de documentos, formado por un gran conjunto de macros de TeX, escritas inicialmente por Leslie Lamport (LamportTeX) en 1984, con la intención de facilitar el uso del lenguaje de composición tipográfica creado por Donald Knuth.

Es muy utilizado para la composición de artículos académicos, tesis y libros

técnicos, dado que la calidad tipográfica de los documentos realizados con LaTeX es comparable a la de una editorial científica de primera línea. La memoria de éste PFC ha sido escrita en éste lenguaje.

## 2.2. Herramientas

### 2.2.1. GNU/Linux

El sistema operativo utilizado para realizar éste PFC ha sido ha sido GNU/Linux.

GNU/Linux, también llamado *GNU con Linux*, es un sistema operativo libre tipo UNIX donde la mayoría de utilizades y bibliotecas han sido programadas por el proyecto GNU y donde el núcleo del sistema se llama Linux.

Sobre el proyecto GNU y el *software libre* ya he hablado en el capítulo de presentación.

Sobre Linux, se trata de un núcleo o *kernel* libre publicado bajo la licencia GNU GPL versión 2 escrito originalmente y dirigido por el finlandés Linus Torvalds. Linux sigue un desarrollo constante desde su creación en 1992 y del actualmente está disponible [1] la versión 2.6.25.

Figura 2.1: Tux es la mascota oficial del *kernel* Linux.

Una distribución GNU/Linux es una variante de éste sistema operativo que

[1] http://kernel.org/

incopora determinados paquetes de software y herramientas específicos que la distinguen de otras distribuciones. Existen cientos de distribuciones GNU/Linux, y el ranking de las más usadas se puede ver en el sitio web Distrowatch [2].

Para éste PFC me he decantado por usar una variante de la distribución Ubuntu llamada Kubuntu, ya que ésta última lleva el escritorio KDE en vez de GNOME. KDE o *Kool Desktop Environment* es un entorno de escritorio muy avanzado que cuenta con una serie de herramientas que me han sido muy útiles a la hora de realizar el proyecto de manera eficiente. Algunas de ellas han sido konsole, yakuake, basket, konqueror, kopete, amarok, kpdf.

### 2.2.2. Apache

Apache es un servidor web libre que funciona con el protocolo HTTP 1.1 y que soporta múltiples sistema operativos. Apache es un proyecto que se ha desarrollado dentro del proyecto HTTP Server de la Apache Software Foundation.

El desarrollo de Apache comenzó en 1995 basandose en una serie de parches sobre el servidor web NCS HTTP. Más tarde fue modificado por completo para que Apache funcionara sobre una nueva arquitectura.

La arquitectura de Apache es modular: su estructura es de un núcleo base al que se van añadiendo módulos necesarios para proveer de las funcionalidades necesarias. Quizás el módulo más usado de Apache sea el módulo PHP, que es el que se utiliza en éste PFC. Apache soporta seguiridad SSL y TLS (vía módulo *mod_ssl*), tiene un módulo proxy, un módulo de reescritura de URLs (vía *mod_rewrite*), módulos de filtraje, módulos de compresión (por ejemplo *mod_gzip*).

Apache es actualmente el servidor web más utilizado, siendo alrededor del 51 % de páginas webs servidas por éste software, según datos de Netcraft [3]

[2] http://distrowatch.com/
[3] http://news.netcraft.com/archives/2008/04/14/april_2008_web_server_survey.html

de Abril de 2008.

### 2.2.3. Vim

Vim o *Vi IMproved* es una versión mejorada del editor de texto Vi. Vi fue creado por Bill Joy en 1976 y la primera versión de Vim fue creada por Bram Moolenaar y publicada en 1991. Hay versiones de Vim disponibles para muchos sistemas operativos y se puede encontrar en casi cualquier sistema GNU/Linux, donde en muchas ocasiones se puede ejecutar a través del comando vi, que invoca a Vim a través de un enlace simbólico o un alias.

La potencia de Vim se debe a principalmente a dos características:

- Vim es modal, es decir, tiene diferentes modos: modo inserción y modo comando. La posibilidad de poner a Vim en modo comando y ejecutar acciones mediante pocas teclas es su característica principal.

- Vim es extensible. Mediante *maps* (vienen a ser *key-bindings*) combinados con la programación de funciones es posible aumentar las funcionalidades de Vim hasta el infinito y más allá. Éstas nuevas funcionalidades se empaquetan como *plugins* o *scripts* que es posible descargar desde la página oficial de Vim.

Si ya de por sí las características que lleva Vim de serie son superiores a cualquier IDE, si estamos acostumbrados a una serie de características de éstos últimos podemos bajarnos una serie de *plugins* que las implementan y otros que las superan.

Algunos de los utilizados para la realización de éste proyecto son:

- SuperTab: permite utilizar el tabulador para utilizar las funciones de autocompletado de Vim, con lista desplegable incluida.

- LustyExplorer: muestra una ventana con una lista de los posibles ficheros a editar desde el directorio de trabajo y soporta autocompletado.

- TagList: añade un panel vertical a la izquierda con la estructura del fichero de código fuente que estamos editando: *includes*, clases, variables/atributos, métodos/funciones... y permite acceder a ellas a golpe de enter.
- XHTML: permite insertar tags XHTML mediante la pulsación de un par de caracteres.
- Closetag: permite vía *"Control + _"* permite insertar el cierre del último *tag* X/HTML abierto.
- NERDCommenter: permite comentar/descomentar líneas de código, soportando decenas de lenguajes de programación.
- MultipleSearch: vía *:Search patrón* permite realizar múltiples resaltados de elementos en diferentes colores.

### 2.2.4. Navegadores

Para probar el funcionamiento correcto del código XHTML obtenido por las nuevas funciones implementadas en NWiki, se han testeado con varios navegadores:

#### Firefox

Mozilla Firefox es un navegador web libre con interfaz gráfica de usuario, desarrollado por la Corporación Mozilla y un gran número de voluntarios externos. Firefox comenzó como un derivado del programa Mozilla Application Suite, que terminó por reemplazarlo como el producto bandera del proyecto Mozilla bajo la dirección de la Fundación Mozilla. El programa es multiplataforma y está disponible para múltiples sistemas operativos: Windows, Mac OS X, GNU/Linux, FreeBSD, etc.

Firefox es el segundo navegador más utilizado después de Internet Explorer

y su uso está creciendo enormemente desde su publicación. Actualmente su uso llega al casi al 20 % global, mientras que su uso es del 40 % en Europa.

Firefox cuenta también con una serie de extensiones o *plugins* desarrollados por la comunidad y disponibles en sus web oficial. Algunos *plugins* son muy útiles para la programación web y han sido usados en éste PFC, como:

- Firebug [4]
- Web Developer [5]
- Vimperator [6]
- Split Browser [7]
- ScreenGrab [8]

Éste PFC ha sido testeado extensivamente con Firefox 2 y con Firefox 3 beta. La versión final de Firefox 3 ha sido publicada hace unos días, el 17 de Junio de 2008.

### Konqueror

Konqueror es un navegador web, administrador de archivos y visor de archivos. Forma parte oficial del proyecto KDE. Es software libre y de código abierto, y al igual que el resto de los componentes de KDE, está liberado bajo la licencia GPL. El nombre Konqueror es un juego de palabras con el nombre de otros navegadores: primero vino el Navigator (navegador), después el Explorer (explorador), y finalmente el Konqueror (conquistador). Además, sigue la convención de KDE de que los nombres de los programas contengan la letra K.

[4] https://addons.mozilla.org/es-ES/firefox/addon/1843
[5] https://addons.mozilla.org/es-ES/firefox/addon/60
[6] https://addons.mozilla.org/es-ES/firefox/addon/4891
[7] https://addons.mozilla.org/es-ES/firefox/addon/4287
[8] https://addons.mozilla.org/es-ES/firefox/addon/1146

KHTML es el motor de renderizado de Konqueror. En 2005 éste motor fue mejorado por Apple para utilizarlo en su navegador Safari bajo el nombre de WebKit, proyecto que sirve actualmente de base para muchos navegadores de escritorio y de dispositivos móviles.

Éste PFC ha usado mucho Konqueror tanto para administración de archivos como para testeo de renderizado de código XHTML.

### Opera

Opera (en inglés sin acento) es un navegador privativo creado por la empresa noruega Opera Software. La aplicación es gratuita desde su versión 8.50, habiendo sido previamente *shareware* (versión que tiene las funcionalidades limitadas a un cierto periodo de tiempo) o *adware* (versión con publicidad incluída ) y, antes de su versión 5.0, únicamente de pago.

Es reconocido por su velocidad, seguridad, soporte de estándares (especialmente CSS), tamaño reducido, internacionalidad y constante innovación. Fue el primer navegador que implementó las pestañas para la navegación de sitios web y el reconocimiento de gestos y movimientos del ratón en la navegación, siendo estas su principales características en sus primeras versiones. Opera está disponible para distintos sistemas operativos, como Windows, GNU/Linux, MacOS X, BSD y otros.

Éste PFC no ha usado extensamente Opera pero sí que se ha usado para comprobar la correcta visualización del código XHTML generado así como el correcto uso de AJAX.

### Internet Explorer

Internet Explorer es un navegador web producido por Microsoft para el sistema operativo Windows y más tarde para Apple Macintosh y Solaris Unix, estas dos últimas descontinuadas en el 2006 y 2002 respectivamente. Fue creado en 1995 tras la adquisición por parte de Microsoft del código fuente de

Mosaic, un navegador desarrollado por Spyglass, siendo rebautizado entonces como Internet Explorer.

Actualmente es el navegador de Internet más popular y más utilizado en el mundo, rebasando en gran medida a las competencias existentes, aún cuando algunas de éstas han incrementado su popularidad en los últimos años. Su popularidad es debido a que Internet Explorer es el navegador oficial de Windows, y viene incluido de fábrica en dicho sistema operativo. Al estar relacionado con el Navegador de Archivos de Windows, no es posible desinstalar esta aplicación de forma estándar.

Para éste proyecto se han usado las versiones 5.5 y 6 de Internet Explorer desde GNU/Linux mediante Wine, un programa que permite ejecutar aplicaciones Windows en un entorno GNU/Linux.

### 2.2.5. CVS

CVS (*Concurrent Versions System* o también *Concurrent Versioning System*) es una aplicación informática que implementa un sistema de control de versiones: mantiene el registro de todo el trabajo y los cambios en los ficheros (código fuente principalmente) que forman un proyecto (de programa) y permite que distintos desarrolladores (potencialmente situados a gran distancia) colaboren. CVS se ha hecho popular en el mundo del software libre y sus desarrolladores difunden el sistema bajo la licencia GNU GPL.

CVS utiliza una arquitectura cliente-servidor: un servidor guarda las versiónes actuales del proyecto y su historial. Los clientes se conectan al servidor para sacar una copia completa del proyecto. Esto se hace para que eventualmente puedan trabajar con esa copia y más tarde ingresar sus cambios con comandos GNU.

CVS se ha usado extensamente en éste PFC para acceder al código de Moodle y para acceder y modificar el código de NWiki. Ambos proyectos están al-

macenados en el sitio web SourceForge [9], ya que ofrece servicios integrados para gestionar proyectos libres como CVS, wikis, *tracker*, etc.

### 2.2.6. PHPMyAdmin

phpMyAdmin es una herramienta libre escrita en PHP con la intención de manejar la administración de MySQL a través de páginas web, utilizando Internet. Actualmente permite crear y eliminar Bases de Datos, crear, eliminar y alterar tablas, borrar, editar y añadir campos, ejecutar cualquier sentencia SQL, administrar claves en campos, administrar privilegios, exportar datos en varios formatos y está disponible en 50 idiomas. Se encuentra disponible bajo la licencia GPL.

Este proyecto se encuentra vigente desde el año 1998, siendo el mejor evaluado en la comunidad de descargas de SourceForge como la descarga del mes de diciembre del 2002.

Se ha utilizado en éste PFC para realizar acciones con la base de datos que necesitan Moodle y NWiki para poder funcionar, así como para consulta rápida de datos.

### 2.2.7. Keynote

Keynote es un software privativo creado por Apple que sirve para crear presentaciones, al estilo Presenter de OpenOffice o PowerPoint de Microsoft. Forma parte de la suite iWork en conjunto con los programas Pages (procesador de texto) y Numbers (hoja de cálculo). Funciona solamente en versiones modernas de MacOS X y su última versión es de Agosto de 2007.

Keynote ha sido el software utilizado para hacer la presentación en la defensa del proyecto ante el tribunal. Se ha utilizado éste programa por la alta

[9] http://sourceforge.net/

calidad de sus presentaciones: texto en alta resolución y *anti-aliasing*, transiciones 3D entre transparencias, transiciones 3D en objetos internos de cada página, visualización de las notas en pantalla mientras en el proyector sale solamente la presentación, etc. También porque al contar con un ordenador portátil Apple MacBook Pro con mando a distancia resulta más fácil hacer la presentación en entorno Mac.

# Capítulo 3

# Desarrollo técnico

En este capítulo se explican las nuevas funcionalidades desarrolladas a lo largo del PFC. Se comentan para cada una principalmente su diseño e implementación, así como otras cuestiones técnicas relacionadas con cada una.

La explicación del diseño de las funcionalidades se acompaña con imágenes de las funcionalidades integradas actualmente en la última versión de NWiki para Moodle 1.9, cuyo código está disponible en el sitio web Crom [1], hospedado por UPCNet.

El hecho de que prácticamente no se use *Orientación a Objetos* (excepto para la clase WikiStorage) indica la ausencia de diagramas de clases y de casos de uso típicos de proyectos basados en éste paradigma de programación, siendo el usado el paradigma de *programación imperativa*.

En cuanto a la implementación en sí, se explican las funcionalidades y algoritmos en alto nivel, ya que si se explicara a nivel de código PHP la memoria sería demasiado larga.

En cuanto a la organización temporal de cada una, esto se explica en el Capítulo 4: *Planificación* en la página 95.

[1] http://morfeo.upc.es/crom/course/view.php?id=4

## 3.1. Fase previa

Aunque no se trata de una funcionalidad en sí es importante resaltar el trabajo de análisis de las tecnologías a usar en el proyecto realizado antes de empezar en el desarrollo de las funcionalidades.

El primer paso fue repasar las tecnologías del PFC con las que hacía tiempo que no trabajaba:

- Repaso del protocolo HTTP y de los lenguajes X/HTML, CSS y JavaScript.
- Instrucciones más comunes del SGBD MySQL: cómo realizar *querys* con *select, insert, update, joins* etc.
- Instalación y configuración del servidor web Apache en Kubuntu.
- Cómo descargar desde CVS la última versión de Moodle, y cómo instalarlo y configurarlo.
- Cómo interactuar con el sistema de control de versiones CVS para poder recibir y enviar cambios en el código de NWiki.

También tuve que aprender las tecnologías con las que nunca habría trabajado:

- PHP. Para ello leí el libro "*Web Database Applications with PHP & MySQL*", de la editorial O'Reilly, así como diversos tutoriales en Internet y la página oficial de PHP (http://php.net).
- AJAX. Para aprender AJAX me documenté con el libro "*Begginning AJAX with PHP: from Novice to Professional*" de la editorial Apress, así como múltiples ejemplos de funcionalidades AJAX típicas en Internet.

Después tuve que familiarizarme con el código de Moodle y sobretodo con su

API. Moodle es un sistema muy bien documentado y en la web MoodleDocs [2] hay mucha información para programadores. El foro de programación oficial de Moodle [3] también ha servido para resolver dudas.

Más tarde tuve que probar las funcionalidades de NWiki para ver cómo funcionaba desde el punto de vista de usuario.

Por último tuve que analizar el código de NWiki: arquitectura, estructuración del código, clases importantes, funcionalidades, API... La documentación y tutoriales del sitio web Crom [4] del DFWikiTeam me ayudaron a realizar éste trabajo de investigación. El foro oficial de la wiki de Moodle [5] también me sirvió para preguntar y resolver dudas.

## 3.2. Edición y visualización por secciones

### 3.2.1. Introducción y objetivos

El primer conjunto de funcionalidades que se ha desarrollado en éste PFC ha sido a la su vez la parte más compleja y larga de todas, y se ha resuelto en varias etapas. Además hay varias sub-funcionalidades relacionadas con ésta que han sido implementadas y que se comentan también.

Como se ha indicado anteriormente, una wiki no es más que un conjunto de páginas wiki relacionadas mediante enlaces internos. Una página wiki típica se puede estructurar en secciones y subsecciones, como se puede ver en la figura 3.1.

La edición de páginas wiki por secciones de manera independiente es una característica importante para una wiki. Normalmente la mayoría de personas que han utilizado alguna wiki alguna vez ha sido por usar Wikipedia y el

[2] http://docs.moodle.org/
[3] http://moodle.org/mod/forum/view.php?id=55
[4] http://morfeo.upc.es/crom/course/view.php?id=11
[5] http://moodle.org/mod/forum/view.php?id=1952

*software* sobre el que corre, MediaWiki, soporta edición por secciones (ver figura 3.1).

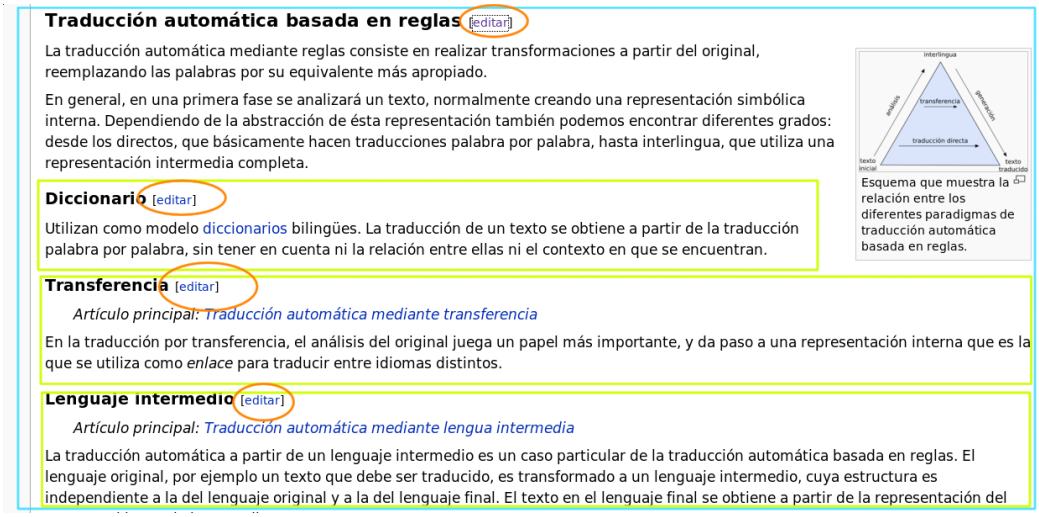



Figura 3.1: *Edición por secciones en MediaWiki.*

En la figura 3.1 se pueden ver un recuadro azul con el contenido de la sección *Traducción automática basada en reglas* y tres subsecciones de ésta en recuadros verdes. Junto a cada título (ver círculos naranjas) de sección existe un enlace que conduce a la página donde se edita el contenido de cada una de ellas respectivamente.

Así pues, un objetivo del proyecto es implementar en NWiki una funcionalidad similar para poder editar las secciones independientemente.

También se quiere ofrecer la posibilidad de poder visualizar el contenido las secciones por separado. Para ello es necesario crear un tipo de enlace interno a secciones de una página wiki que conduzca a esa sección en concreto.

### 3.2.2. Edición por secciones

#### Diseño

Las características a diseñar en ésta funcionalidad han sido:

1. Añadir un enlace en la parte derecha de las cabeceras de las secciones para posibilitar la edición de las mismas (ver en figura 3.2 el recuadro rojo).

2. Modificar la función que se encarga de mostrar el contenido de la página a editar en la pestaña de edición de NWiki para que sólo mostrara el contenido de la sección a editar, incluidas las subsecciones dentro de ella (ver figura 3.3, recuadro azul). Además mostrar también el nombre de la sección que se está editando al lado del nombre de la página (misma figura, recuadro verde).

3. Modificar la función que guarda el contenido de una página para sólo actualizar la sección que se está editando y dejar el resto de la página igual.

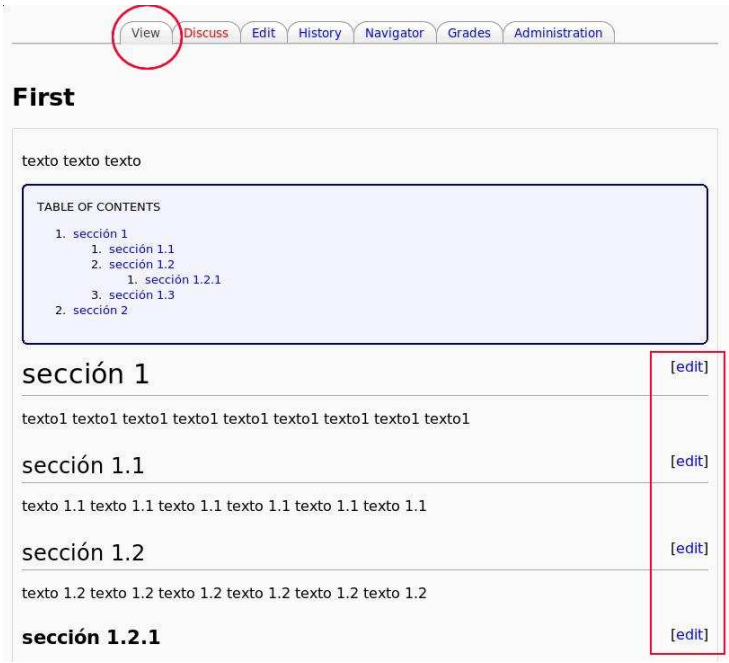



Figura 3.2: *Pestaña de visualización de una página de NWiki que contiene varias secciones y subsecciones.*


Figura 3.3: *Pestaña de edición de una página de NWiki donde se muestra el contenido de una sección y sus subsecciones*

### Implementación

La implementación de éstas funcionalidades ha ido cambiando a lo largo del tiempo debido a las limitaciones encontradas después en la fase de testeo a la hora de adquirir una identificación única para las secciones. Éstas limitaciones se han dado al comprobar casos extremos como que una página wiki cuente con más de una sección con el mismo nombre, o que se añadan o eliminen secciones al editarse éstas por el mismo u otros usuarios.

*Caso general*

Los algoritmos realizan estos pasos:

- En el parser (fichero wiki/nwikiparser.php) y en la función que analiza la sintaxis correspondiente a las cabeceras de secciones, añadir el código XHTML y CSS que crea un enlace [edit] a la derecha de cada una de ellas. El enlace contiene un identificador de la sección.

