PFC- Aplicaciones Web para trabajo colaborativo

Estudio de la plataforma *Moodle*: Implantación de metodologías de trabajo en grupo

Memoria

Versión 1.1

Última actualización: 01/06/2006

Consultor: Fatos Xhafa

Alumno: Asier Cuevas Gómez

asiercu@uoc.edu

Agradecimientos

Antes de introducirnos en el mundo de las aplicaciones Web para trabajo colaborativo y las plataformas de aprendizaje en línea quisiera dedicar estas primeras líneas a todas aquellas personas que me han ayudado y apoyado durante su realización.

A mi pareja, familiares, amigos y compañeros, por su apoyo y el ánimo recibido y, dicho sea de paso, por soportar estoicamente mis intentos más o menos afortunados de explicarles en que consistía este proyecto.

Y finalmente a mi consultor, Fatos Xhafa, por su ayuda a la hora de introducirme en el mundo del trabajo colaborativo y sus aplicaciones.

Lista de figuras

Nota: El nombre de las figuras indica el capítulo, la sección y orden de aparición.

- **Figura 2.3-1.** Diagrama de Gantt.
- **Figura 3.2.1-1.** Ejemplo de tarea en línea.
- **Figura 3.2.1-2.** Ejemplo de consulta rápida.
- **Figura 3.2.1-3.** Ejemplo de mensaje de un foro.
- **Figura 3.2.1-4.** Ejemplo de lección.
- **Figura 3.2.1-5.** Ejemplo de encuesta.
- **Figura 3.2.1-6.** Ejemplo de cuestionario.
- **Figura 3.2.1-7.** Ejemplo del *Hot Potatoe JQuiz*.
- **Figura 3.2.1-8.** Ejemplo del *Hot Potatoe JCloze*.
- **Figura 3.2.1-9.** Ejemplo del *Hot Potatoe JCross*.
- **Figura 3.2.1-10.** Ejemplo del *Hot Potatoe Sequitur*.
- **Figura 3.2.1-11.** Ejemplo del *Hot Potatoe RhuBard*.
- **Figura 3.2.1-12.** Ejemplo del SCORM.
- **Figura 3.2.1-13.** Ejemplo de Wiki.
- **Figura 3.2.2-1.** Ejemplo de visualización de un directorio.
- **Figura 3.2.3-1.** Visualización del bloque de actividades.
- **Figura 3.3.3-1.** Ejemplo de definición de clase con método init()
- **Figura 3.3.3-2.** Ejemplo de método get_content()
- **Figura 3.3.3-3.** Configuración del método instance_allow_config()
- **Figura 3.3.3-4.** Ejemplo de configuración a través de un editor WYSIWYG.
- **Figura 3.3.3-5.** Modificación del método get_content() para mostrar la configuración del bloque.
- **Figura 3.3.3-6.** Configuración del método instance_allow_multiple()
- **Figura 3.3.3-7.** Configuración del método has_config()
- **Figura 3.3.3-8.** Ejemplo de configuración global.
- **Figura 3.3.3-9.** Ejemplo de configuración del método config_save()
- **Figura 3.3.3-10.** Configuración del método hide_header()
- **Figura 3.3.3-11.** Ejemplo de configuración del método prederred_width()
- **Figura 3.3.3-12.** Ejemplo de configuración del método html_attibutes()
- **Figura 3.3.3-13.** Ejemplo de configuración del método applicable_formats()
- **Figura 3.3.3-14.** Ejemplo de configuración del método applicable_formats()
- **Figura 3