- Al clickar en un enlace se procede a la página de edición donde se recoge por el método *GET* del protocolo HTTP y mediante las funciones que dispone la API de Moodle para ello (función optional_param() el identificador de la sección. Se llama a la función implementada wiki_split_sections(), la cual analiza el contenido de la página y divide el

texto en tres partes:

1. La parte previa a la sección a editar, si existe.

2. La parte referente a la sección a editar y sus subsecciones, si tiene.

3. La parte posterior a la edición a editar, si existe.

El algoritmo recorre el contenido línea por línea, ya que las cabeceras de las secciones se encuentran en líneas ellas solas. Al detectar una línea con formato de cabecera, se procede a su análisis.

- Al guardar la página (botón Save) se llama a la función wiki_edit_treatment() a la que se le ha añadido una llamada a una nueva función, wiki_join_sections(), la cual se encarga de actualizar el contenido antiguo de la sección con el nuevo. Posteriormente se pasa al *tab* de visualización donde se pueden ver los cambios realizados.

- Ésta funcionalidad no interfiere con la edición de páginas que había en NWiki hasta ahora, sino que sólo la complementa.

*Implementación por nombre de sección*

La primera implementación realizada ha sido utilizando como identificador de una sección el nombre de la misma.

El problema de esta aproximación es el caso en que hay varias secciones con el mismo nombre, cosa no habitual pero que puede producirse. El algoritmo en éste caso seleccionaría el contenido de la primera sección encontrada, por lo que las otras que tuvieran el mismo nombre nunca podrían ser editadas independientemente.

*Implementación por nombre y número de sección*

La segunda aproximación fue añadir el número de sección además del nombre. Esto hizo que funcionara en el caso de que varias secciones tuvieran el mismo nombre, pero fallaba en un caso. Si varios usuarios editaban secciones de una misma página y añadían o eliminaban alguna entonces los números de sección

utilizados quedaban inconsistentes y el algoritmo de unión del contenido nuevo de sección con el contenido de la página no funcionaba correctamente.

Se aprovechó ésta indentificación por número para cambiar los enlaces de la *Table Of Contents* (TOC) de las páginas wiki para que al clickar en ellos el *scroll* funcionara aunque hubiera dos secciones con el mismo nombre. Por ejemplo, en la figura 3.4 se ven varias secciones con nombre secA y clickando en el quinto enlace de la TOC se desplazaría el navegador hasta la tarcera sección llamada secA y no a la primera.

*Implementación por* hash *MD5*

Por último, y gracias a la recomendación de Jordi Piguillem, se procedió a añadir una identificación mediante un *hash* MD5.

Un *hash* MD5 o *Message-Digest algorithm 5* es un algoritmo usado en criptografía cuya entrada es transformada en una cadena de texto de 128 bits o 32 caracteres hexadecimales, de tal manera que no existen dos entradas tales que el resultado del algoritmo sobre esas entradas devuelva la misma salida.

La API de PHP provee la función md5(), la cual es utilizada para calcular el *hash* del nombre más el contenido de cada sección para así tener un identificador único de cada una.

Se utiliza ésto en las funciones wiki_split_sections() y wiki_join_sections() para dividir una sección (y subsecciones) de una página del resto de la página y para unir el nuevo contenido de una sección (aunque se hayan añadido o eliminado mientras secciones) a una página, respectivamente.

De todas maneras se sigue manteniendo el identificador de nombre de sección para mostrarlo en la pestaña de edición de páginas/secciones como se puede ver en la figura 3.3.

#### Resultados

Al ser NWiki un módulo utilizado por varias universidades (UPC incluida) en la actualidad, ésta funcionalidad ha sido probada extensamente por muchos usuarios, que han informado de los problemas que han tenido. En la actualidad el código es sólido y no han habido problemas en éste aspecto.

En cuanto a eficiencia los algoritmos implementados para esta funcionalidad son de coste lineal con el temaño de la página wiki, siendo el coste $\theta(N)$ con N el número de líneas de la página wiki.

### 3.2.3. Visualización por secciones

#### Diseño

Ésta funcionalidad ha consistido en dos partes:

1. Crear un tipo de enlace interno a sección de página.
2. Cambiar la lógica relacionada con la visualización de una página para permitir visualizar sólo una sección (y subsecciones, si tiene) de ésta.

*Enlaces internos a secciones*

En los sistemas wiki como NWiki hay dos tipos de enlaces: internos a páginas wiki y externos a páginas de Internet. Los primeros se crean mediante corchetes dobles [[nombrePáginaWiki]] y los segundos mediante corchetes simples [http://www.google.com].

La funcionalidad implementada añade a los enlaces internos a páginas wiki una parte para referenciar una sección de una página, siguiendo una sintaxis como la siguiente:

- [[nombrePáginaWiki♯nombreSección]]: con una almohadilla (♯) se crea un enlace tal que al clickar en él se procede a la visualización de la

página *nombrePáginaWiki* y se hace un *scroll* vertical hasta la sección *nombreSeción*.

- [[nombrePáginaWiki##nombreSección]]: con dos almohadillas (##) se crea un enlace tal que al clickar en él se procede únicamente a la sección (y subsecciones, si tiene) *nombreSección* de la página *nombrePáginaWiki*.

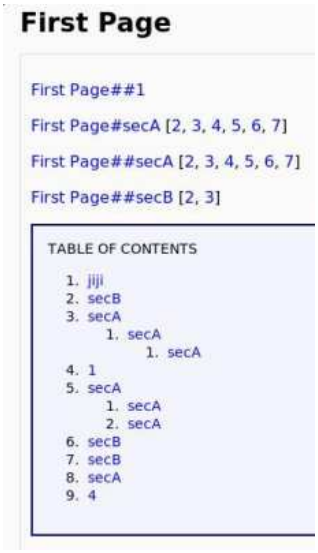



Figura 3.4: *Links internos y* Table Of Contents *en NWiki.*

También se quiere:

- Que si se enlaza a una sección de una página no existente en tipo de enlace se visualice en rojo, como pasa con los enlaces a páginas que no existen.

- Que los enlaces internos funcionen con la funcionalidad *synonyms* o sinónimos, que permite que varios nombres apunten a la misma página.

- Que si se crea un enlace a una sección de una página tal que tiene varias secciones con ése mismo nombre, se visualicen en forma de lista de enlaces.

*Visualización de secciones*

Consiste en cambiar el código que muestra el contenido de una página (pestaña *View* en NWiki) para que sólo muestre el contenido de una sección de página. Además se visualizar el nombre de la sección que se está viendo.

### Implementación y resultados

*Enlaces internos a secciones*

Para implementar éste tipo de enlaces se ha modificado el parser nwiki, en concreto la función parse_internal_link() del fichero wiki/nwikiparser.php para comprobar si se trata de un enlace a páginas o a secciones.

En el caso de secciones, se ha creado la función wiki_section_exists() que comprueba si existe una sección en una determinada página wiki. Si existe en enlace se renderiza en color azul, y si no existe en color rojo.

También se añade una comprobación en la función wiki_page_exists() para comprobar si el nombre de la página es un sinónimo de la página.

Para la funcionalidad de enlazar a secciones de una página tal que en esa misma página existen múltiples secciones con el mismo nombre se ha procedido a modificar la función *wiki_view_page_url()* para detectar éste caso. Dicha función devuelve un *string* XHTML con uno o más enlaces a dichas secciones. Ésta última funcionalidad en concreto se puede ver en la figura 3.4, donde existen múltiples secciones llamadas secA y secB de la página First Page.

El resultado del resto de funcionalidades se puede ver en la figura 3.5.

En ésta figura se pueden ver diferentes elementos:

- Existen dos páginas en esa wiki, First Page con dos secciones secA y secB y Second Page con una sección secB.
- Existen enlaces internos a sección de página con *scroll* y a visualización de sección (con uno y dos ♯ respectivamente).
- Hay siete enlaces internos:


Figura 3.5: *Links internos a páginas y secciones en NWiki.*

1. Enlace interno a página First Page. Como dicha página existe, sale en azul.

2. Enlace interno a sección secA de página First Page. Como existe la página y también la sección, sale en azul.

3. Enlace interno a sección secB de página First Page. Como existe la página y también la sección, sale en azul.

4. Enlace interno a sección secC de página First Page. Como existe la página y pero no la sección, sale en rojo.

5. Enlace interno a página Second Page. Como dicha página existe, sale en azul.

6. Enlace interno a sección secA de página Second Page. Como existe la página y pero no la sección, sale en rojo.

7. Enlace interno a sección secB de página Second Page. Como existe la página y también la sección, sale en azul.

*Visualización de secciones*

Para implementar esta funcionalidad se ha añadido a la función parse_nwiki_text() del fichero wiki/nwikiparser.php una llamada a la función wiki_split_sections() para, en el caso de una visualización de una sección de página, capturar el contenido de esa sección y mostrarla en vez de todo el texto.

El resultado se puede ver en la figura 3.6.

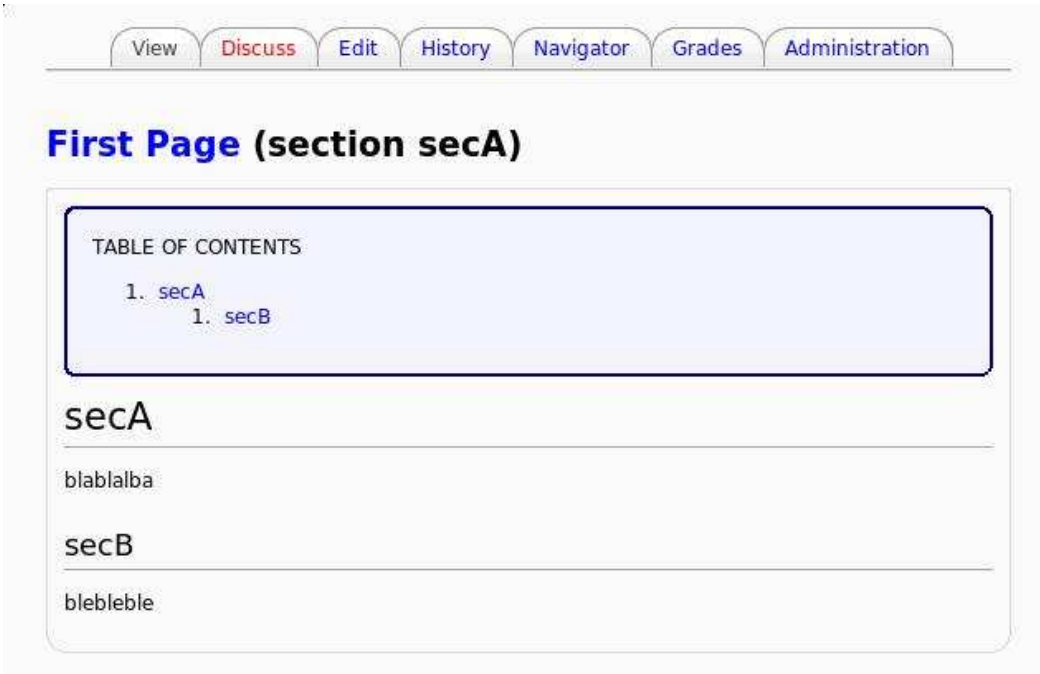



Figura 3.6: *Visualización de una sección en NWiki.*

## 3.3. Sistema de bloqueos para ediciones concurrentes

### 3.3.1. Introducción y objetivos

Antes de la implementación de edición por secciones, en NWiki existían bloqueos de página para evitar que dos usuarios pudieran editar la misma página a la vez. Al agregar edición por secciones se quiere ampliar el sistema para permitir bloqueos o *locks* a nivel de secciones.

### 3.3.2. Diseño

El sistema de bloqueos funciona de ésta manera:

- Se comprueba si el elemento a editar está disponible (no hay ningún elemento que le bloquee).
- Si no lo está, se muestra un mensaje de advertencia explicando qué elementos están bloqueando al elemento a editar.
- Si lo está, se bloquea y se procede a la edición. Mientras se está editando se refresca el bloqueo cada X tiempo, siendo por defecto 120 segundos. Éste *refresco* del bloqueo se hace vía AJAX.

Así pues, el sistema a extender debe tener en cuenta la jerarquía página - sección - subsección de tal manera que no no debe permitir que:

- Si un usuario edita una página, que otro edite esa misma página.
- Si un usuario edita una página, que otro edite una sección de esa misma página.
- Si un usuario edita una sección de una página, que otro edite esa misma página.

- Si un usuario edita una sección de una página, que otro usuario edite esa sección de esa página.
- Si un usuario edita una sección de una página, que otro usuario edite una sección de esa página tal que esa sección contenga a la sección editada por el primer usuario (sección padre).
- Si un usuario edita una sección de una página, que otro usuario edite una sección de esa página tal que esa sección sea una subsección de la que se está editando por el primer usuario.

En cualquier otro caso la edición concurrente de secciones de una misma página que no colisionen (no tengan relación padre-hijo con la página ni con una sección de ella) ha de ser posible. En caso de que haya colisiones se mostrará una tabla con la información sobre los bloqueos que evitan una determinada edición.

Además se quiere dar la posibilidad de que los profesores y administradores de la wiki puedan sobreescribir los bloqueos y poder editar en cualquier momento cualquier página y/o sección de ella. Esta característica se llama *sobreescritura de bloqueos* o *lock overriding*.

### 3.3.3. Implementación

Lo que se ha hecho es modificar las funciones existentes wiki_lock_message(), wiki_obtain_lock() (para crear bloqueos) y wiki_release_lock() (para eliminar bloqueos) existentes para trabajar con secciones mediante funciones nuevas como wiki_is_page_locked(), wiki_is_section_locked() y sobretodo wiki_related_sections() que se encargan de comprobar si una página o sección está bloqueada o no, según los casos comentados en la parte de diseño.

También se ha tenido en cuenta que puede haber diferentes instancias de una misma wiki para diferentes estudiantes y grupos de estudiantes tales que tienen mismos nombres de página y secciones a la hora de editarlas. Por ello se ha tenido que cambiar la tabla wiki_locks de la base de datos de NWiki

para añadir los campos que identifican a los estudiantes y grupos, ownerid y gropid respectivamente.

Para el requerimiento de los bloqueos puedan ser sobreescritos por administradores o profesores se ha creado la función wiki_override_info(), la cual añade un botón que, en caso de aviso de bloqueo, permite eliminar los bloqueos que evitan una determinada edición del profesor/administrador. En caso de eliminarlos, se llama al fichero overridelock.php que ha sido modificado para permitir la eliminación de múltiples bloqueos en el caso de que haya varias secciones de una página siendo editadas por alumnos.

### 3.3.4. Resultados

Aquí se pueden ver una serie de imágenes que muestran colisiones por bloqueos de páginas y secciones:

- En la figura 3.7 se puede ver cómo el sistema bloquea la edición de la página nuevapagina porque hay secciones (diez, en concreto) de esa página que están siendo editadas.

- En la figura 3.8 se puede ver cómo el sistema bloquea la edición de la sección 1.1.1 porque ya está siendo editada.

- En la figura 3.9 se puede ver cómo el sistema bloquea la edición de la sección 1.1 porque hay 3 subsecciones de ésta sección que están siendo editadas.

- En la figura 3.10 se puede ver cómo el sistema bloquea la edición de la sección otra porque ésta es una subsección de una sección que está siendo editada.

En todas estas figuras podemos ver quién está editando las páginas o secciones que producen el bloqueo, cuándo empezaron a editarse y cuándo fue la última vez que se refrescó el bloqueo.

Además, como las imágenes se han tomado desde un usuario administrador, el sistema ofrece la posibilidad de eliminar los bloqueos mediante el botón Override these locks que se encuentra debajo de la tabla de bloqueos.

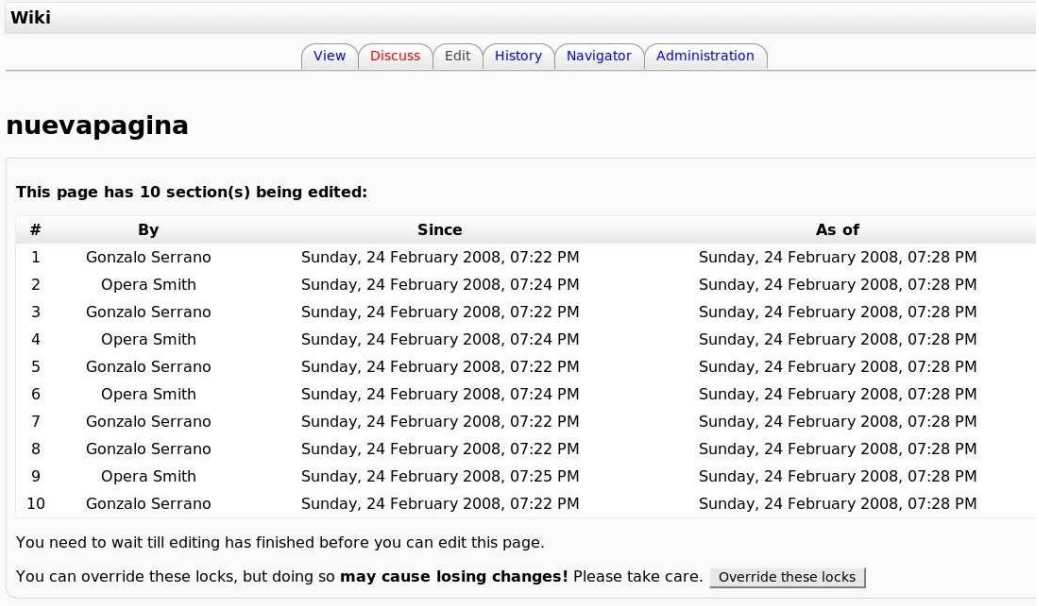



Figura 3.7: *Bloqueo de página por secciones.*

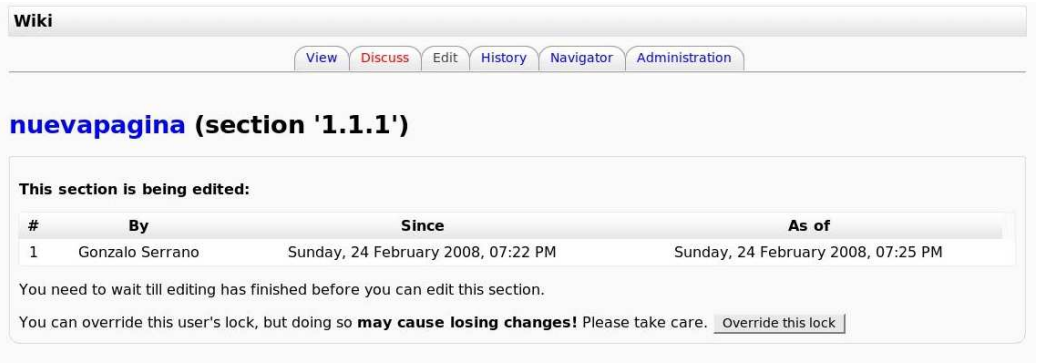



Figura 3.8: *Bloqueo de sección porque ya está siendo editada.*


Figura 3.9: *Bloqueo de sección por subsecciones.*

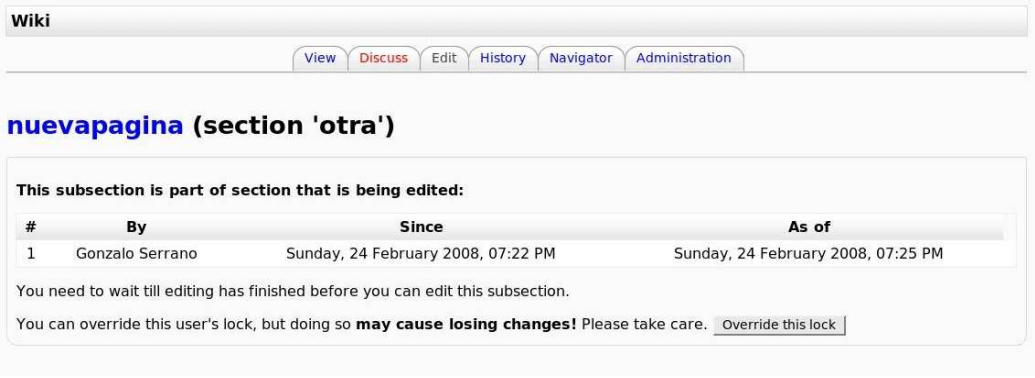



Figura 3.10: *Bloqueo de sección por sección padre.*

## 3.4. Mejoras en el parser nwiki

### 3.4.1. Introducción y objetivos

Cuando se habla de *parsing* en computación se refiere al proceso de analizar un texto como secuencia de unidades básicas llamadas *tokens* y comprobar si esa secuencia sigue una determinada gramática incontextual.

En el contexto de NWiki se trata de convertir ciertas secuencias de palabras que siguen una sintaxis wiki (por ejemplo, el *token* [[página]] (que indica un enlace interno) a código XHTML interpretable por un navegador web (en el ejemplo se convertiría en un enlace <a href="url">etiqueta</a>).

NWiki soporta múltiples *parsers* que dotan a esta wiki de gran versatilidad. Existe un *parser* compatible con la wiki original de Moodle, ewiki, otro compatible con el editor HTML de Moodle, otro compatible con DFWiki y finalmente está el nwikiparser o parser de NWiki, que es el más moderno y con el cual se trabaja en el PFC ya que es el parser por defecto que se usa desde Moodle 1.8 en NWiki.

El objetivo es mejorar el parser:

- Añadir enlaces internos a secciones (esto se ha ha explicado ya en secciones anteriores).
- Resolver problemas de seguridad del tipo *SQL injection* o inyección de código SQL.
- Mejorar el código de la *sintaxis Creole*.

### 3.4.2. Diseño e Implementación

#### Seguridad

Consiste en analizar el código del parser en busca de vulnerabilidades de tipo *SQL Injection*. Éste tipo de vulnerabilidades permiten ejecutar código SQL malicioso si no se *escapan* (filtran) determinados caracteres considerados *peligrosos*.

El ejemplo típico es el siguiente: se quieren consultar los datos de un usuario tal que el nombre del usuario es un parámetro dado.

El código dispone de una variable para crear la sentencia SQL:

```
$consulta = "SELECT * FROM usuarios WHERE nombre = '"
            + nombreUsuario + "';"
```

Si el parámetro introducido es Alicia entonces no hay problema ya que la consulta queda:

```
SELECT * FROM usuarios WHERE nombre = 'Alicia';
```

En cambio, si se intruduce un parámetro malicioso a propósito como Alicia'; DROP TABLE usuarios; SELECT * FROM datos WHERE nombre LIKE '% entonces la consulta queda:

```
SELECT * FROM usuarios WHERE nombre = 'Alicia';
DROP TABLE usuarios;
SELECT * FROM datos WHERE nombre LIKE '%';
```

Dicha consulta lo que hace es:

- Consultar los datos del usuario Alicia.
- Eliminar la tabla de usuarios (!).
- Consultar los datos de otra tabla, en este caso tabla *datos* que puede tener información confidencial (!).

La solución es utilizar correctamente las funciones de la API de PHP y de la API de Moodle que impiden este tipo de comportamientos, escapando los caracteres peligrosos.

Moodle utiliza una API genérica para el acceso a base de datos que necesita que los parámetros sean escapados según si se trata de funciones de consulta, inserción o actualización de datos. Esto se hace mediante dos funciones, add_slashes() (escapa caracteres) y strip_slashes() (los desescapa), como muestra la figura 3.11 que se encuentra en la página de documentación de Moodle para desarrolladores [6].

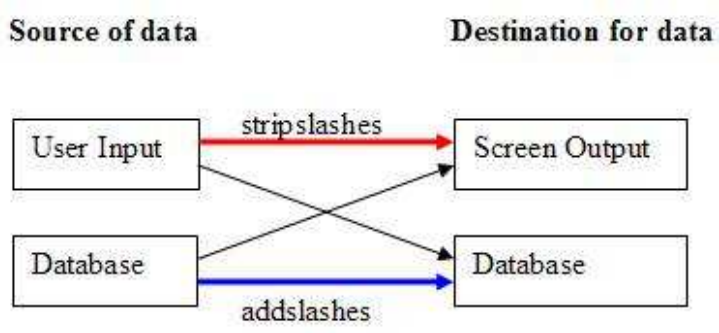



Figura 3.11: *Uso de add_slashes() y strip_slashes() en Moodle.*

También es necesario escapar caracteres conflictivos cuando se pasan parámetros mediante el método GET del protocolo HTTP, ya que los valores se pasan por la URL de la página. Para solucionarlo hay que escapar esos valores con la función de PHP urlencode().

La implementación ha consistido en analizar el código del parser en busca de éste tipo de vulnerabilidades y añadir llamadas a éstos tres tipos de funciones en los casos necesarios.

[6] http://docs.moodle.org/en/Developer:Slashes

### Sintaxis Creole

La Sintaxis Creole forma parte del proyecto WikiCreole [7], el cual busca unificar las diferentes sintaxis wiki que usan los diferentes proyectos wiki en busca de una sintaxis general que pueda ser usado por todos ellos.

En el caso de NWiki consiste en implementar una serie de funciones que soporten éste tipo de marcas Creole, modificando el fichero wiki/nwikiparser.php para ello.

Las nuevas marcas Creole que soporta NWiki son:

- Negrita: **palabra**
- Cursiva: //palabra//
- Salto de línea: línea1\\línea2
- Salto de línea a principio de línea:

```
línea1
%%%línea3
```

- Imágen: {{Imagen.jpg|Titulo de la imágen}}
- Tablas:

```
|=Heading Col 1 |=Heading Col 2         |
|Cell 1.1       |Two lines\\in Cell 1.2 |
|Cell 2.1       |Cell 2.2               |
```

La implementación se ha realizado añadiendo transformaciones directas mediante expresiones regulares para las marcas de negrita, cursiva y salto de línea, y con funciones nuevas parse_creole_table() y parse_creole_image() para

[7] http://www.wikicreole.org/

las tablas e imágenes respectivamente. Éstas dos últimas funciones se encargan de recrear el código XHTML correspondiente mediante algoritmos iterativos.

### 3.4.3. Resultados

Los resultados de éstas dos implementaciones han sido satisfactorios y se han probado casos extremos para comprobar que los algoritmos son correctos.

En la figura 3.12 se puede comprobar cómo las marcas creole se renderizan perfectamente al hacer un *preview* en la pestaña de edición de Nwiki.

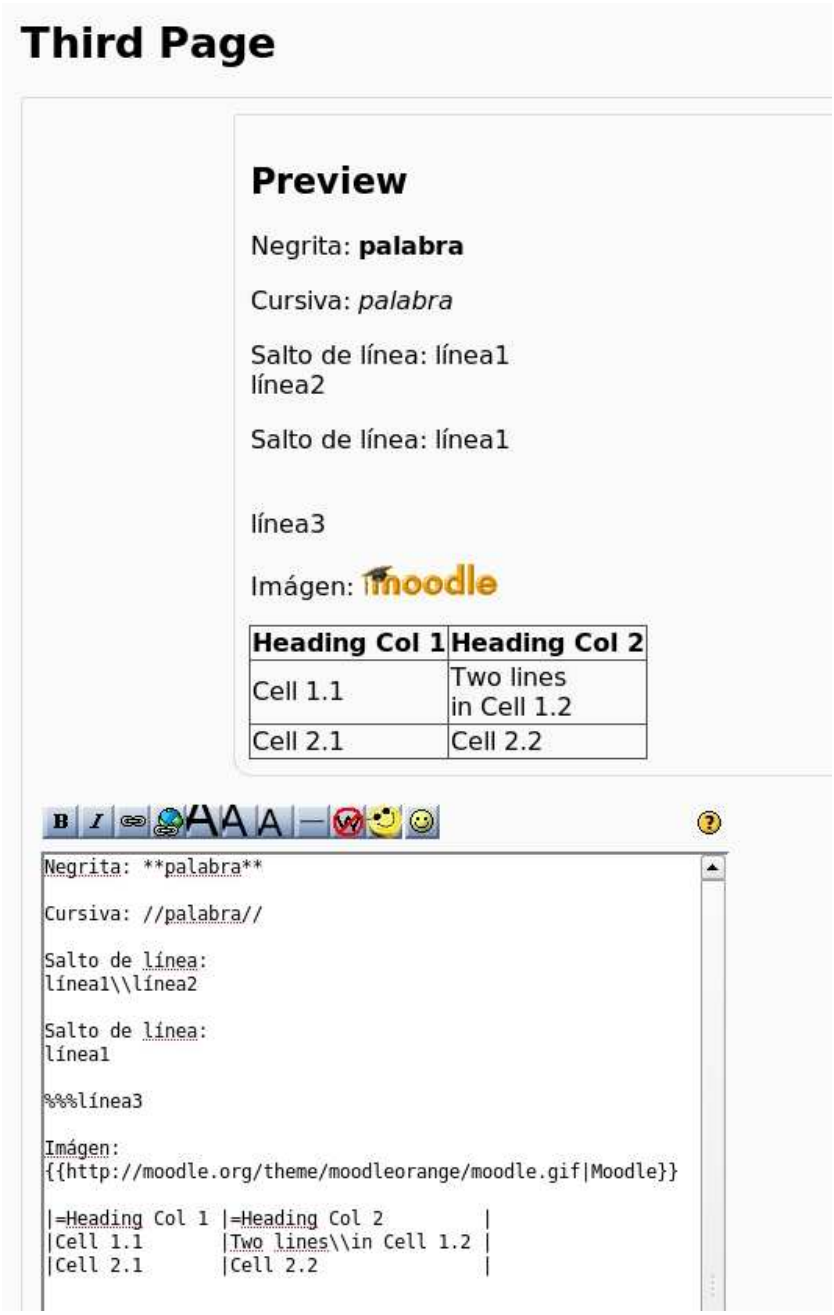



Figura 3.12: *Preview de la sintaxis Creole en el parser* nwiki.

## 3.5. *Backport* a Moodle 1.8

### 3.5.1. Introducción y objetivos

Durante el desarrollo de las funcionalidades explicadas hasta ahora hubo un cambio de versión de Moodle, que pasó de la 1.8 a la 1.9 en Marzo de 2008. Éste cambio no fue brusco si no que hubo una serie de versiones 1.9 *beta* anteriores a la versión final.

La nuevas funcionalidades se desarrollaron sobre versiones *beta* de 1.9, pero como aún no había salido la versión final se decidió publicar una *release* o versión de NWiki para Moodle 1.8 a principios de 2008. Era necesario para ello pasar el código de la versión más nueva de Moodle, la 1.9, a Moodle 1.8, lo que se conoce en informática como *backport*.

Un *backport* (portar hacia atrás) no es más que eso, parchear una versión antigua de un software con los cambios de una versión más moderna.

### 3.5.2. Diseño, implementación, resultados

El diseño de NWiki para Moodle 1.8 es exactamente el mismo que había en las funcionalidades de la versión de NWiki para Moodle 1.9, y es el que se ha comentado en los capítulos anteriores.

En cuanto a la implementación, pasar del código de 1.9 a 1.8 no ha sido siempre trivial ya que el código de Moodle había cambiado y también el tema de librerías dinámicas que en la versión de 1.8 de NWiki no estaban y sí en NWiki 1.9.

El resultado es que tanto en las versiones de NWiki para Moodle 1.8 como para Moodle 1.9 se cuentan con las características de edición y visualización por secciones, bloqueos a nivel de sección y mejoras en el parser y la sintaxis Creole.

## 3.6. Mejoras en el Wikibook

### 3.6.1. Introducción y objetivos

Ésta funcionalidad consite en arreglar una serie de errores en el subproyecto Wikibook.

WikiBook es un proyecto del DFWikiTeam integrado en NWiki que consiste en utilizar las páginas wiki de una manera especial para que darle un formato tipo libro, con un índice de páginas-capítulo y una secuencia de páginas wiki a la cual se accede mediante enlaces especiales hacia la siguiente y anterior página del libro.

### 3.6.2. Diseño, implementación y resultados

Los errores arreglados han sido:

- Error en la página índice cuando no hay texto antes de la primera entrada.
- Errores en los *strings* de previous y next página.
- Al guardar una página del wikibook no se vuelve a la visualización de tipo wikibook.
- Los *fakechapters* o capítulos falsos (sirven como índice de sub-capítulos del libro) muestran contenido que no deben.

Para implementar estos *bugfixes* se han hecho una serie de cambios en el fichero locallib.php y sobretodo en el fichero wiki/wikibook.php, creando la función extra wiki_is_node_leaf().

### 3.6.3. Resultados

A continuación se muestran imágenes del wikibook mejorado.

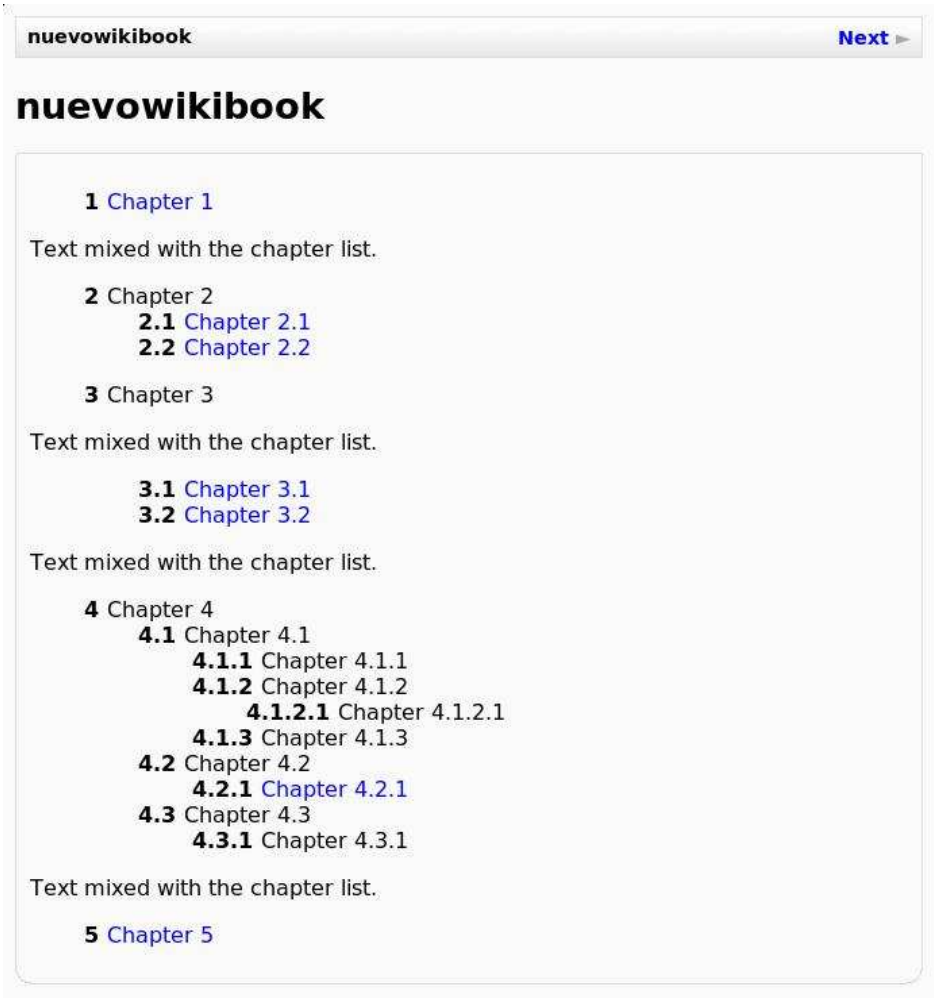



Figura 3.13: *Página índice del wikibook con texto entre capítulos. Algunos capítulos y subcapítulos tienen páginas* fake chapter *que sirven de índice de sub-capítulos, como por ejemplo el 2, 3, 4, 4.1, 4.2, 4.3, etc.*


Figura 3.14: *Segunda página del wikibook, que es la primera página del primer capítulo y que no tiene* fake chapter.


Figura 3.15: *Tercera página del wikibook, que es la primera página del segundo capítulo. Tiene* fake chapter *y por eso muestra los sub-capítulos del capítulo 2..*

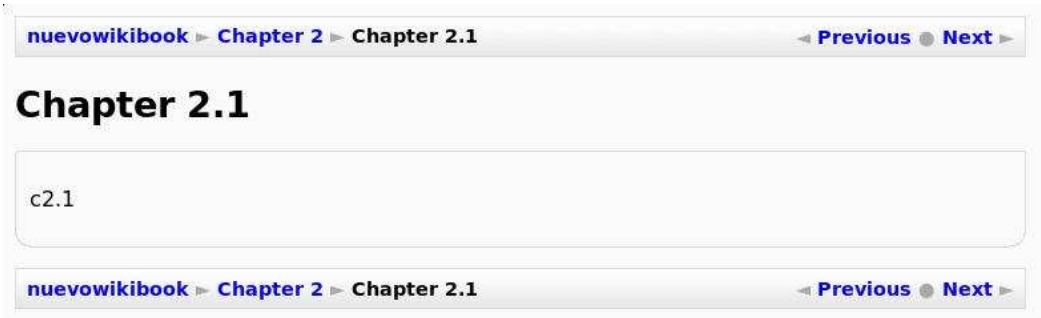



Figura 3.16: *Cuarta pagina del wikibook, que es la segunda página del segundo capítulo y a su vez la primera del subcapítulo 2.1.*

## 3.7. Sistema de etiquetado o *tagging*

### 3.7.1. Introducción y objetivos

Una etiqueta o *tag* no es más que una palabra o conjunto de palabras que expresan un concepto.

La versión 1.9 de Moodle integra un sistema de *tags* con los que se pueden etiquetar intereses en el perfil de los usuarios y también entradas de los *blogs* de Moodle.

El objetivo de esta funcionalidad es dotar a las páginas wiki de NWiki de un sistema de etiquetado similar al de usuarios y entradas de blogs que utilice la API de Moodle para ello. Además se quiere contar con un bloque Moodle que contenga una nube de etiquetas o *tag cloud* que muestre todos los tags de wikis.

Además se quiere que estas funcionalidades sean lo más independiente posibles del núcleo de NWiki para que se puedan añadir y quitar fácilmente.

### 3.7.2. Diseño

El diseño del sistema de *tags* en páginas wiki se divide en varias partes:

1. Añadir y eliminar *tags* desde la página de edición de wikis (edit tab) mediante un nuevo campo de texto.

2. Ver los *tags* de la wiki desde la página de visualización de wikis (view tab).

3. Al clickar en un *tag* se muestra una página con todas las páginas wiki que contienen ese *tag*, y además para ese tag muestra una serie de herramientas de administración.

4. En cuanto al bloque de nube de etiquetas wiki se quiere hacer un diseño parecido al del bloque de etiquetas general de Moodle.

### 3.7.3. Implementación

Para que estas funcionalidades sean independientes del *core* de Moodle se ha creado un directorio tags/ dentro del directorio base de NWiki que contiene dos ficheros:

1. tags.lib.php: corresponde a las funcionalidades que van a permitir añadir, eliminar y visualizar etiquetas de wikis, así como una función para generar la nube de tags. Utiliza extensivamente la API de etiquetas de Moodle que se encuentra en moodle/tag/lib.php. El listado de funciones implementadas es el siguiente:

   - wiki_tags_save_tags(): recibe una lista de tags separa por comas y la guarda en base de datos.
   - wiki_tags_print_editbox(): renderiza el código XHTML necesario para generar el campo de texto para insertar nuevos *tags* en la página de edición de wikis.
   - wiki_tags_print_viewbox(): renderiza el código XHTML necesario para visualizar los *tags* de una wiki desde la página de visualización de wikis.
   - wiki_tags_get_all_wikitags(): devuelve la lista de todos los campos de todos los *tags* wiki disponibles.
   - wiki_tags_get_tag_names(): devuelve un *string* con los nombres de todos los *tags* wiki separados por comas.
   - wiki_tags_get_tag_links(): devuelve el código XHTML que genera una lista de enlaces a *tags* wiki.
   - wiki_tags_get_wikipages(): devuelve una lista con todas las páginas

wiki que contienen un determinado *tag*.

- wiki_tags_get_wikipages_list(): recibe una lista de páginas wiki y devuelve una lista XHTML en forma de árbol con las wikis y las páginas wiki que tienen un determinado *tag*.
- wiki_tags_get_wikipages_table(): recibe una lista de páginas wiki y devuelve una tabla XHTML con las wikis y las páginas wiki que tienen un determinado *tag*.
- wiki_tags_print_tag_cloud(): devuelve un *string* XHTML con la nube de *tags* de tipo wiki.

2. view.php: corresponde a la página que visualiza el contenido de un *tag*, esto es, la lista de páginas wiki que contienen ese *tag*. Para ello utiliza las funciones implementadas en tags.lib.php.

### 3.7.4. Resultados

Aquí se pueden ver una serie de imágenes con los resultados obtenidos.


Figura 3.17: *Cuadro de inserción con tres* tags *en la página de edición de páginas wiki.*

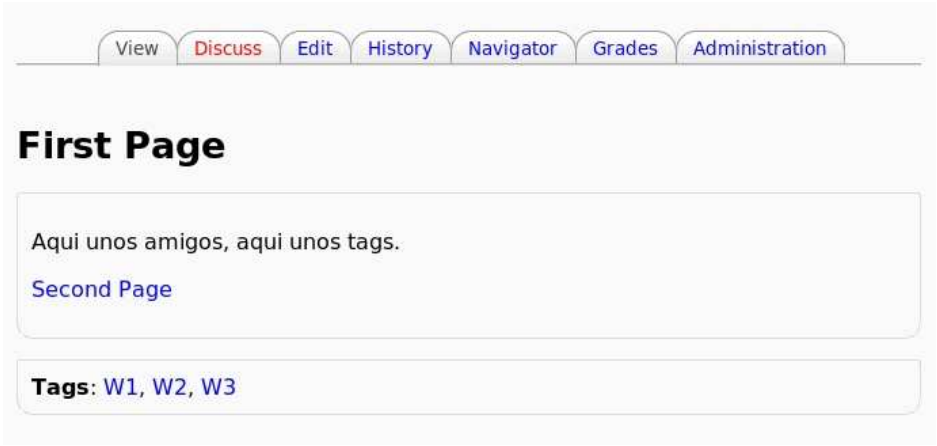



Figura 3.18: *Cuadro de visualización de tres* tags *en la página de visualización de páginas wiki.*


Figura 3.19: *Página de visualización de todas las wiki y sus páginas tales que tienen el* tag *W1, así como opciones de administración sobre ese* tag.


Figura 3.20: *Bloque de nube de etiquetas wiki o* wiki tag cloud.

## 3.8. Arreglos en el sistema de notas o *grades*

### 3.8.1. Introducción y objetivos

Moodle 1.9 incorpora un sistema nuevo para poner notas o *grades* a alumnos que realicen ciertas actividades.

Durante estos últimos meses otro proyectista de Marc Alier estuvo trabajando en la integración del sistema de notas en NWiki para poder evaluar los trabajos que hacen los alumnos con wikis. Por motivos de falta de tiempo dicho proyectista realizó una primera implementación funcional pero no tuvo tiempo de refinarla ni de crear una interfície acorde con los requisitos de Moodle.

El sistema de notas en NWiki funciona básicamente así:

- El profesor o administrador crea una escala de notas para la actividad o elige una existente.
- El profesor o administrador elige el tipo de evaluación a los alumnos:
  - Sin evaluación.
  - Evaluación sólo por profesores.
  - Evaliación por profesores y alumnos (*Peer Review*).
- Los usuarios permitidos ponen nota a las wikis desde la página de visualización del contenido de las wikis. Junto con la nota, se puede poner un comentario o *feedback* justificando esa nota.
- Opcionalmente se puede poner nota a las ediciones parciales de la wiki accediendo a las versiones antiguas desde la pestaña *History*.
- Los profesores o administradores acceden a una página especial donde se muestra información de alumnos y las notas de las wikis con las que

han trabajado y ponen una nota a cada alumno. Dicha nota se integra con el sistema de notas general de Moodle.

El objetivo es estudiar el código realizado por dicho proyectista, arreglar los problemas funcionales que se encuentren y crear nuevas funcionalidades (como por ejemplo la posibilidad de que salga la nota puesta a una wiki en la página Discussion) y una nueva interfaz gráfica para interactuar con el sistema de notas de NWiki.

### 3.8.2. Bugfixes

#### Diseño e implementación

Durante el análisis del código se han encontrado una serie de errores que se han ido arreglando a medida que se encontraban. Algunos de los arreglos o *bugfixes* han sido, entre otros:

- Eliminar echos() sobrantes.
- Eliminar vulnerabilidades del tipo *SQL injection*.
- Arreglado el combo de usuarios.
- Se permiten usar escalas globales de notas y no sólo de la wiki o del curso.
- Se guarda correctamente la configuración de la evaluación de la wiki.
- El cuadro de poner notas ahora solo sale en la página de visualización de notas y no en otras páginas.
- Añadida la lógica para gestionar las funcionalidades separated syudents y separated groups.
- Añadido un *combo box* para cambiar de usuario si está activado el *student mode* o *group mode*.

- Arreglada la función que busca la última página de una página de grupo.

También se ha reestructurado mucho todo el código para hacerlo más acorde con los estándares Moodle. Ahora todas las funciones empiezan por wiki_grade_ y se ha añadido información PHPDoc en las cabeceras de las funciones.

#### Resultados

El resultado de esto es que el sistema de notas funciona correctamente aunque con la interfaz gráfica antigua.

### 3.8.3. Añadir nota a página de discusión

NWiki cuenta con una pestaña o página de discusión donde los usuarios pueden comentar sobre los contenidos de la wiki o cualquier otro tema.

El director del proyecto sugirió la posibilidad de poner añadir automáticamente a la página de discusión la nota que se pone a una wiki en un determinado momento. Se quiere saber qué usuario ha puesto la nota, qué nota ha puesto, qué nota tenía la wiki y qué comentario lleva adjunto la nota. También se quiere dar la posibilidad de no salga el nombre del usuario que ha puesto la nota sino que salga como anónimo.

#### Diseño, implementación y resultados

El diseño e implementacion consisten en añadir *checkboxes* junto al cuadro donde se ponen las notas en las páginas wiki de tal manera que si están seleccionadas las opciones de añadir a la página de discusión (*Send feedback as a comment to Discussion Page* y evaluación anónima (*Anonymous evaluation* se llama a la función wiki_grade_append_to_discussion() del fichero

grades/grades.lib.php que se encarga de recuperar el cotenido de la página de discusión de esa wiki y añadir al final la nueva información.

A continuación se pueden ver las imágenes con la inserción de notas y la página de discusión con comentarios de notas:

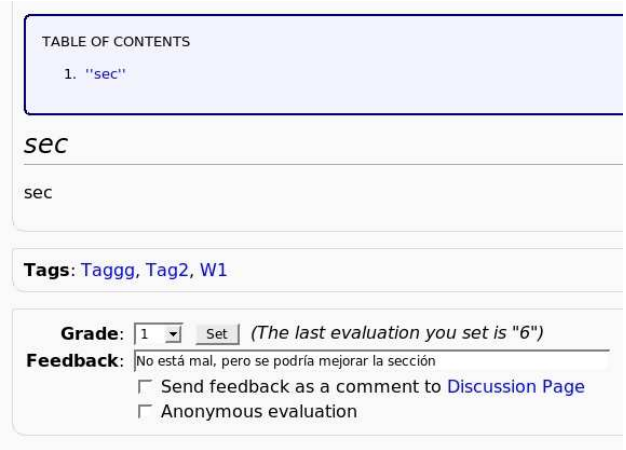



Figura 3.21: *Seleccion de opciones para añadir la nota de una wiki a la página de discusión.*

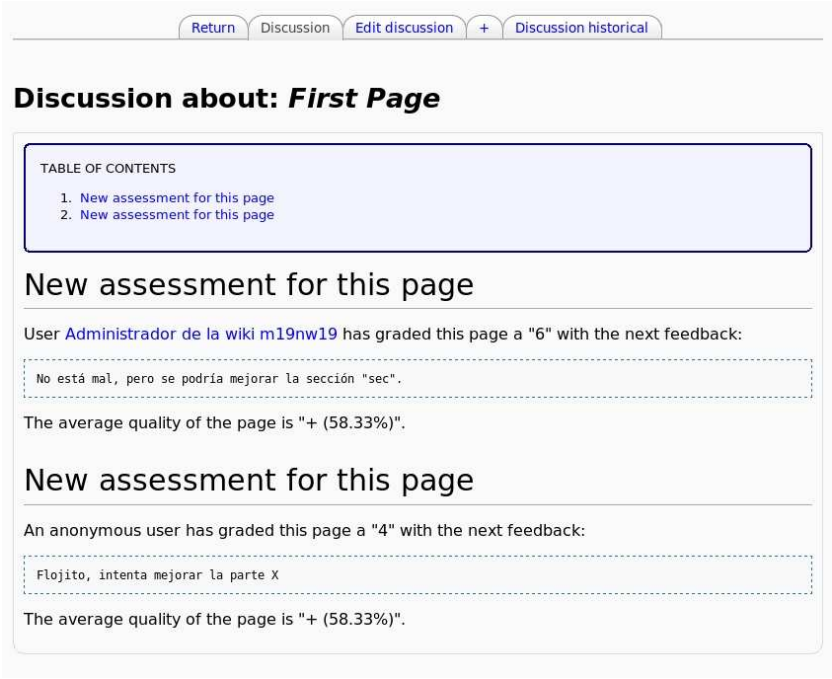



Figura 3.22: *Página de discusión con dos evaluaciones, una de un usuario administrador y otra de un usuario anónimo.*

### 3.8.4. Nueva interfaz gráfica

El director del proyecto decidió que el sistema de notas era una característica importante para NWiki y en la cual había que dedicar más tiempo, por lo que mandó modificar toda la interfaz gráfica existente y añadir nuevas características.

#### Diseño, implementación y resultados

Para implementar la nueva interfaz se ha cambiado extensamente el código del fichero grades/grades.evaluation.php y se han creado muchas funciones nuevas en grades/grades.lib.php para separar las funcionalidades. Algunas de estas funciones nuevas son:

- wiki_grade_append_to_discussion()
- wiki_grade_get_num_userpages()
- wiki_grade_get_sql_filter()
- wiki_grade_get_user_info()
- wiki_grade_get_userpages()
- wiki_grade_get_users_info()
- wiki_grade_get_wikigrade()
- wiki_grade_got_permission()
- wiki_grade_item_exist()
- wiki_grade_print_edition_evaluation_box()
- wiki_grade_print_page_evaluation_box()
- wiki_grade_print_tables()

- wiki_grade_print_user_info()
- wiki_grade_print_user_picture()
- wiki_grade_print_user()

A continuación se puede ver una comparativa entre una imagen de la interfaz antigua y varias imágenes de la nueva.

La imágen izquierda muestra la interfaz antigua para poner nota a una página wiki y a la derecha la interfaz nueva:


Figura 3.23: *La imágen izquierda muestra la interfaz antigua para poner nota a una página wiki y a la derecha la interfaz nueva.*

Aquí se muestra como era la antigua interfaz gráfica que muestra las notas de las páginas de la wiki:

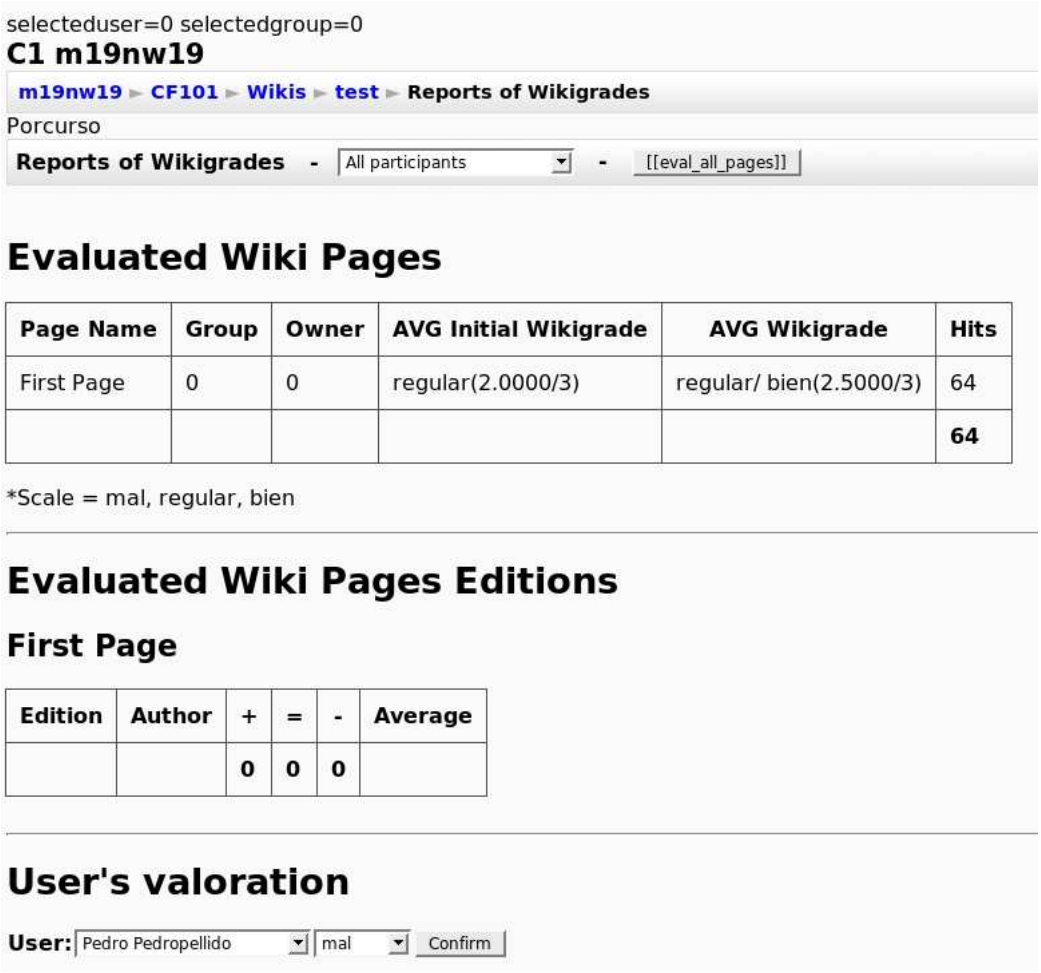



Figura 3.24: *Antigua interfaz de muestra de notas de páginas wiki*

Y aquí se muestra como es la nueva interfaz gráfica:

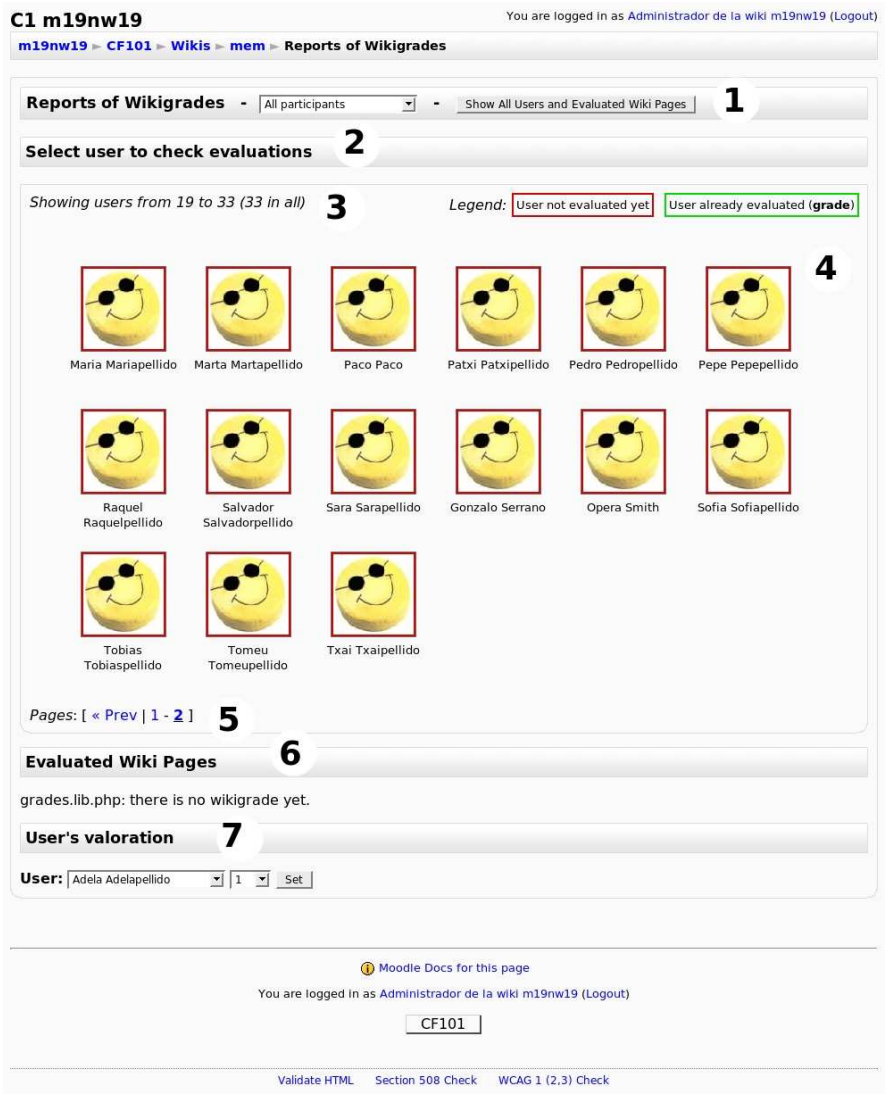



Figura 3.25: *Nueva interfaz de muestra de notas de páginas wiki.*

La nueva interfaz se integra mejor con la interfaz propia de Moodle (*headers*, tablas Moodle, etc).

Además, se divide en 3 partes:

- La parte superior (Select user to check evaluations muestra la nueva

interfaz para seleccionar usuarios. Al clickar en un usuario nos muestra cómo ha trabajado en las páginas wiki y que notas tiene.

- La parte intermedia muestra información sobre las páginas de la wiki a las que se ha puesto nota, si hay alguna.

- En la parte inferior se muestra el cuadro para poner nota a los alumnos en esa wiki según las notas de las páginas wiki en las que ha participado.

Algunos de sus elementos (con número en la imagen) se comentan a continuación:

1. Éste botón sirve para volver a la pantalla general de notas si está seleccionado algún usuario (no lo está en este caso pero si en la figura 3.26.

2. Recuadro de selección de usuarios. Como se ve en la leyenda, los usuarios tales que su fotografía o *avatar* están rodeados de un recuadro rojo son aquellos a los que aún no se ha puesto nota, mientras que los que están envueltos en un un recuadro verde sí que se les ha puesto nota (la nota saldría al lado del nombre, como se puede ver en la figura 3.27).

3. Texto que muestra el número de usuarios del curso que se están visualizando y cuántos hay en total.

4. Los usuarios salen todos en rojo porque ninguno tiene una nota asignada aún.

5. El recuadro de usuarios soporta paginación para el caso de que haya muchos usuarios. En la imagen se está viendo la segunda página, del 19 al 33.

6. Recuadro que muestra las notas de las páginas wiki. Como no hay ninguna nota puesta aún, sale vacío.

7. Recuadro para poner nota global de la wiki a los alumnos.

En la siguiente figura se muestra la pantalla que aparece al clickar en uno de los estudiantes, bien desde la lista del *combo box* o bien desde el *avatar* del estudiante.

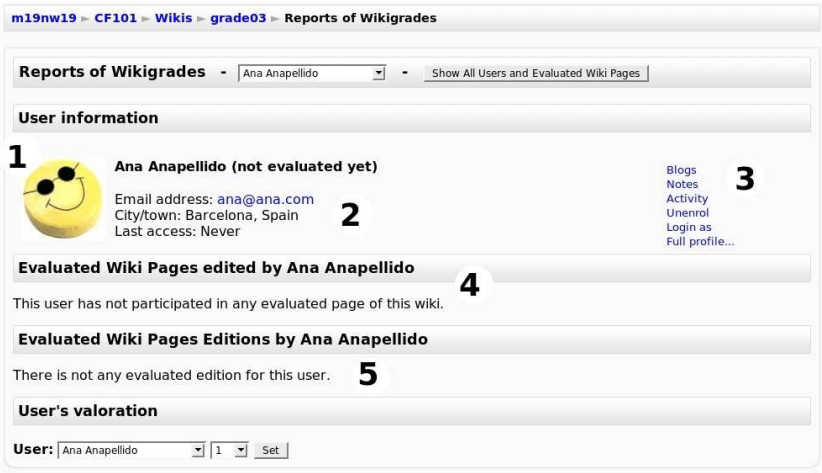



Figura 3.26: *Página de información de las notas de las páginas en las que ha participado el estudiante seleccionado.*

En la figura están señalados los siguientes elementos:

1. La foto o *avatar* del alumno.

2. Su nombre y apellido, la nota que tiene (si tiene) y otros datos de interés.

3. Enlaces a otras actividades del alumno (blog, notas), así como herramientas de administración (quitar al alumno de la actividad wiki, acceder a la wiki como el propio usuario) y un enlace al perfil general del usuario.

4. Páginas wiki con nota puesta en las que ha participado el usuario. En el ejemplo no sale ninguna, más adelante veremos otra imagen donde sí salen.

5. Ediciones wiki del usuario con nota puesta. Aún no hay ninguna.

Aquí podemos ver una pantalla de notas con páginas y usuarios ya evaluados:

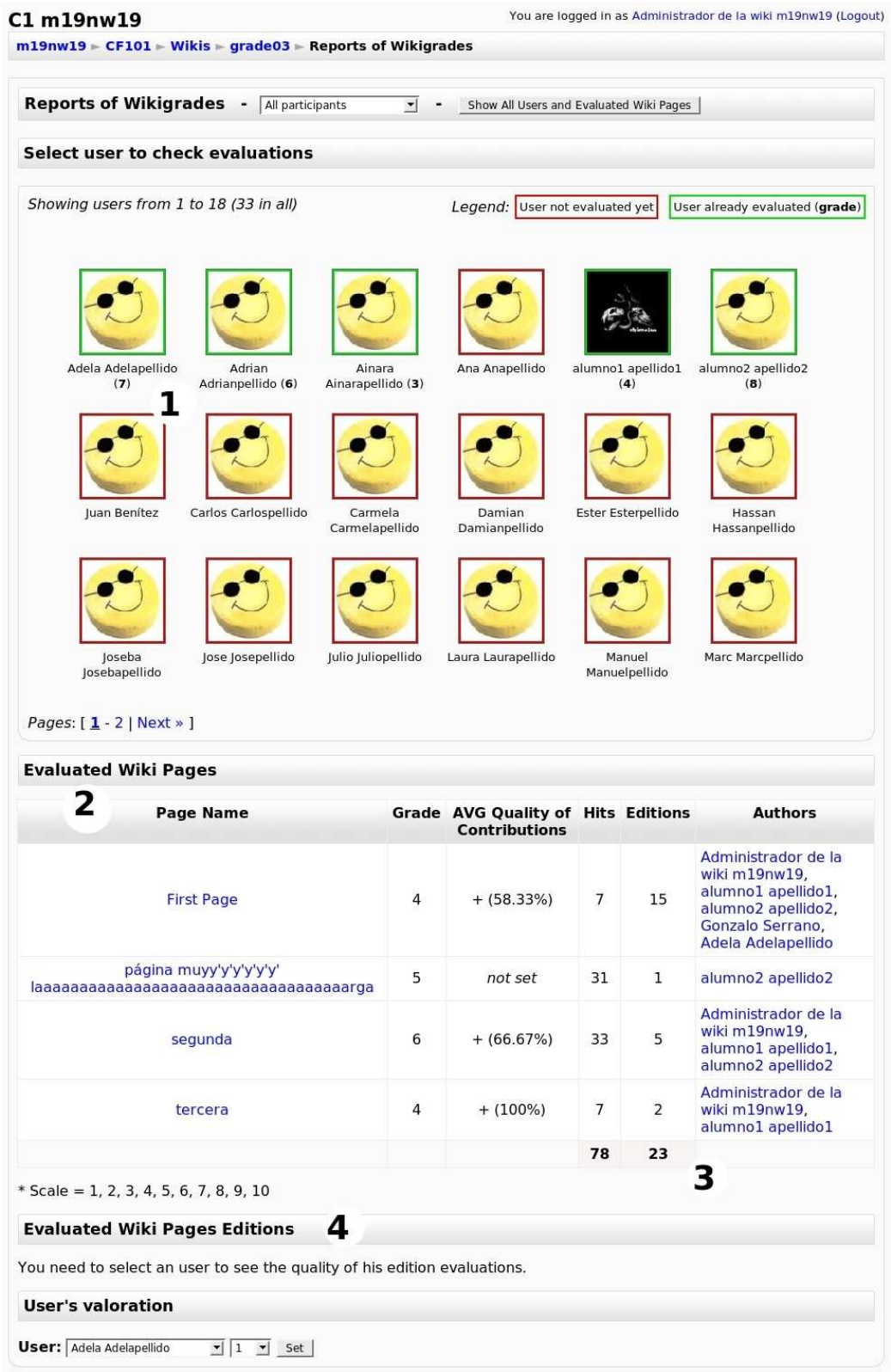



Figura 3.27: *Página que muestra las notas de páginas wiki con varias evaluaciones de usuarios y páginas.*

En la figura están señalados los siguientes elementos:

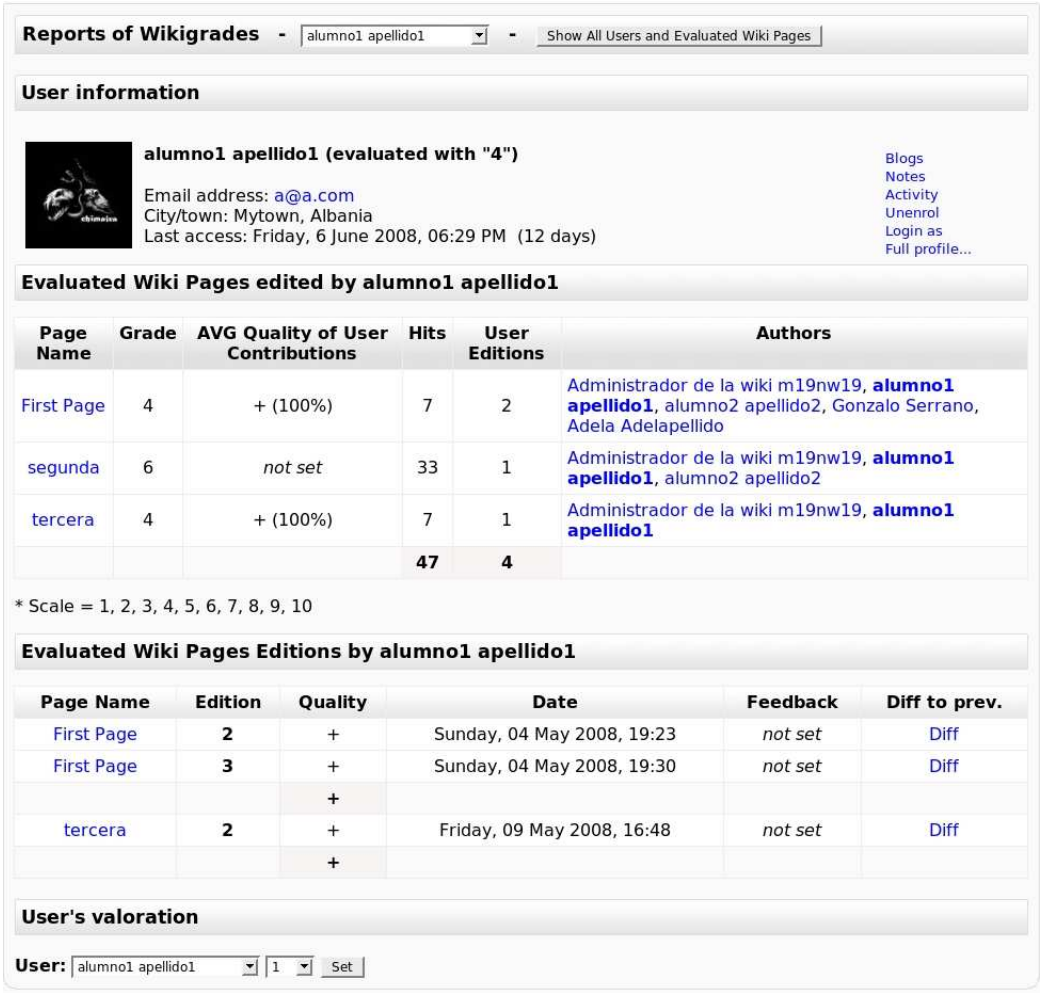
1. Aquí ya vemos usuarios evaluados (en verde) junto con su nota entre paréntesis, y otros no evaluados.

2. En éste cuadro podemos ver ya wikis evaluadas con ciertas notas. En la tabla salen las columnas con el nombre de la página de la wiki, la nota de la página, la calidad media de las ediciones (positiva, igual, negativa) y su porcentaje respecto al total, el número de visualizaciones de la página (*hits*), el número de ediciones de la página y quienes son los autores que han contribuido a la página.

3. Aquí podemos ver un sumatorio de las columnas de *hits* y *editions*.

4. Para ver las notas de las ediciones de las páginas hay que seleccionar primero un alumono, y así lo muestra el mensaje de aviso.

Por último podemos ver cómo es la pantalla que muestra la nota de un usuario y la información y notas de las páginas y ediciones de páginas en las que ha participado:



**Reports of Wikigrades** - alumno1 apellido1 - Show All Users and Evaluated Wiki Pages

**User information**

**alumno1 apellido1 (evaluated with "4")**

Email address: a@a.com
City/town: Mytown, Albania
Last access: Friday, 6 June 2008, 06:29 PM (12 days)

Blogs
Notes
Activity
Unenrol
Login as
Full profile...

**Evaluated Wiki Pages edited by alumno1 apellido1**

| Page Name | Grade | AVG Quality of User Contributions | Hits | User Editions | Authors |
|---|---|---|---|---|---|
| First Page | 4 | + (100%) | 7 | 2 | Administrador de la wiki m19nw19, **alumno1 apellido1**, alumno2 apellido2, Gonzalo Serrano, Adela Adelapellido |
| segunda | 6 | *not set* | 33 | 1 | Administrador de la wiki m19nw19, **alumno1 apellido1**, alumno2 apellido2 |
| tercera | 4 | + (100%) | 7 | 1 | Administrador de la wiki m19nw19, **alumno1 apellido1** |
| | | | **47** | **4** | |

* Scale = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10

**Evaluated Wiki Pages Editions by alumno1 apellido1**

| Page Name | Edition | Quality | Date | Feedback | Diff to prev. |
|---|---|---|---|---|---|
| First Page | **2** | + | Sunday, 04 May 2008, 19:23 | *not set* | Diff |
| First Page | **3** | + | Sunday, 04 May 2008, 19:30 | *not set* | Diff |
| | | **+** | | | |
| tercera | **2** | + | Friday, 09 May 2008, 16:48 | *not set* | Diff |
| | | **+** | | | |

**User's valoration**

**User:** alumno1 apellido1 1 Set

Figura 3.28: *Página de información de las notas de las páginas en las que ha participado el estudiante seleccionado.*

Como se puede ver, hay información destacable como:

- Al lado del nombre del alumno ahora sale su nota.
- La tabla central muestra las páginas en las que ha participado el alumno, e información como:
  - El nombre de la página.
  - La nota de la página.
  - La calidad media de sus contribuciones a la página (positiva, igual, negativa).
  - Los *hits* de la página.
  - Las ediciones de ese usuario en esa página.
  - Todos los autores de la página y el usuario seleccionado en negrita.
- En la penúltima tabla se puede ver la nota de las ediciones del alumno. Para cada página evaluada, la información es la siguiente:
  - Nombre de la página.
  - Número de edición de página evaluada.
  - Calidad de la evaluación (positiva, igual, negativa).
  - La fecha de la edición.
  - El comentario de la nota de la edición, si existe.
  - Un enlace para ver las diferencias con la anterior edición.
  - Además, al final de cada conjunto de ediciones de página sale una media de calidad de cada página.
- Ya en la última se pueden ver los elementos necesarios para poner nota a un alumno según sus aportaciones.

## 3.9. Cambios en la pestaña *History*

### 3.9.1. Introducción y objetivos

Aprovechando los cambios en el sistema de notas se quieren realizar una serie de cambios en la pestaña de historia. La pestaña historia es una página de NWiki que muestra información

1. Quitar una serie de columnas que no interesan, agregar otras y cambiarlas de orden para tener la información mejor estructurada.

2. Agregar la nota de la página.

3. Agregar una interfaz para ver los cambios entre ediciones de manera similar a MediaWiki.

### 3.9.2. Diseño e implementación

Para cambiar las columnas y agregar y eliminar otras se ha modificado el fichero wiki/hist.php, siendo el trabajo bastante trivial.

Para añadir la nota a la página se han utilizado en ese mismo fichero llamadas a las funciones del fichero *grades/grades.lib.php*, en concreto a la función wiki_grade_get_wikigrade().

Lo más complejo ha sido añadir la interfaz basada en *radio buttons* XHTML y JavaScript. Para ello he estudiado el código de MediaWiki, y en concreto su JavaScript, y lo he transformado para que funcione en la interfaz de NWiki, mediante las funciones que se encuentran en el nuevo fichero wiki/hist.js.

### 3.9.3. Resultados

El resultado es una interfaz como la siguiente:

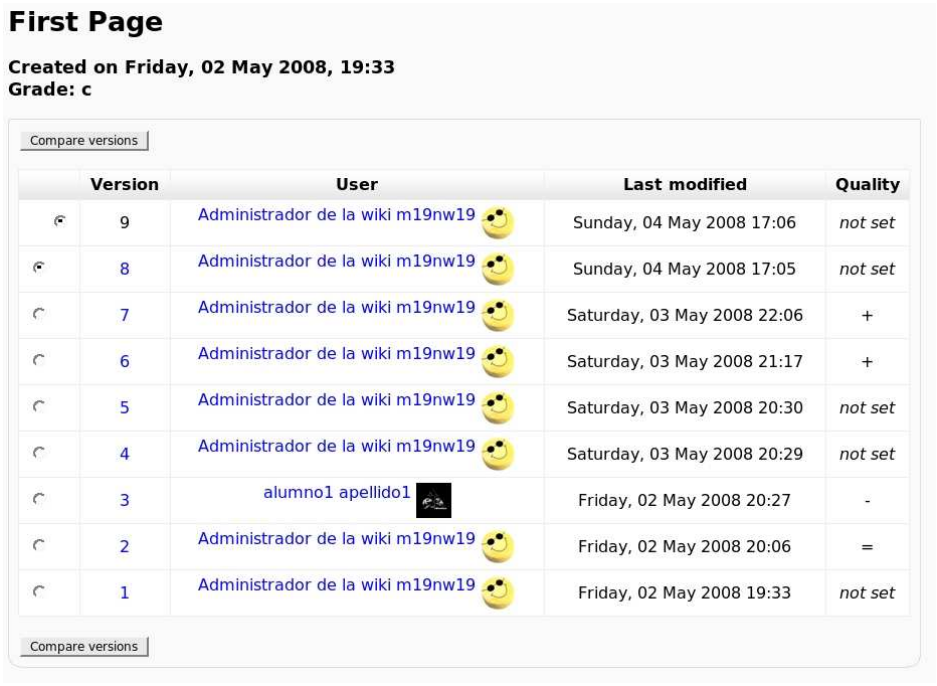



Figura 3.29: *Pestaña* history *de Nwiki*

En la figura podemos ver, entre otras cosas:

- El nombre de la página, la fecha de creación y la nota de la página.
- Dos botones para comparar las versiones seleccionadas (arriba y abajo).
- El número de versión de la página o número de edición.
- El usuario que creó esa versión.
- Cuándo se creó la versión.
- Qué calidad tiene la versión, si se ha especificado.

# Capítulo 4

# Planificación

En este capítulo se muestra una descripción de las fases del proyecto, el diagrama de Gantt de las fases del proyecto y finalmente un análisis de costes.

## 4.1. Fases del proyecto

El proyecto se ha dividido en diferentes fases, según funcionalidad. continuación se muestra el listado de fases, su duración, quién la ha ejecutado y cuánto ha costado la fase.

- Aprender PHP y MySQL: 29 días, estudiante, sin coste.
- Etapa inicial:
  - Crear blog: 1 día, estudiante, sin coste.
  - LAMP Setup:
    - Apache y MySQL: 1 día, administrador de sistemas, 60 euros.
    - PHP: 1 día, administrador de sistemas, 60 euros.
    - IDE: 2 días, administrador de sistemas, 120 euros.

        - CVS Moodle y NWiki: 1 día, administrador de sistemas, 60 euros.

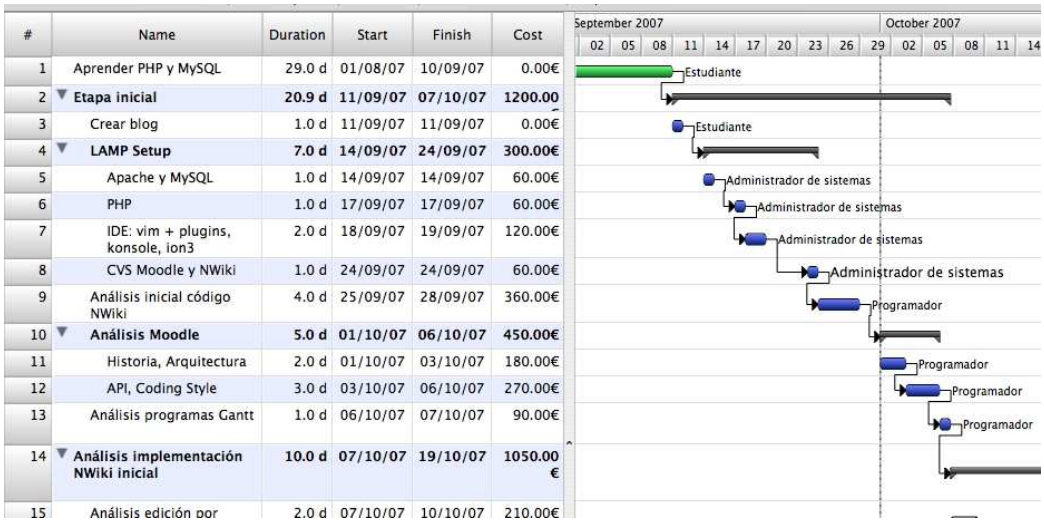
    - Análisis código NWiki: 4 días, programador, 360 euros.
    - Análisis Moodle:
        - Historia, arquitectura: 2 días, programador, 180 euros.
        - API, *Coding Style*: 3 días, programador, 270 euros.
    - Análisis programas Gantt: 1 día, programador, 90 euros.

- Análisis implementación NWiki inicial:
    - Análisis edición por secciones: 2 días, Jefe de proyecto, 210 euros.
    - Análisis bloques Moodle: 2 días, Jefe de proyecto, 210 euros.
    - Análisis bloque y *tag cloud*: 2 días, Jefe de proyecto, 210 euros.
    - Análisis AJAX: 2 días, Jefe de proyecto, 210 euros.
    - Análisis *tags*: 2 días, Jefe de proyecto, 210 euros.
- Edición y visualización por secciones
    - Etapa inicial, implementación por nombre: 16 días, programador, 1440 euros.
    - Mejoras I: Número de sección: 6 días, programador, 540 euros.
    - Mejoras II: Hashes MD5: 8 días, programador, 720 euros.
    - Mejoras III: Bloqueos: 11 días, programador, 990 euros.
    - Backport a Moodle 1.8: 4 días, programador, 360 euros.
- Preparación examen de I.A: 31 días, estudiante, sin coste.
- Mejoras IV: Parser y Creole Syntax: 7 días, programador, 630 euros.

- Mejoras V: *bugfixes*: 10 días, programador, 900 euros.
- *Backport* mejoras IV y V a Moodle 1.8: 4 días, programador, 360 euros.
- Wikibook: 10 días, programador, 900 euros.
- Tags: 14 días, programador, 1470 euros.
- Grades
  - Bugfixing: 9 días, programador, 810 euros.
  - Nueva interfaz: 14 días, programador y jefe de proyecto, 1470 euros.
  - Bugfixing II: 3 días, programador, 270 euros.
  - History: 7 días, programador, 630 euros.
- Tags II: bloque y *bugfixes*: 6 días, programador, 540 euros.
- Memoria: 17 días, estudiante, sin coste.
- Presentación: 6 días, estudiante, sin coste.

## 4.2. Diagrama de Gantt

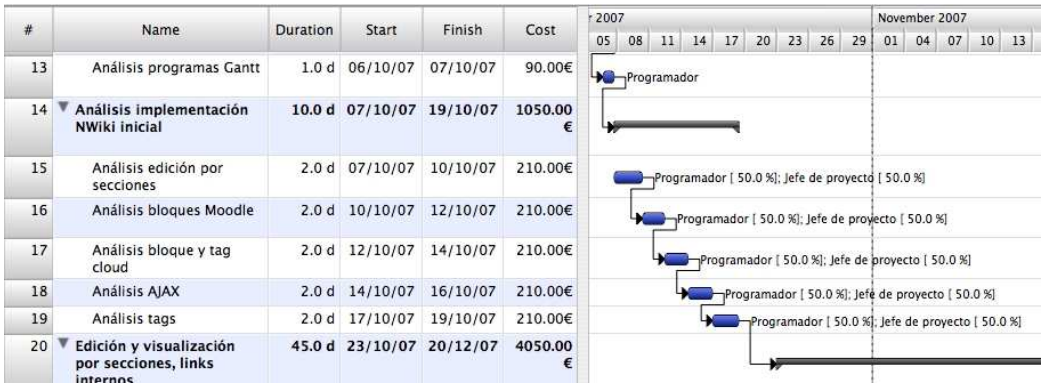
A continuación se muestra el diagrama de Gantt con los tiempos y costes de las fases anteriormente citadas.



| # | Name | Duration | Start | Finish | Cost |
|---|---|---|---|---|---|
| 1 | Aprender PHP y MySQL | 29.0 d | 01/08/07 | 10/09/07 | 0.00€ |
| 2 | **Etapa inicial** | **20.9 d** | **11/09/07** | **07/10/07** | **1200.00** |
| 3 | Crear blog | 1.0 d | 11/09/07 | 11/09/07 | 0.00€ |
| 4 | **LAMP Setup** | **7.0 d** | **14/09/07** | **24/09/07** | **300.00€** |
| 5 | Apache y MySQL | 1.0 d | 14/09/07 | 14/09/07 | 60.00€ |
| 6 | PHP | 1.0 d | 17/09/07 | 17/09/07 | 60.00€ |
| 7 | IDE: vim + plugins, konsole, ion3 | 2.0 d | 18/09/07 | 19/09/07 | 120.00€ |
| 8 | CVS Moodle y NWiki | 1.0 d | 24/09/07 | 24/09/07 | 60.00€ |
| 9 | Análisis inicial código NWiki | 4.0 d | 25/09/07 | 28/09/07 | 360.00€ |
| 10 | **Análisis Moodle** | **5.0 d** | **01/10/07** | **06/10/07** | **450.00€** |
| 11 | Historia, Arquitectura | 2.0 d | 01/10/07 | 03/10/07 | 180.00€ |
| 12 | API, Coding Style | 3.0 d | 03/10/07 | 06/10/07 | 270.00€ |
| 13 | Análisis programas Gantt | 1.0 d | 06/10/07 | 07/10/07 | 90.00€ |
| 14 | **Análisis implementación NWiki inicial** | **10.0 d** | **07/10/07** | **19/10/07** | **1050.00 €** |
| 15 | Análisis edición por | 2.0 d | 07/10/07 | 10/10/07 | 210.00€ |



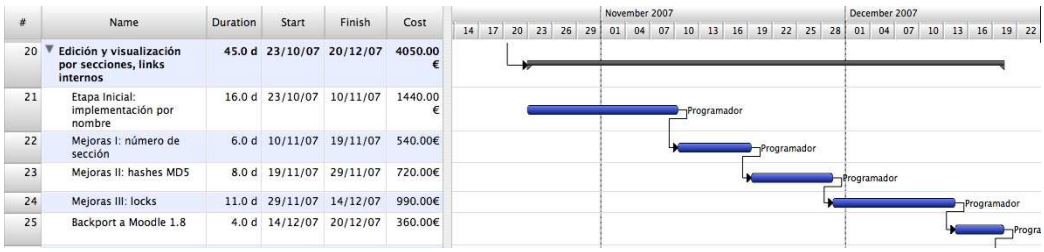
Figura 4.1: *Diagrama de Gantt, parte I*



| # | Name | Duration | Start | Finish | Cost |
|---|---|---|---|---|---|
| 13 | Análisis programas Gantt | 1.0 d | 06/10/07 | 07/10/07 | 90.00€ |
| 14 | **Análisis implementación NWiki inicial** | **10.0 d** | **07/10/07** | **19/10/07** | **1050.00 €** |
| 15 | Análisis edición por secciones | 2.0 d | 07/10/07 | 10/10/07 | 210.00€ |
| 16 | Análisis bloques Moodle | 2.0 d | 10/10/07 | 12/10/07 | 210.00€ |
| 17 | Análisis bloque y tag cloud | 2.0 d | 12/10/07 | 14/10/07 | 210.00€ |
| 18 | Análisis AJAX | 2.0 d | 14/10/07 | 16/10/07 | 210.00€ |
| 19 | Análisis tags | 2.0 d | 17/10/07 | 19/10/07 | 210.00€ |
| 20 | **Edición y visualización por secciones, links internos** | **45.0 d** | **23/10/07** | **20/12/07** | **4050.00 €** |



Figura 4.2: *Diagrama de Gantt, parte II*



| # | Name | Duration | Start | Finish | Cost |
|---|---|---|---|---|---|
| 20 | **Edición y visualización por secciones, links internos** | **45.0 d** | **23/10/07** | **20/12/07** | **4050.00 €** |
| 21 | Etapa Inicial: implementación por nombre | 16.0 d | 23/10/07 | 10/11/07 | 1440.00 € |
| 22 | Mejoras I: número de sección | 6.0 d | 10/11/07 | 19/11/07 | 540.00€ |
| 23 | Mejoras II: hashes MD5 | 8.0 d | 19/11/07 | 29/11/07 | 720.00€ |
| 24 | Mejoras III: locks | 11.0 d | 29/11/07 | 14/12/07 | 990.00€ |
| 25 | Backport a Moodle 1.8 | 4.0 d | 14/12/07 | 20/12/07 | 360.00€ |

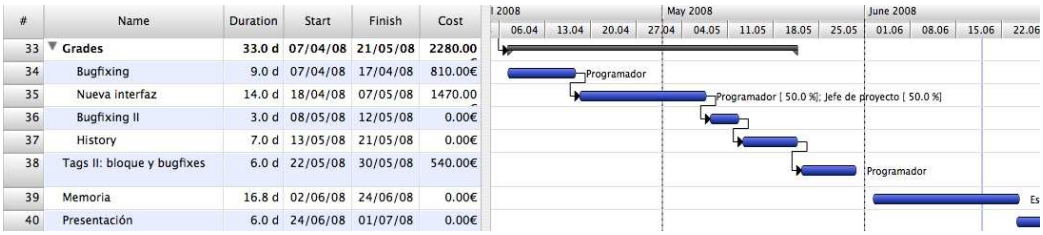


Figura 4.3: *Diagrama de Gantt, parte III*

| # | Name | Duration | Start | Finish | Cost |
|---|---|---|---|---|---|
| 26 | Clases IA | 78.0 d | 11/09/07 | 19/12/07 | 0.00€ |
| 27 | Preparación examen IA | 31.6 d | 20/12/07 | 01/02/08 | 0.00€ |
| 28 | Mejoras IV: Parser y Creole Syntax | 7.0 d | 04/02/08 | 12/02/08 | 630.00€ |
| 29 | Mejoras V: bugfixes | 10.0 d | 13/02/08 | 26/02/08 | 900.00€ |
| 30 | Backport Mejoras IV y V | 4.0 d | 27/02/08 | 03/03/08 | 360.00€ |
| 31 | Wikibook | 10.0 d | 04/03/08 | 17/03/08 | 900.00€ |
| 32 | Tags | 14.0 d | 18/03/08 | 04/04/08 | 1470.00 |
| 33 | **Grades** | **33.0 d** | **07/04/08** | **21/05/08** | **2280.00** |



Figura 4.4: *Diagrama de Gantt, parte IV*



| # | Name | Duration | Start | Finish | Cost |
|---|---|---|---|---|---|
| 33 | **Grades** | **33.0 d** | **07/04/08** | **21/05/08** | **2280.00** |
| 34 | Bugfixing | 9.0 d | 07/04/08 | 17/04/08 | 810.00€ |
| 35 | Nueva interfaz | 14.0 d | 18/04/08 | 07/05/08 | 1470.00 |
| 36 | Bugfixing II | 3.0 d | 08/05/08 | 12/05/08 | 0.00€ |
| 37 | History | 7.0 d | 13/05/08 | 21/05/08 | 0.00€ |
| 38 | Tags II: bloque y bugfixes | 6.0 d | 22/05/08 | 30/05/08 | 540.00€ |
| 39 | Memoria | 16.8 d | 02/06/08 | 24/06/08 | 0.00€ |
| 40 | Presentación | 6.0 d | 24/06/08 | 01/07/08 | 0.00€ |



Figura 4.5: *Diagrama de Gantt, parte V*

## 4.3. Análisis de costes

Los costes asociados a cada fase vienen determinados por el número de días de la fase y por quién ha desarrollado la fase.

Para éste PFC se han contado con tres roles con un sueldo por hora diferente para cada uno. Además se ha calculado una media de 6 horas al día incluyendo sábados y domingos, durante aproximadamente 173 días lo que da un total de alrededor de 1038 horas de trabajo (cuando lo recomendado son 750h para un PFC de 37.5 créditos).

Los roles son:

1. Programador: 15 euros la hora.

2. Jefe de proyecto: 20 euros la hora.

3. Administrador de sistemas: 10 euros la hora.

Así pues, el análisis de coste del proyecto en total suma unos 14130 euros.

# Capítulo 5

# Conclusiones

Alcanzada la fase final del proyecto e intentando ser los más objetivo posible se puede decir que los objetivos a cumplir en éste proyecto han sido alcanzados e incluso en algunas partes superados.

Hay que resaltar que para ser un proyecto de 37.5 créditos he invertido un tiempo en éste PFC equivalente a alrededor de 50 créditos, y eso, creo, se nota bastante en la calidad del trabajo hecho.

Aparte de las funcionalidades requeridas he tenido que *pelear* con código no programado por mi que primero he tenido que estudiar y a veces el hecho de que no funcionara perfectamente ha hecho que lo tuviera que arreglar, cosa que no hubiera pasado tanto si hubiera hecho un PFC desde cero y programado íntegramente por mi.

Por otra parte, el poder trabajar con código de otras personas me ha ayudado a mejorar como programador porque he podido conocer soluciones y métodos que de otra manera hubiera sido difícil aprender.

Además siempre que me he sentido atascado en la implementación de alguna funcionalidad he podido recurrir a Jordi Piguillem, becario en la FIB y a cargo de la gestión del proyecto NWiki y uno de los más entendidos en el proyecto Moodle en España. Para los problemas de diseño he podido contar

con la ayuda del director Marc Alier que me ha explicado las dudas que me iban surgiendo a lo largo del PFC.

La metodología usada en éste PFC con desarrollos incrementales a corto plazo me ha ayudado a mantener el proyecto al día, y el tener que comentar las mejoras que se iban haciendo en el blog me ha ayudado a componer la memoria del proyecto. El trabajar en aplicaciones web con la plataforma LAMP me ha ayudado a afianzar mis conocimientos del protocolo HTTP y del lenguaje y la API de PHP, y me servirán en un futuro a la hora de encontrar un trabajo relacionado con la programación web.

Que el proyecto sea *software libre* y ya haya muchas personas que estén utilizando funcionalidades creadas en éste proyecto es un *plus* al trabajo hecho. La inclusión de NWiki como wiki oficial de Moodle 2.0 es el punto culminante del trabajo que ha hecho el DFWikiTeam y el resto de proyectistas durante estos años.

Gracias.

# Bibliografía

[1] Hugh E. Williams y David Lane. *Web Database Applications with PHP and MySQL*. Editorial O'Reilly.

[2] Lee Babin. *Beggining Ajax with PHP. From Novice to Professional.*. Editorial Apress.

Webgrafía

[3] Manual oficial de PHP.
http://es.php.net/manual/en/index.php

[4] Referencia oficial de XHTML.
http://www.w3.org/TR/2004/WD-xhtml2-20040722/

[5] Francisco José García Peñalvo. *Estado actual de los sistemas e-learning.*
http://www.usal.es/ teoriaeducacion/rev_numero_06_2/n6_02_art_garcia_penalvo.htm

[6] Free Software Foundation.
http://www.fsf.org

[7] Proyecto GNU
http://www.gnu.org

[8] Proyecto Moodle.

- http://www.moodle.org

- docs.moodle.org

[9] Proyecto NWiki.

- http://dfwikilabs.org
- http://morfeo.upc.edu/crom/
- http://morguapo.upc.edu/crom
- http://orangoodling.blogspot.com/
- http://elpigui-i-lawiki.blogspot.com/

[10] Wikipedia:

- E-learning: http://www.wikipedia.org/wiki/E-learning
- GNU General Public License: http://www.wikipedia.org/wiki/Gpl
- PHP: http://www.wikipedia.org/wiki/PHP
- HTML: http://www.wikipedia.org/wiki/HTML
- XML: http://www.wikipedia.org/wiki/XML
- XHTML: http://www.wikipedia.org/wiki/XHTML
- AJAX: http://www.wikipedia.org/wiki/AJAX
- MySQL: http://www.wikipedia.org/wiki/MySQL
- JavaScript: http://www.wikipedia.org/wiki/JavaScript
- LaTeX: http://www.wikipedia.org/wiki/LaTeX

# Apéndice A

# Anexo: Entradas del blog

Aquí se presentan los artículos escritos en el *blog* utilizado para dar cuenta de la evalación del PFC, disponibles en http://parabol.wordpress.com.

## A.1. Task 0: Crear un blog

**Fecha: 17-09-2007**

La carrera acaba de empezar. ¿O quizás está acabando?. Éste *blog* servirá de *log*. Aquí documentaré el desarrollo del PFC de la ingeniería informática de la UPC que como todo buen samaritano me toca hacer. *Enjoy*.

*Sigue, sigue...*

*Disclaimer*: ojo, en este blog van a haber muchos ladrillos. Esto es porque cuanto más escriba ahora, menos habré de escribir al redactar la memoria.

**Intro**

Ah, se acabó el verano y llega septiembre. La city. *The Stress Strikes Back*.

Este veranito he ido documentadome (*poc a poc i bona lletra*, como dicen

en Sa Roca) para ir tomando contacto con las tecnologías en las que se basa NWiki/DFWiki, una wiki (sitio web que sirve de editor colaborativo) integrada en Moodle, que es un sistema de contenidos para dar cursos sobre Internet usado por mucha gente y que además es software libre.

**Por qué NWiki**

Inicialmente tenía pensado desarrollar un proyecto para entregar PFC *from scratch*. Tenía claro que iba a estar enfocado al desarrollo web, es un área de las ciencias de la computación (*yeah*) en la que tengo algo de experiencia laboral (5 meses con, aunque en el currículum pondre medio año, que parece más *pro* :) y en la que me apetece profundizar y aprender las tecnologías más extendidas en este mundillo, cuyas iniciales forman el acrónimo LAMP: GNU/**L**inux, **A**pache, **M**ySQL y **P**HP, y que forman junto con la tecnología AJAX (Asynchronous JavaScript and XML) parte importante de lo que ahora está *de moda* en la red de redes, la *hypeada* web 2.0.

En cuanto a la **L** ya llevo unos añitos (como pasa el tiempo) como luser, de la **A** me defiendo, con la **M** también y es la de PHP (siglas de **P**HP: **H**ypertext **P**reprocessor) el lenguaje de programación que me interesaba meter mano.

Tenía varias ideas *dospuntoceronianas* en mente, básicamente parecidas a ofrecer determinados servicios a usuarios agrupados en comunidades virtuales, inspirándome en cosas como flickr, YouTube, Google maps y calendar, menéame, last.fm y un largo etc.

Tenía un par de ideas:

- Algo centrado en las bandas musicales. Un MySpace-killer. Algo guapo técnicamente y que sirviera para promocionar a los grupillos, integrar el sistema de conciertos, un sistema automático para aquellos que les interesara liberar su música y venderla en Magnatune, colgar videos de conciertillos o videoclips, montar fácilmente conciertos con otros músicos, radio online y votos a las canciones y bandas de la semana, sistema de *lyrics* y *tabs,* y otras cosillas.

- Algo relacionado con el colegueo 2.0. Quedadas, fotos, grupos, vídeos, chat etc. No le veía mucho futuro y me parecía algo cursi, pero ahí estaba la idea.

Las dos pensaba hacerlas *from scratch*. Desarrollo desde cero y a mi bola. Sin CMSs ni leches, a pelo, como los campeones.

La segunda la descarté cuando mi hermana me dijo "*pero... si eso ya está inventado, no?*". La primera cuando intenté escanear un poco a unas bandas de colegas y no saqué mucho feeback de ello, no quería hacer un proyecto luego se olvidara en un cajón. Además, revisando memorias de otros PFCs vi cosas similares, y eso también me echó para atrás.

Mientras seguía el *brainstorming* fui leyendo la lista de proyectos que ofrecen los profesores en la web de la FIB. Me fijé en temas que me interesaban como la web, la bioinformática y el software libre en general.

Y encontré un proyecto dirigido por Marc Alier que llamó la atención por varias cosas:

- Desarrollo web con PHP: justamente lo que me interesaba aprender. Tanto Moodle como NWiki se basan en LAMP.

- Software Libre: siendo un GNU/Linuxero y seguidor de RMS (para los que no le conozcan, es como el William Wallace de la informática pero en vez de una espada de dos metros lleva la GPL). Tanto Moodle como NWiki son software libre.

- Proyecto ya en marcha: no se parte de desarrollo desde cero, sino que la tarea sería añadir funcionalidades a un código ya existente. Esto que por una parte puede ser peligroso si el código original es un infierno, en caso contrario puede servir para aprender más rápido revisando código de otra gente.

- Proyecto útil: el código actual es usado por mucha gente, por lo que las mejoras introducidas por mi PFC potencialmente también.

- Director-no-suicida: conviene, antes de aceptar el proyecto de un profesor, *investigar* si uno se está metiendo el la boca del lobo. Con las horas que hay que meter a un PFC más vale que haya una buena comunicación.

Así que aquí estamos, en la tercera casilla. La primera era documentarse un poco. La segunda, crear este blog. Ya queda menos.

## A.2. Instalar Moodle en Kubuntu (Feisty)

**Fecha: 17-09-2007**

Chuletario para tener funcionando Moodle (1.8) en una Ubuntu o derivadas:

- Instalar vía apt-get los paquetes de PHP5 y PHPMyAdmin:
  - $ sudo apt-get install php-pear php5 php5-cli php5-common php5-curl php5-gd phpmyadmin
- Los de Apache2:
  - $ sudo apt-get install apache2 apache2-mpm-prefork apache2-utils apache2.2-common libapache2-mod-php5
- Y los de MySQL 5:
  - $ sudo apt-get install libdbd-mysql-perl libmysqlclient15off mysql-client mysql-client-5.0 mysql-common mysql-server mysql-server-5.0 php5-mysql
- Editar la configuración de Apache:
- Fichero: /etc/apache2.conf:
  + # El tema de PHPMyAdmin
  + Include /etc/phpmyadmin/apache.conf
  +

+ # Evitar error en el FQDN
+ ServerName localhost

- Instalar PhpDocumenter mediante PEAR:
- $ sudo pear install phpDocumentor
- Descargar Moodle: http://docs.moodle.org/en/Installing_Moodle
  - Desde los tarballs de moodle.org
  - Desde CVS: cvs -z3 -d:pserver:anonymous@es.cvs.moodle.org:/cvsroot/moodle co -r MOODLE_18_STABLE moodle
- Copiar fuentes a /var/www o linkar
- Configurar apache:
  - Fichero: /etc/apache2.conf: + # Moodle + DirectoryIndex index.php index.html index.htm + AcceptPathInfo on
- Configurar PHP (ya están los settings necesarios)
- Configurar MySQL: $ mysql -u root -p
- nueva BBDD: moodle

  create database moodle; ALTER DATABASE moodle DEFAULT CHARACTER SET utf8 COLLATE utf8_unicode_ci;
- nuevo usuario: *moodleuser* con pass 'unpassword'
- mysql> GRANT SELECT,INSERT,UPDATE,DELETE,CREATE,CREATE TEMPORARY TABLES, DROP,INDEX,ALTER ON moodle.* TO moodleuser@localhost IDENTIFIED BY 'unpassword';

  *Nota: ojo si haceis copy&paste porque las comillas simples de wordpress son diferentes a las normales.*
- recargar MySQL:

    - $ sudo mysqladmin -u root -p reload

- Añadir directorio para datos de uploading de usuarios de moodle (se necesita en la instalación):
    - $ mkdir donde_este/moodle/uploading
    - Añadir dentro fichero .htaccess con "deny from all"
    - cambiar el propietario: $ sudo chown www-data uploading/
- Crear el config.php a partir de la ejecución de install.php desde el navegador (http://localhost/moodle/install.php) y con Apache en marcha.
- Seguir la instalación vía web. Los usuarios/contraseñas que pide son los creados en los pasos anteriores.
- Pasar el http://localhost/moodle/admin/health.php por si hay algún problema.

Act: para que Moodle muestre los errores PHP hay que configurarlo en Server > Debuggin' > Debug Messages » Developer

## A.3. RTFS + TODO-list

**Fecha: 21-09-2007**

*RTFS: Read The Fucking Source.*

Eso es lo que he estado haciendo estos días. Leer código y más código, e intentar entenderlo ya de paso. **RTFS** es la máxima en el mundillo de la programación. Si no tienes a nadie que te lo explique, entiéndelo por ti mismo (o algo así, vamos).

Quizás el problema que tiene leer código que no has escrito es que al principio cuesta un poco *ubicarse*, y a mi en particular me está costando bastante pero

creo que ya le voy cogiendo el truquillo, gracias a la ayuda de ctags, al cual le dedicaré una entrada junto a VIM y otras (varias) herramientas de desarrollo que he estado usando estos años en la universidad.

Resulta que Marc me puso deberes el lunes, una serie de features a añadir a la NWiki para que me vaya introduciendo en el meollo. "Timeline: una semana" me dijo el jefe; lo veo un poco justito para ser honestos ;)

Son éstas:

- 1. Cambiar el parser de la NWiki para que, al encontrar un link interno con la forma [[**página#sección**]] para que al clickar en él se enlace dicha página de la wiki pero donde el contenido de la página sea únicamente el de dicha sección (y sus subsecciones también).

- 2. De manera similar, añadir la posibilidad de editar por secciones, como hace por ejemplo MediaWiki, el software que hay detrás de la Wikipedia. Así, al lado del título de cada sección y subsección aparece un pequeño enlace llamado "edit" para editar justamente una sección y sus sub-secciones (esto último MediaWiki no lo hace), cosa que incrementa la usabilidad (evita que edites algo que no te interese sin querer) y la eficiencia (cargar sólamente el texto necesario).

- 3. Crear un nuevo tipo de bloque. Los bloques son esos *cuadros* funcionales que salen a derecha e izquierda de Moodle con sus respectivas funcionalidades, como pueden ser el índice de links, el calendario, información de usuario, etc etc). Éste bloque concretamente va a ser un bloque de categorías de páginas. Mostrará un listado con al lista de categorías existentes, y al clickar en una categoría se enlazará con una página que muestre las páginas de esa categoría. Además tengo que implementar un *tag clould* o *nube de tags* estilo de ésta de menéame, cuyo código es libre y está disponible y quizás me pueda servir (gracias profe). Y si puedo implementar algún tipo de *caché* mejor que mejor.

  La información de éste bloque no se guardará en BBDD sinó en una página normal de Moodle, con un formato como éste:

```
=categoria1=
  [[pagina1]]
  [[pagina2]]
  [[pagina3]]
=categoria2=
  [[pagina1]]
  [[pagina2]]
```

Lo único que no me ha quedado claro de ésto es si éstas categorías se refieren a los tags que voy a comentar en el siguiente punto o se trata de otra cosa. Definitivamente se trata de otra cosa, se refiere a las categorías como a las de la Wikipedia, como un namespace/espacio de nombres para las páginas.

- 4. Añadir un sistema de *tags* o etiquetas a las páginas de la wiki vía AJAX, al estilo del blogger de Google.

  Dichos tags se cargarían vía AJAX directamente al añadirlos en un cuadro de texto situado debajo del contenido de la página de la wiki y se guardarían en la propia página, seguramente en un formato estilo

  tags: ”tag1”, ”esto es otro tag”, ”otrootaggg”

PD: he reeditado este artículo 7 veces, lo juro. Que está pasando con el editor de Wordpress?

## A.4. El editor (visual) de Wordpress falla

**Fecha: 21-09-2007**

Hasta el gorro estoy del editor de Wordpress. Tanto del visual como el de código. En Firefox 2 va mal, y en Konqueror 3.5 peor aún. Me cambia el formato él solito, me borra partes del texto, es anti-intuitivo, si usas la opción de “guardar y seguir editando” te cambia el código... Horrible!

Mañana más, si esto me deja: acabo de ver que ha desaparecido el último artículo!

Actualización: Parece ser que no soy el único. Hay una opción en el menú Usuarios - Utilizar el menú visual para escribir, ya está desactivado. A ver qué pasa.

Actualización 2: Con el editor normal va perfecto, ya decía yo.

PD: Escribiendo este post directamente desde VIM.

## A.5. Análisis [1]: visualización por secciones

**Fecha: 25-09-2007**

Me comentó Marc la última vez que nos vimos que antes de ponerme a picar código hiciera un análisis previo y que lo colgara en el blog, por si había que discutir sobre el tema. Como todavía estoy algo verde voy a poner el del primer punto, a ver si voy encaminado o más perdido que Bush en una biblioteca.

La primera de las características que me mandaron programar, que era ésta:

- 1. Cambiar el parser de la NWiki para que, al encontrar un link interno con la forma [[página#sección]] para que al clickar en él se enlace dicha página de la wiki pero donde el contenido de la página sea únicamente el de dicha sección (y sus subsecciones también). Además, hacerlo sólo para el parser de la NWiki (porque NWiki tiene varios parsers: nwiki, dfwiki, ewiki)

Lo que he pensado ha sido lo siguiente:

- *nwikiparser.php::parse_internal_link()*: Si la página existe, pasar por la URI un nuevo parámetro que va a ser el nombre del *anchor* (es decir, "seccion" si tenemos un link estilo *link #seccion*).

- *wikistorage.class.php*: Añadir a la clase WikiStorage un nuevo atributo llamado *anchor* y cargarlo en *recover_variables()*, que es llamada desde view.php.

- *nwikiparser.php::parse_nwiki_text()*: Al principio de la función comprobar si está definida *$WS->anchor* llamar a una función estilo *wiki_filter_section()* que se encargará de filtrar el texto de la página de la wiki de la siguiente manera:

  - Recorrer el texto línea a línea eliminandolas hasta encontrar la sección con el nombre igual al *anchor*

  - Una vez encontrada la sección, calcular de qué nivel es (1, 2 o 3).

  - Recorrer líneas hasta encontrar una del mismo nivel o superior (si es inferior es una subsección y pertenecería también a la sección) hasta llegar al final. En el primer caso, borrar desde esa línea hasta el final. Mucho más sencillo hacerlo con las funciones de PHP de manipulación de strings como *strpos()* y *substr()*

También se me ha ocurrido que puede ser interesante dejar los links internos con *anchor* tal como están, es decir, que salga la página completa con scroll hacia la sección, y utilizar quizás en vez de [[pagina#seccion]] algo como [[pagina##seccion]] o [[pagina@seccion].

## A.6. Análisis [2]: Edición por secciones

**Fecha: 27-09-2007**

Análisis de la edición de una página de la wiki por secciones.

La diferencia respecto a la implementación de la visualización por secciones es que en el caso de la edición, al salvar la página lo que se necesita es actualizar sólamente la sección editada.

Los pasos que he pensado para implementarlo han sido éstos:

- *nwikiparser.php::parse_header()*: añadir un enlace similar al del tab 'Edit' a la derecha de cada cabecera de sección, hará falta quizás añadir alguna clase al CSS para que se muestre de un determinado tamaño y a la derecha. Hay que referenciar cuál es la (sub)sección a editar, he pensado en aprovechar $WS->anchor o similar a la de la visualización parcial.

- Añadir un nuevo atributo a la clase *wikistorage*, por ejemplo *$WS->dfformpartialcontent* (similar a la existente *$WS->dfformcontent*)

- En *locallib.php::wiki_edit_content()* y antes de imprimir el editor con el texto, comprobar si se trata de una edición parcial (con *$WS->anchor* o similar) y si lo es utilizar la función implementada para la visualización parcial *wiki_filter_section()*, modificando con el resultado de la función la variable *$content_text* que es la que se contiene el texto de la página y que ahora albelgará la sección correspondiente a editar.

- La función *wiki_edit_treatment()* es la que se encarga de recoger los cambios en el texto. En el caso de que sea una edición parcial, recogerá la sección actualizada (*$WS->dfformpartialcontent*) y actualizará el texto original (*$WS->dfformcontent*) con dicha sección (con una *wiki_update_section()* por ejemplo), para luego guardarse en BBDD como en el caso de las ediciones totales.

## A.7. Sobre los bloques en Moodle (Resumen)

**Fecha: 28-09-2007**

Para definir un bloque en moodle sólo es necesario definir una clase PHP en un determinado directorio del árbol de moodle. Por ejemplo *$MOODLE_HOME/blocks/wiki_categories/block_wiki_categories.php*

Notas:

- Dicha clase tiene que heredar de la clase block_base: class block_simplehtml extends block_base ...
- Es necesario sobrecargar los atributos *tittle* y *version* en el método *init()*.
- Es necesario sobrecargar el método *get_content()* y el objeto *content* con sus atributos *text* y *footer*, que es donde se añadirá el código XHTML a mostrar en el bloque.
- Bloques configurables:
  - Añadir el método

    ```
    function instance\_allow\_config() return true;
    ```

  - Con este método Moodle añade automágicamente un link "Edit..." que enlaza a un formulario XHTML que se ha de definir en un fichero con el nombre *config_instance.html* en el mismo directorio donde se encuentra la clase del bloque.
  - Además Moodle carga cada campo del formulario, al guardar éste, en una variable accesible desde *$this->nombrecampo* que permite analizar los parámetros de configuración del bloque.
- Para habilitar múltiples instancias del mismo bloque:

  ```
  function instance\_allow\_multiple() return true;
  ```

- Si queremos cambiar el aspecto de nuestro bloque podemos sobregargar una serie de métodos:
  - *hide_header()*
  - *preferred_width*
  - *html_attributes*

Existen otras posibilidades, como añadir iconos al bloque, cuestiones de seguridad para filtrar por cursos y roles, etc.

## A.8. Análisis [3]: bloque y nube de categorías

**Fecha: 30-09-2007**

En el argot wiki, una categoría es un grupo de páginas relacionadas en un mismo *namespace* o espacio de nombres. Esto significa que es una manera de agrupar páginas relacionadas bajo un mismo nombre.

En MediaWiki, el software que hay debajo de la Wikipedia, para definir que una página pertenezca a una determinada categoría se utiliza la expresión [[Category:Nombre_categoría]] al final de la página.

Como se ha introducido anteriormente, en NWiki las categorías se van a guaradar en una única página (*nwiki-categories*) en forma de lista de este estilo:

```
=categoria1=
   [[pagina1]]
   [[pagina2]]
   [[pagina3]]
=categoria2=
   [[pagina4]]
   [[pagina5]]
=categoria3=
   [[pagina6]]
```

Entonces, lo que habría de implementarse sería lo siguiente:

- *locallib.php::wiki_edit_treatment()*: analizar si existe la marca [[Category:nombre_categoria]] y en el caso de que sí exista:
    - Comprobar que exista esa categoría en el fichero de categorías.

- 1. Si existe, comprobar que la página está en dicha categoría.
  1.1. si está, no hacer nada.
  1.2. si no está, añadir la página.

- 2. Si no existe, crear la categoría y añadir la página a esa categoría.

- *nwikiparser.php:*

  - Añadir la expresión regular que encaja con las categorías en *$regex* y el nombre de función *parse_category*

  - *parse_category()*: al encontrar una tag de categorías, si se trata de una visualización se lo salta para que no se imprima. En caso de que se edite la página, se imprime.

- Crear un moodle-bloque para mostrar la lista de categorías:

  - Crear una clase *block_wiki_categories.php* en *$MOODLE_HOME/blocks/wiki_categories/*

  - Sobrecargar el método *get_content()*, en el objeto *$content* definir el atributo *$content->text* de tal manera que cargue los nombres de las categorías del fichero de categorías y cree un enlace a dicha categoría para así mostrar las páginas que pertenecen a ella.

  - Estudiar otras características a implementar en el bloque.

- Crear un fichero *wiki_tag_cloud.php* que muestre los nombres de las categorías con tamaños relativos proporcionales al número de páginas que hay por categoría. Estudiar algún tipo de caché para hacer la funcionalidad más ligera.

## A.9. Sobre AJAX

**Fecha: 01-10-2007**

AJAX, acrónimo de *Asynchronous JavaScript And XML* es una técnica de programación web que pretende hacer que una página web sea más interactiva, rápida y usable.

Básicamente se trata de ejecutar una serie de funcionalidades de **JavaScript** (principalmente el objeto *XMLHttpRequest -ActiveXObject* en IE-) del lado del cliente (navegador web) para lanzar peticiones de manera **asíncrona** en *background* al servidor (PHP en el caso de NWiki). Como respuesta a las peticiones del servidor, el navegador recibe los datos encapsulados en formato **XML** (no es obligatorio, se puede utilizar el formato que se crea conveniente), los procesa y modifica el DOM vía JavaScript para mostrar la información que interese.

Hablando en plata, es una forma de cargar datos desde el servidor sin tener que recargar la página, haciendo que la aplicación web se parezca más a una aplicación de escritorio.

Actualmente la mayoría de navegadores modernos como Mozilla Firefox, Opera, Konqueror/Safari y otros no tan modernos soportan AJAX. Aplicaciones de Google como Google Mail, Google Maps y Google Docs, YouTube, Flickr... usan AJAX para ofrecer una navegación más rápida y cómoda.

A la hora de utilizar AJAX en un proyecto, podemos optar por dos enfoques para el lado del cliente:

- Programar nosotros el código JavaScript, ya que no requiere muchas líneas de código y no es complejo.
- Utilizar una librería libre para PHP, que encapsule las funcionalidades y ofrezca alguna extensión que nos pueda ser útil. Las más usadas son:
  - Sajax: el más usado y testeado.
  - Xajax, parecido a Sajax y bastante usado también.
  - *PEAR::HTML_AJAX*: aunque forma parte de PEAR aún se encuentra fase beta.

- JPSpan.
- También existen muchos otros frameworks.

A nivel de implementación, el flujo de instrucciones en la parte del cliente es más o menos el siguiente:

- 1. Instanciar el objecto JavaScript: comprobar el navegador para ver si hay que instanciar un *XMLHttpRequest* o, en caso de detectar un IE, *ActiveXObject*.
- 2. Mandar el *request* utilizando:
  - Método *open()*: especifica la URL de la página PHP a llamar, y si la petición es por *GET* o *POST*.
  - Atributo *onreadystatechange*: indica qué función JavaScript es la que recogerá el XML de respuesta.
  - Método *send()*: envía la petición.
- Función que evalue la respuesta y modifique el **DOM** pertinentemente.

**DOM** o *Document Object Model* es la colección de objetos que crea el navegador al cargar el código (X)HTML de una página web para representar el conjunto de elementos que la componen. Así, lenguajes como JavaScript actúan sobre esos objetos, analizandolos, mofidicandolos, creando nuevos y eliminando otros... para cambiar el aspecto de la página de una manera más óptima que si se trabajara con texto/código (X)HTML directamente.

## A.10. Análisis [4]: Etiquetas / *Tags*

**Fecha: 02-10-2007**

Los *tags* o etiquetas son palabras o frases que permiten *adornar semánticamente* una página wiki. A diferencia las categorías, donde se especifica a qué jerar-

quía pertenece una página, las etiquetas permiten especificar simples conceptos relacionados con una página. Aunque pertenezcan a categorías diferentes, puede interesar relacionar conceptualmente algunas páginas si tienen algo en común, por lo que dichas páginas utilizarían la misma etiqueta.

Vía AJAX se pretende cargar todos los *tags* ya existentes en otras páginas para ofrecer un autocompletado a medida que se escriban las letras de una etiqueta ya existente, al estilo blogger.

El análisis de la implementación del sistema de etiquetas en NWiki es el siguiente:

- El formato de los *tags* será el siguiente:
  - En el editor, serán palabras o conjuntos de palabras separados por comas.
  - Para la persistencia, las etiquetas se guardaran al final (justo después de las categorías) de cada página con el siguiente formato:

    tags: "tag 1", "tag 2", "tag 3", ..., "tag n"
- En *editor/editor.php* añadir una nueva función, por ejemplo *wiki_print_edit_tags()*, que se encargue de añadir el campoextra en el editor para mostrar los *tags* actuales de la página y poder escribir nuevos.
- Modificar *editor.php::wiki_print_editor()* para incluir el fichero JavaScript con las funciones AJAX.
- Crear el fichero JavaScript con las funciones necesarias a usar por el cliente:
  - Función para inicializar el objeto *XMLHttpRequest*.
  - Función para enviar el *request* a la página PHP al escribir alguna letra en el cuadro de texto de los *tags*.
  - Función para recibir el *response* con los *tags* que empiezan por

esa/s letra/s enviados desde la página PHP.

- Función para mostrar los tags en una lista desplegable para poder ser usados por el usuario.

- Crear la función PHP encargada de recoger todos los *tags* que existen en todas las páginas en un array y los devuelva. El problema que veo con ésta aproximación es que esta función puede ser bastante costosa debido al formato en que se guardan los tags, por el hecho de que, *para cada página*, es necesario:

  - Recoger el campo *content*
  - Filtrar la línea de *tags*
  - Recoger cada *tag*

  Habrá que realizar pruebas de rendimiento para ver si hay que aplicar otra solución, como:

  - Añadir una columna *tags* a la tabla *mdl_wiki_pages* con las etiquetas separadas por comas, por ejemplo.
  - Crear una función que al guardar la página guarde las nuevas etiquetas en un fichero que servirá de caché
  - Otras que no se me ocurren ahora :)

## A.11. Entorno de trabajo (I): ion3

**Fecha: 2104-10-2007**

Cuando uno va a pasarse unos cuantos meses programando cada día más le vale tener un buen entorno de desarrollo donde poder trabajar de manera cómoda y eficiente (para eso somos informáticos, ¿no? :). Después de unos cuantos en las facultades de la UIB y la UPC y gracias a gente como David

(si no llega a ser por él no me gustaría ni programar), Ricardo y Raimón he acabado usando un conjunto de programas que mejoran muchísimo la *ardua* tarea del programador.

Si uno dispone de un sistema libre como GNU/Linux puede encontrar tranquilamente a su disposición (y gratis, señora!) decenas de herramientas enfocadas a los desarrolladores de software. Yo particularmente me quedo con ésta combinación: ion3 como *window manager*, y konsole y VIM con *plugins* para la edición de código fuente (de ellos hablaré en siguientes entradas).

**De escritorios y gestores de ventanas**

[flame]Verdad de la buena: KDE y GNOME son para nenazas, y además te obligan a usar el ratón! Sacrilegio![/flame]

Bromas aparte, uso KDE a diario (a veces con Compiz Fusion) para el tema ofimática/internet porque tiene una serie de programas buenísimos muy bien integrados en el ***K**ool **DE**sktop*:

- Konqueror: la navaja suiza de KDE. Navegador y visor universal de todo tipo de ficheros, es perfecto para administrar el sistema de ficheros como para navegar por la web (y sino que se lo digan a Apple)
- Konsole: terminal para KDE que soporta, entre otras cosas, *tabs* y *profiles*.
- YaKuake: terminal basado en Konsole que emula a la famosa consola de Quake, que se caracteriza porque *aparece* de la parte superior de la pantalla al presionar una tecla.
- Amarok: el mejor reproductor de música de la historia. Ahí queda eso.
- KMail: un cliente de correo bastante bueno y que integra cositas como SpamAssassin.
- aKregator: un agregador de noticias para seguir lo último de la blogosfera que soporta RSS y Atom.

- Digikam: un gestor de fotografías digitales muy completo.
- Basket: un programa para tomar notas que integra texto, links, imágenes y mucho más.
- K3B: el mejor programa libre para tostar CDs/DVDs y demases.

GNOME también tiene buena pinta pero no lo he usado mucho así que no comento :)

Pero a la hora de programar está clara una cosa: cuanto menos quites las manos del teclado, más rápido vas a programar. En eso se basa VIM, y también **ion3**.

### ion3

Ion3 es la evolución de ion, el primer *window manager* clasificado como *tiling window manager* y cuya primera versión se publicó en el verano del año 2000. Está diseñado para poder controlar las ventanas únicamente con el teclado, aunque también permite en cierta medida utilizar el ratón (pero quién va a perder el tiempo en eso :). Al igual que ion3, existen otros *window managers* con una filosofía parecida, como son Larswm, ratpoison (recomendación de Rai, a ver si lo pruebo de una vez), StumpWM, wmii (bastante usado, se ve), y TrsWM.

El tema es que ion3 se basa en varios conceptos, entre los cuales están los *workspaces* y los *frames*.

Los *workspaces* vienen a ser los escritorios múltiples de toda la vida (menos para los que usan Windows). Básicamente hay que saber que se crean con "Alt + F9" y que se puede cambiar al número *num* mediante "Alt + num". Por ejemplo, creo recordar que al arrancar ion3 por primera vez sólo hay un *workspace*. Pues nada, creamos unos cuantos para poder poner en ellos nuestros programillas. Para cambiar de *workspace* podemos usar "Alt + ," y "Alt + ." para ir al anterior/siguiente *workspace* respectivamente, o "Alt + 1" para ir al primero, etc.

Luego están los *frames*, que vienen a ser contentedores para las ventanas de los programas que lanzamos. Para lanzar un programa basta con apretar F3 y introducir el nombre del programa/comando a ejecutar.

Lo bueno de ion3 es que aprovecha el tamaño de las pantallas de los ordenadores al máximo. Tiene una pequeña barra inferior de estado de unos pocos píxels y en la parte superior podemos ver las barras de los *frames* que también ocupan muy poco. Además ion3 muestra todas las ventanas maximizadas por defecto, por lo que no hay una manera más óptima de visualizar los programas (a parte de la pantalla completa). Personalmente lo que me gusta es tener pocos *frames* por *workspace* y tener varios (hasta 6) de éstos últimos. Además, se pueden hacer *splits* de los frames para que ocupen media pantalla vertical u horizontal, pero es algo a lo que nunca me he acostumbrado pero que lo usa mucha gente.

Un caso práctico: supongamos que tengo creados 5 *workspaces* (con "Alt + F9" o con F9 interactivamente). Lo que yo hago es posicionarme en el primero ("Alt + 1"), lanzar un programa (F3), posicionarme en el segundo ("Alt + 2"), lanzar un programa en el segundo *workspace*, etc etc. Suelo cargar en *workspaces* consecutivos el firefox, konsole, konqueror, amarok y kopete. Así tengo los programas a golpe de "Alt + numero", y además maximizados. Si necesito ejecutar algo en una consola, ion3 la lanza con F2 automáticamente en el *worskpace* actual, con lo que aparte de la aplicación que ya teníamos tenemos otro *frame* con la consola. Para cambiar entre *frames* de un mismo *workspace* se hace con "Alt + K - num", donde "num" es la posición del frame de izquierda a derecha. Otro shortcut que se usa bastante es "Alt + K - K", que pone el foco en el anterior *frame* al actual independientemente del *workspace* en el que estemos (buenísimo).

Para los que querais profundizar un poco más (lo dudo porque el único que lee este blog soy yo) teneis éste tutorial de Pau Rullán y éste otro de Root Zero.

## A.12. Entorno de trabajo (II): IDEs vs Konsole + VIM

**Fecha: 07-10-2007**

A la hora de escribir programas de tamaño medio-grande normalmente la mayoría de gente utiliza un IDE, que son programas muy potentes que suelen llevar integrados un editor, un compilador, un *debugger*, integración con herramientas de control de versiones, herramientas para diseñar GUIs y mil cosas más. Pero existen alternativas puede que a alguno le vaya mejor.

### De IDEs

Quizás el IDE más usado sea Eclipse, ya que es *open-source*, multiplataforma (está escrito en Java), soporta muchísimos lenguajes y tiene una comunidad enorme detrás que ayuda a mejorarlo y contribuye con *plugins* a extender su funcionalidad. También hay otros *IDEs* que están muy bien, como KDevelop (ideal para programar sobre GNU/Linux y KDE), NetBeans (de Sun, para Java), JDeveloper (de Oracle, también enfocado a Java) y muchos otros.

En cuanto a funcionalidades los IDEs están muy bien y en mi opinión son una herramienta a tener en cuenta dependiendo del proyecto. Por ejemplo, a la hora de diseñar interfaces gráficas de usuario o alguna librería gráfica nos puede venir muy bien ver un *preview* de lo que estamos programando. Pero tienen un par de *peros*:

- 1. Requieren cantidades salvajes de memoria, aunque hoy en día ésto no suele ser muy problemático ya que la memoria está barata.
- 2. Están diseñados para ser usados con el ratón (o, mejor dicho, no están diseñados para ser utilizados sólo con el teclado :)
- 2.1. El editor que llevan no es VIM (bueno, ya existe un *plugin* de eclipse que lo integra, ViPlugin, pero no lo he probado aún).

### La alternativa: VIM + *plugins* sobre Konsole

Lo primero que hay que saber es lo siguiente:

**VIM es EL EDITOR. Puntoyaparte.**

O sea que es la caña, vamos. Un monumento habría que hacerle al que escribió este programa.

Quizás el único *problemilla* es que tiene una curva de aprendizaje algo bestia, por lo que al principio puede llegar a ser frustrante y tanto es así mucha gente pasa de él completamente huye de él como si fuera el demonio. Pero bueno, una vez que le coges el truquillo no puedes dejarlo de usar porque cualquier otra cosa es una pérdida de tiempo.

La potencia de VIM se debe a principalmente a dos características:

- VIM es **modal**, es decir, tiene diferentes modos: modo inserción y modo comando. La posibilidad de poner a VIM en modo comando y ejecutar acciones mediante pocas teclas es su característica principal.
- VIM es **extensible**. Mediante *maps* (vienen a ser *key-bindings*) combinados con la programación de funciones es posible aumentar las funcionalidades de VIM hasta el infinito y más allá. Éstas nuevas funcionalidades se empaquetan como *plugins* o *scripts*que es posible descargar desde la página oficial de VIM.

Si ya de por sí las características que lleva VIM de serie son superiores a cualquier IDE, si estamos acostumbrados a una série de características de éstos últimos podemos bajarnos una serie de *plugins* que las implementan y otros que las superan. Algunos de ellos son los siguientes:

- SuperTab: permite utilizar el tabulador para utilizar las funciones de autocompletado de VIM, con lista desplegable y todo. Aquí hay un enlace que explica cómo activar el autocompletado para PHP.
- LustyExplorer: muestra una ventana con una lista de los posibles ficheros a editar desde el directorio de trabajo y soporta autocompletado.
- TagList: añade un panel vertical a la izquierda con la estructura del

fichero de código fuente que estamos editando: *includes*, clases, variables/atributos, métodos/funciones... y permite acceder a ellas a golpe de *enter*.

- XHTML: permite insertar tags XHTML mediante un par de caracteres; por ejemplo, pulsando *;ah* te inserta <em><a href=""></a></em> dejando el cursor entre las comillas para poner el enlace.
- Closetag: permite vía "Control + _" insertar el cierre del último tag X/HTML abierto.
- NERDCommenter: permite comentar/descomentar líneas de código, soportando decenas de lenguajes de programación.
- MultipleSearch: vía ":Search *patrón*" permite realizar múltiples resaltados de elementos en diferentes colores.
- Otros: cecutil, vimpress, matrix, AlignPlugin, java_apidoc.

Si a alguien le interesa una pequeña presentación sobre la extensibilidad de VIM que hicen con Rai para EADAX (una asignatura de la FIB de la que se aprende algo, sorpresa!), se la puede bajar de aquí, y si alguien tiene algún plugin interesante que lo ponga en los comentarios d:D

**El resto del IDE: Konsole**

Konsole es la consola de KDE. Ofrece dos cosas interesantes:

- Interfaz con *tabs*
- *profiles*: permite guardar configuraciones de konsole, y en particular, los tabs abiertos en sus directorios correspondientes.

Para trabajar en un proyecto en particular lo suyo es dedicar *tabs* determinados a funcionalidades específicas, que son las que nos ofrecen determinados programas para completar las que pueden existir en un IDE: compilador (GCC), control de versiones (CVS/Subversion/git), debugger (gdb), *profilers* (gprof), herramientas de análisis de código (valgrind), etc.

Por ejemplo, para PHP podemos tener el primero para lanzar los servicios *apache2* y *mysql* y compilar algo si es necesario, el segundo para CVS/SVN, el tercero para documentación, y del cuarto a los siguientes podemos usarlos para editar los ficheros fuente que están en distintos directorios. Una vez que ya tenemos los *tabs* en sus directorios corresponientes, guardamos un *profile* (*preferencias* > *guardar perfil de sesiones*) que podremos recuperar lanzando el programa como *$ konsole –profile "nombredelprofile"*.

## A.13. Reunión [1] y acceso CVS a Moodle y NWiki

**Fecha: 08-10-2007**

Pozi, hoy hemos tenido la primera reunión de los proyectistas de NWiki y alrededores con Marc (a.k.a El Jefe) y con Pigui (a.k.a El Master).

Y nada, hemos ido a las 8 al despacho de Marc 3 (o 4?) proyectistas a contarle a Marc cómo llevamos los *encargos* que nos había dado hace un par de semanas. Por lo que a mi respecta, se ve que me estoy columpiando un poco: voy a dejar de hacer análisis tan específicos y dedicarme más a programar posibles soluciones, rollo *extreme programming* pero en versión *light* supongo.

Después de hablar con Marc hemos ido a hablar con Pigui, que empezó como proyectista y está como becario haciendo varios proyectos sobre Moodle y NWiki. Tiene un dominio muy alto del tema y nos va a echar una manita (o más de una me parece a mi ;).

El tema de la visualización y edición más o menos voy a programarlo como lo había pensado, y de las categorías y *tags* se me había ido un poco la pinza y haré algo más simple y intentando aprovechar la infraestructura de tags de Moodle que me indicó Marc en otra entrada. Además está previsto guardar los *tags* en una nueva tabla de la BBDD (*wiki_tags* p.ej)

Ah, también me ha dado de alta Pigui como *Developer* en el repositorio de la NWiki en sourceforge (el de La Farga lleva más de un año sin actualizarse!). Se necesita que un administrador (Marc o Pigui) den a los *developers* de alta para poder utilizar más comandos que el *checkout* del CVS de SF. Conviene usar este repositorio para trabajar con la última versión (rama *HEAD*) de NWiki, junto con la última versión de Moodle disponible, la 1.9 *beta*.

Para descargar la última versión de NWiki desde CVS:

$ export CVS_RSH=ssh
$ cvs -z3 -d:ext:*developername*@dfwiki.cvs.sourceforge.net:/cvsroot/dfwiki co -P moodle

Donde *developername* es el nombre de usuario que hemos dado de alta en sourceforge.

Para Moodle, como no nos va a dar nadie acceso como *developer* es necesario bajar el código como usuario anónimo. Como hay problemas con el servidor *oficial* podemos usar otro distinto. En este caso uso el *mirror* español y la última versión disponible (rama *HEAD*):

$ cvs -d:pserver:anonymous@es.cvs.moodle.org:/cvsroot/moodle login $ cvs -z3 -d:pserver:anonymous@es.cvs.moodle.org:/cvsroot/moodle co -r HEAD moodle

## A.14. Diagramas de Gantt

**Fecha: 09-10-2007**

Un diagrama de Gantt es un gráfico de barras que se usa mucho para la organización temporal de proyectos. Se basa en asignar tareas a recursos, y se especifica cuánto dura cada una de ellas.

Pues bien, en la memoria del PFC es obligatorio entregar la planificación del

proyecto, así que más vale ir haciéndola poco a poco a medida que se van haciendo las cosas.

Existen varios programas libres para hacer diagramas de Gantt:

- GanttProject: quizás uno de los más extendidos, está programado en Java por lo que es multiplataforma. Me parece algo lento y además la navegación por el *timeline* es poco intuitiva, pero viene bien para salir del paso.

- TaskJuggler: cuenta con una inferfaz CLI y una GUI, pero parece algo complejo si sólo se quieren hacer cosas básicas. Además para cambiar una tarea el editor gráfico muestra el fichero de texto para editarlo directamente.

- KPlato: forma parte de la *suite* KOffice de KDE. Parece algo verde, además no he podido ni cambiar la duración de una tarea de manera fácil.

- OpenWorkbench: la alternativa a Microsoft Project, está programado en Java pero de momento sólo funciona en entornos windows así que no lo he podido probar, pero tiene buena pinta. *Port* para GNU/Linux en camino.

- OpenProj: también desarrollado en Java y multiplataforma, es el programa que más me ha convencido. No tiene unas barras espectaculares pero es muy sencillo y intuitivo crear tareas y modificar su duración y fechas de inicio/fin y además cuenta con varias vistas del proyecto interesantes. Es un programa *open-source* pero no libre, pero tiene hasta paquetes debian para su descarga.

- Planner: un gestor de projectos para el entorno GNOME desarrollado con las librerías GTK+, simple pero efectivo.

Teneis una lista con otros programas en la Wikipedia.

## A.15. Enlaces (I)

**Fecha: 22-10-2007**

A lo largo de estas semanas he ido recopilando una serie de links sobre Moodle, NWiki, PHP y otras cosillas que voy a ir poniendo en el blog para no perderlos de vista.

Aquí están:

**PFC**

- Using Moodle: Foros
- Español: Foros
- Curs: Moodle en Català
- Curso: Moodle en Español

**Tags**

- Tags - MoodleDocs
- Manage tags - MoodleDocs
- Search tags - MoodleDocs
- Tag editing - MoodleDocs
- Using Moodle: Resultado
- Student projects/Social Networking features - MoodleDocs
- Using Moodle: Social Networking Features - Google Summer of Code
- [#MDL-10169] META: Social Networking Features (GSOC) - Moodle Tracker
- Tag permissions - MoodleDocs

- Moodle: Marcas
- [#MDL-6491] wiki categories... - Moodle Tracker

**DFWikiTeam**

- DFWikiTeam
- DFWikiLabs
- Crom
- Crom-II
- LaFarga.cat: dfwikiteam: Repositorio SCM
- Ludo (Marc Alier)

**NWiki**

- DFWiki: Moodle NWiki Wiki!
- LaFarga.cat
- NWiki phpDoc
- NWiki roadmap - MoodleDocs
- SourceForge.net: CVS
- wiki: api simpletests
- wiki: seguridad
- wki: manual CSS nwiki

**Dev**

- Development:Blocks - MoodleDocs
- Development:Coding - MoodleDocs

- Development:ctags - MoodleDocs
- Development:CVS for developers - MoodleDocs
- Development:Developer documentation - MoodleDocs
- Development:Developer FAQ - MoodleDocs
- Development:Output functions - MoodleDocs
- Development:Slashes - MoodleDocs
- Development:vim - MoodleDocs
- Development:Working with the Community - MoodleDocs
- Development:XMLDB defining an XML structure - MoodleDocs
- Installation FAQ - MoodleDocs
- Installing Moodle - MoodleDocs
- Interface guidelines - MoodleDocs
- Moodle architecture - MoodleDocs
- Release Notes - MoodleDocs
- Unit tests - MoodleDocs
- Upgrading - MoodleDocs
- Using Moodle: What gives with Wiki?
- Using Moodle: Wiki module
- phpDocumentor Quickstart
- Moodle Tracker

**PHP**

- Recommended PHP reading list
- Debugging techniques for PHP programmers

**LAMP**

- ApacheMySQLPHP - Community Ubuntu Documentation
- How To Set Up A Ubuntu/Debian LAMP Server

**AJAX**

- Using Ajax Agent and PHP for Auto-Complete

**XSS**

- DOM Based Cross Site Scripting or XSS of the Third Kind
- XSS, Trust, and Barney

## A.16. Reunión (II): *tags* y *grades*

**Fecha: 23-10-2007**

Han pasado dos semanas desde la primera reunión con Marc, así que hemos vuelto a quedar para ver como tenemos el panorama.

Y bueno, como el otro día hice mi primer *commit* (yeah!) me ha dicho que el tema de la visualización/edición de secciones para NWiki estaba bien, aunque quedan por *pulir* algunos aspectos que me ha comentado Pigui:

- Utilizar las funciones de traducción de strings de Moodle para el soporte multilenguaje.
- Documentar las nuevas funciones con PHPDoc.
- Concurrencia: falta gestionar como bloquear secciones de páginas cuando se editan.

Luego hablamos del tema de los *tags*, le comenté que había estado investigando como funcionaban los moodle-tags que van a entrar con Moodle 1.9 (la siguiente versión estable) implementados como parte (junto al tema de *friends* en Moodle) de las nuevas características sociales que ha implementado Luiz Eduardo Laydner Cruz como parte del Google Summer Of Code de este año en Moodle. En cuanto pula lo que me ha comentado Pigui me pongo con esto y el resto de cosillas a implementar relacionadas con los *tags*.

Por último Marc me comentó que después me tengo que poner con el tema de los *grades*, es decir, las notas de los cursos. Moodle 1.9 implementa un nuevo sistema de notas y habrá que adaptar el código existente de NWiki para que funcione con el nuevo sistema.

Ya tengo trabajo para rato.

## A.17. Reunión III: imprevistos y soluciones

**Fecha: 26-10-2007**

*No todo el oro reluce, ni toda la gente errante anda perdida. – J.R.R. Tolkien.*

La edición por secciones y los *tags* tienen una serie de *efectos colaterales* que no había tenido en cuenta y que Pigui me ha explicado esta mañana en la facultad.

En fin, en Mallorca se dice "*poc a poc i bona lletra*". Hay que hacer las cosas bien aunque lleven más tiempo, y más aún cuando NWiki va a ser la wiki oficial de Moodle 2.0.

Aquí la lista de cambios a hacer:

- Por una parte está el problema de las secciones que tienen el mismo nombre. La implementación actual se basa en los nombres de las secciones a la hora de filtrar el contenido de las páginas para las ediciones parciales; si existen dos secciones con el mismo nombre el algoritmo

escogerá la primera. Wikimedia (el software que hay debajo de la Wikipedia) asigna identificadores a las cabeceras de las secciones, habrá que echarle un vistazo aunque el código es complejo.

- Luego están parte están los relacionados con la edición concurrente de secciones. Resulta que hasta ahora si un usuario editaba una página lo que se hacía era bloquear la página entera para evitar que otro usuario cambiarla la misma página y fastidiara los cambios del primero. Al añadir la posibilidad de editar secciones de página independientemente hay tener en cuenta una serie de factores:

  - Si se edita una página hay que bloquearla. Se evita editar cualquier sección de ella.

  - Si edita una sección hay que evitar bloquear la sección y subsecciones, pero no las secciones *hermanas* (del mismo nivel) ni las de niveles superiores.

  Además es necesario haber resuelto el tema de la identificación única de secciones porque sino puede haber problemas de concurrencia con dos secciones con el mismo nombre.

- Como me recomendó Pigui, ahora tenemos dos tipos de links internos a secciones de páginas:

  - El que había anteriormente, del tipo [[página#sección]] que visualiza la página y el navegador hace *scroll* hasta la sección.

  - El link de visualizaciones parciales, del tipo [[página##sección]]. Bueno, pues la implementación de estos enlaces fastidia un poco varios wiki-bloques, como el bloque *index*, el *navigator*, el *wanted-pages*, etc, que habrá que arreglar para que no aparezcan éste tipo de enlaces.

- Hay que arreglar un par de *bugs* menores:

  - Cambiar la expresión regular que reconoce los nombres de las ca-

beceras para que acepte espacios.

- Cambiar el último parámetro de la función *optional_param()* para evitar ataques de inyección de código.

- En cuanto a los *tags*, existen una serie de *daños colaterales* derivados del hecho de añadir una nueva tabla (probablemente *mdl_wiki_tag*) a la BBDD y relacionados con los sistemas de *backup* de Moodle y NWiki: es necesario modificar estos sistemas de *backup* para guardar los datos de los *tags* (y habrá que estudiar qué datos se necesitan guardar y cuáles no), por lo que habrá que tocar diversos ficheros:

  - *db/install.xml*: generar el código XML para crear la tabla desde cero con XMLDB. Moodle tiene un programita que genera el XML a base de clicks de ratón en "Site Ad ministration > Miscellanious > XMLDB Editor"

  - *db/upgrade.php*: programar el código PHP para permitir actualizar NWiki con la nueva tabla en el caso de no querer instalarla desde cero, para que se detecten los cambios hay que modificar también el fichero *version.php*.

  - *restorelib.php*: será necesario añadir una función que restaure la tabla de *tags*.

- Por último también habrá que cambiar las funcionalidades de importación/exportación de páginas wiki propias de NWiki, que están en los ficheros *xml/importxml.db* y *xml/exportxml.php*

Divertido, ¿eh? :D

## A.18. Edición por secciones (2.0)

**Fecha: 18-11-2007**

Acabo de hacer las pruebas y parece que los nuevos cambios introducidos para la edición concurrente de secciones funcionan correctamente, a falta de implementar los bloqueos de secciones.

Los cambios son los siguientes...

En la entrada anterior decía:

*Por una parte está el problema de las secciones que tienen el mismo nombre. La implementación actual se basa en los nombres de las secciones a la hora de filtrar el contenido de las páginas para las ediciones parciales; si existen dos secciones con el mismo nombre el algoritmo escogerá la primera. Wikimedia (el software que hay debajo de la Wikipedia) asigna identificadores a las cabeceras de las secciones, habrá que echarle un vistazo aunque el código es complejo.*

Lo que se ha implementado para resolver esto es un sistema de indentificación mediante nombre y número de sección como identificación de una sección además de añadir un *hash* del contenido algunas líneas de la misma para distinguir las secciones con el mismo nombre. Además se ha resuelto el tema de que un usuario puede añadir/eliminar secciones mientras otro está editando alguna otra sección, con un algoritmo que busca cuál es la posición correcta según el contenido de la última versión de la página en la BBDD. Éste algoritmo también tiene en cuenta el caso en que otros usuarios añadan secciones con el mismo nombre de la sección que se está editando, utilizando el *hash* comentado antes.

Además se han modificado las funciones de parsing de links internos del tipo [[Página#sección]] (scroll) y tipo [[Página##sección]] (visualización parcial) para que muestren, si existen varias secciones con ese nombre, un listado de todas las secciones que hacen *matching* con el nombre 'sección'.

Como muestra la siguiente imagen, donde tenemos siete secciones con el nombre "secA" y tres con el nombre "secB":

En la imagen salen links internos a la misma página y se muestra la TOC

para comprobar que son esas sus secciones, pero pueden ser links internos a otras páginas también.

Además ahora el identificador de la clase XHTML 'header' es un entero que corresponde con la posición de la sección en la página. Esto es así porque queremos que la TOC (Table Of Contents) de la página de la wiki puede enlazar a diferentes secciones aunque tengan el mismo nombre.

Otros cambios menores:

- Ahora sólo se pueden editar secciones desde la página principal de la wiki (como en MediaWiki).
- Además al modificar una sección se vuelve a la página principal (como en MediaWiki)
- Se utilizan la función *error()* de Moodle para mostrar los errores que se puedan producir porque un usuario intente ver una sección que ha cambiado de posición porque se han añadido/quitado secciones de esa página mientras él la veía, o porque intente ver una sección y mientras él veía la página se hayan añadido secciones con el mismo nombre.

## A.19. Reunión IV y V - Backport a Moodle 1.8

**Fecha: 27-11-2007**

Los dos pasados lunes he ido a la facultad para reunirme con Marc y Pigui para hablar del estado del proyecto.

El lunes 19 fui a ver a Marc a su despacho, estaba Pigui también y me comentaron que Moodle 1.9+ y Moodle 2.0-dev han cambiado varias cosas en el núcleo de la gestión de grupos que hacen que NWiki deje de funcionar por varios sitios, por lo que como solución inmediata han pensado en sacar

ésta semana del 26/11 una nueva versión de NWiki (1.9) para que se adapte a Moodle 1.8.3+ y con los cambios para soportar la edición por secciones, que es lo que he estado haciendo en esta semana del 19 al 25: copiando código y adaptándolo a las diferencias (que haberlas haylas) entre las diferentes funcionalidades que hay entre NWiki 1.8 y NWiki 2.0. Además Ferran, uno de los autores de la DFWiki original, me comentó que había varios bugs en la generación de links para enlaces internos de la wiki, que también arreglé. Aproveché esos días también para arreglar los bloques de la wiki que fallaban y añadir el soporte a los *synonyms*, una característica de NWiki que permite que una página tenga diferentes nombres.

Ayer 26/11 fui a ver a Pigui a contarle cómo me habían ido las cosas y que me echara un cable con los *locks* o bloqueos. Hasta ahora si un usuario editaba una página se bloqueaba y listos. Ahora, como la edición de una sección es independiente de otra hay que implementar una lógica que soporte ésto, cómo puse en la otra entrada:

- Si se edita una página hay que bloquearla. Se evita editar cualquier sección de ella.
- Si se edita una sección hay que evitar bloquear la sección y subsecciones, pero no las secciones hermanas (del mismo nivel) ni las de niveles superiores.

Si tengo tiempo también he de solucionar un par de *bugs* que hay con los bloqueos para que soporten los grupos de Moodle.

## A.20. Sobre los nombres de variables externas en PHP

**Fecha: 15-12-2007**

He estado varias horas peleándome con lo que yo pensaba que era un bug en PHP por mi desconocimiento en el tratamiento de los nombres de las

variables externas (también llamadas *incoming variables*) en PHP.

Resulta que he estado indentando un código JavaScript que se veía bastante feo (aunque funciona perfectamente) y lo he formateado siguiendo el coding style de Moodle, que es el que ha de seguir NWiki.

Pues bien, resulta que tenía el código JavaScript que enviaba por AJAX (mediante las librerías YUI de Yahoo) un identificador de un *lock* de una página o sección de la wiki a un script PHP para refrescar el tiempo de dicho *lock*.

El código original era como éste y el indentado éste otro.

Parece que no hay mucha diferencia, ¿verdad?. Es más, a parte de los espacios/tabs/saltos de línea el resto de caracteres son exáctamente iguales.

Pues bien, el problema era no enteder qué hacía exactamente esa función JavaScript (no la programé yo y en teoría ese código no había que tocarlo). El último argumento, que originalmente era 'lockid=$lock->id' lo cambié a 'lockid = $lock->id', y como se pasaba a un script PHP por el método POST lo que hace PHP es, para las variables externas (como $lock) es sustituir los puntos '.' y los espacios ' ' por guiones bajos (*underscore*), como pone en la página que puse al principio. Así, el script no encontraba ningún valor dentro del campo $_POST['lockid'] y sí dentro de $_POST['lockid_'], volviéndome loco a mi y a el sistema de bloqueo de secciones que estaba programando, que no refrescaba los bloqueos correctamente.

Menos mal que Pigui me recomendó usar la extensión de Firefox Firebug para poder monitorizar entre otras cosas las peticiones HTTP que realizan las funciones PHP de NWiki.

Moraleja: si funciona, no lo toques.

## A.21. Autenticación automática en el CVS de SourceForge

**Fecha: 16-12-2007**

Cuando se trabaja con el servidor CVS de SourceForge resulta algo incómodo ir poniendo la contraseña de la cuenta cada vez que queremos ejecutar un comando.

Para solventar esto SourceForge dispone de un sistema de autenticación con cifrado de clave pública vía SSH (OpenSSH).

Ponerlo en marcha es muy sencillo si ya tenemos una cuenta en un proyecto de SourceForge:

- Generamos las claves pública/privada: $ ssh-keygen -t dsa -C "nombre-deusuarioensf@shell.sf.net"
  *Nota: desde SF se indica que si sólo vamos a usar la llave desde nuestra máquina personal o una máquina segura se puede correr el riesgo de no introducir ninguna 'passfrase' para asegurar el par de claves. La posible ventaja de esto es que al ejecutar un comando CVS no nos pida la 'passphrase', ya que si nos la pide nos encontramos con el mismo problema de introducir un password cada vez que ejecutamos un comando CVS.*

- Con el navegador accedemos a la web de SF, nos autenticamos y accedemos a nuestra cuenta (*Account*), concretamente a la sección *Host Access Information*. Allí clickamos en *Edit SSH Keys for Shell/CVS* y en el cuadro de texto del formulario pegamos el contenido de nuestra clave **pública** (normalmente está en el fichero /.ssh/id_dsa si usamos SSH2/DSA)

- Clickamos en *Update* y esperamos unos minutos para después comprobar que la ejecución de los comandos CVS no pide contraseña.

## A.22. OMFG o Vivan Los Bocatas de Lomo con Pimiento

**Fecha: 01-02-2008**

Ayer fui un día emocionante...

He aprobado mi **última asignatura** de la carrera!

Y bien justo que me ha ido, entregando prácticas al límite (el profe se ha portado, y mucho) y sufriendo mucho al final, pero lo he conseguido!

Viva yo!

## A.23. Back to bussiness

**Fecha: 01-02-2008**

Ahora que respiro más tranquilo (ver entrada anterior) sigo con el desarrollo de NWiki, donde tengo varias cosas pendientes y otras nuevas.

Por una parte el tema de arreglar bugs y cosillas de la edición por secciones (da de si el tema):

- Arreglar los últimos bugs de la edición por secciones en la rama 1.8 (work in progress).
- Comprobar y arreglar el *merge* a la rama 1.9 (HEAD).
- Arreglar el tema de que los administradores pueden sobreescribir los *locks* de la wiki tanto en 1.8 como en 1.9

Y nuevas cosillas que me ha ido diciendo Marc para implementar:

- Los *tags* de los que ya hablé anteriormente.

- RSS
- Funcionalidad para convertir una wiki con sintaxis ewiki (la wiki antigua que viene por defecto con Moodle) a sintaxis NWiki

Weeeeeeeeeeeeeeeeeeeee!

## A.24. Bugfixing days

**Fecha: 28-02-2008**

Este mes de febrero lo he dedicado casi enteramente al tema de arreglar fallos en las ramas 1.8 y 1.9 de NWiki, algunos de ellos son:

- El navegador de páginas ya no muestra los enlaces a secciones.
- Mejorado el código que detecta enlaces erróneos a secciones de páginas que no existen (tanto las secciones como páginas).
- El sistema de confirmación de bloqueos ahora funciona con al última versión de las librerías YUI (2.3) que tra Moodle.
- El bloqueo funciona al subir/eliminar archivos a la wiki.
- Los bloqueos que evitan la edición de una página o sección ahora se muestran en una tabla tipo moodle (ver imágenes).
- Modificación para permitir la sobreescritura de múltiples locks vía AJAX si es un administrador o profesor de la wiki.
- Ampliación y corrección de los errores al subir/eliminar ficheros y al guardar páginas o secciones que han sido sobreescritas por administradores o profesores.
- **Backport** de todo esto de la rama 1.9 a la rama 1.8 de NWiki

Me falta arreglar una cosilla en el parser que me ha indicado Pigui respecto a los nombres de páginas y secciones con carácteres *especiales* y me pongo ya con los *tags*.

## A.25. Yay!

**Fecha: 28-02-2008**

Dos cosas a celebrar:

- NWiki, después de una votación muy movida, **va a ser** la **wiki oficial** de Moodle 2.0 !
- Hoy hago 26 tacos. Flipa.

## A.26. Reuniones de esta semana

**Fecha: 03-04-2008**

Esta semana me he acercado 3 veces a la facultad para hablar de diferentes temas, de los que he sacado esto:

- Hay que entregar un documento un par de meses antes de la entrega del PFC que se llama informe del projecte y es un resumen de lo que se ha hecho hasta ahora en el proyecto y qué es lo que falta por hacer.
- Antes de ponerme a implementar los *tags* (el diseño esta pensado ya) tengo que arreglar un par de cosas: el tema *wikibook* (sintaxis wiki para agrupar las páginas wiki como si fueran un libro) y los *grades* para NWiki, un sistema para puntuar las notas de los alumnos donde es necesario arreglar unos cuantos bugs para la versión 1.9.
- Después de los *tags* a ver si me da tiempo a implementar el tema de

*RSS* porque también hay que hacer la temida memoria del proyecto, la cual lleva bastante tiempo escribir.

## A.27. Arreglados bugs del wikibook

**Fecha: 06-04-2008**

El jueves Pigui me comentó varios *bugs* que había en el wikibook, acabo de hacer el commit con los arreglos. El *ChangeLog* es este:

```
FIX: arreglat l'índex quan no hi cap text abans de la
     primera entrada.
FIX: arreglats strings de previous / next.
FIX: bug que quan es guardava una pàgina del wikibook
     no tornava al view de wikibook.
FIX: els fakechapters fulla ara ja no mostren cap contingut.
FIX: wikiblock index amb enllços a wikibooks.
```

## A.28. Reunión y wikigrades

**Fecha: 23-04-2008**

El pasado lunes fui a la facultad a ver a Marc i a Pigui para contarles como llevaba el tema de los *grades* (sistema para poner notas de Moodle y ahora integrado en NWiki) y paraque Marc me firmara el informe informe del proyecto, que fuí a entregar a los otros dos profesores miembros del tribunal.

Ayer por la tarde hice el *commit* de los cambios que hice para mejorar los *grades*:

- eliminats echo's sobrants.
- SQL injection a grades.lib.php.

- el combo d'usuaris de la pàgina de report dels grades ara permet elegir al primer usuari.
- es permeten elegir les escales globals a més de les del curs.
- ja es guarda correctament la configuració de la evaluació de la wiki (merci Pigui).
- el gradebox ara surt segons la configuració de la evaluació de la wiki.
- el gradebox només surt al tab 'view'
- millorat una mica el disseny del quadre de notes i del reporter.

Por otra parte ya he empezado a desarrollar los *tags*, que consistirán en:

- Una lista de tag-enlaces en la página de visualización de wikis.
- Un cuadro de tags editable en la página de edición de wikis.
- Una página especial donde se mostrarán las wikis que contienen un tag en cuestión, a la cual se llegará desde los tag-enlaces.
- Un bloque Moodle con el *tag cloud*

## A.29. Cambio de rumbo

**Fecha: 05-05-2008**

La semana pasada me la he pasado retocando el código de *report* de las notas/*grades* para las wikis de NWiki, arreglando algún algoritmo y varios bugs y mejorando la interfaz.

Pero Marc quiere que los *grades* estén impecables para la próxima *release* de NWiki así que voy a dedicarle más tiempo, para hacer cosas como:

- Arreglar el *Student Mode* para que tanto grupos como estudiantes individuales puedan editar wikis de grupo o individuales respectivamente.

- Añadir un *tab* que enlace a las notas de una wiki.

- Mejorar la página de *reporting* de las notas:

  - Hacer que las columnas sean ordenables.

  - Añadir información de usuario (foto etc).

  - Añadir enlaces donde la información sea accesible.

  - Ocultar las tablas de las ediciones dinámicamente.

- Añadir un campo feedback al poner una nota, y hacer que ese feedback se añada a la pestaña *Discussion*

- Modificar la página del historial de versiones de una página wiki:

  - Eliminar columnas *Created* y *Followup* y añadir *Quality*, añadir nota de la página al lado del nombre.

  - Añadir código JavaScript+PHP para permitir comparar diferentes versiones de una página de manera más intuitiva y de una manera parecida a como lo hace MediaWiki.

Con el trabajo que he dedicado estas 2 semanas anteriores a los *grades* y teniendo en cuenta el que le voy a dedicar, no va a ser posible añadir RSS a NWiki por mi parte, así que supongo que otro proyectista se encargará del tema.

En paralelo a esto intentaré ir escribiendo la memoria para poder entregar a mediados de junio, y después de los *grades* intentaré acabar de pulir los *tags* que los tengo casi listos.

## A.30. No news = good news

**Fecha: 02-06-2008**

Bueno, la cosa se va acabando por fin.

Hace 5 días envié lo que espero que haya sido uno de mis últimos *commits*, un bugfix del bloque de *wikitags*.

Durante este mes he implementado todo lo que puse en la entrada anterior más otras cosas para los *tags* y una nueva interfaz para seleccionar usuarios para ponerles nota. No he escrito mucho en el blog porque he estado muy liado programando, le he metido bastantes horas.

Me ha dicho Marc que esta semana me avisará si hay que hacer algún cambio *estético* y que después de eso se publicará una nueva versión de NWiki para Moodle 1.9 con los *grades* y *tags* en marcha.

Por otra parte estoy metido ya con la memoria del proyecto y la presentación.

Para la memoria estoy aprendiendo estos días algo de LaTeX y de Lyx para poder generar un texto de calidad.

En cuanto a la presentación al final la haré con un software privativo de Apple para MacOS X que se llama Keynote. Me fastidia bastante no usar software libre pero la verdad es que este programita está muy pulido, la calidad de las transparencias es muy alta y tiene unos efectos 3D y transiciones entre páginas muy buenos. La alternativa libre que pensaba usar hasta el último momento era crear las las transparencias con OpenOffice Impress, exportarlas a PDF y luego utilizar el programita KeyJnote que ofrece una visualización de las páginas de un PDF con OpenGL y con transiciones 3D. Me he decantado al final por el Keynote de Apple por estas razones:

- Tiempo: quiero acabar cuanto antes y lo mejor posible; el software de Apple me lo da todo hecho.
- El modo presentación que tiene es perfecto: en la pantalla externa/proyector

sale la presentación y en el portátil sale la presentación, las notas de cada transparencia y un cronómetro para controlar el tiempo de exposición.

- Utilizar el mando a distancia de mi MacBook Pro + Keynote sin tener que configurar nada.

Si tengo que hacer otra presentación con algo más de tiempo intentaré currarmelo con software libre, lo cual no quita que la ingeniería que hay detrás de los programas de Apple sea impresionante.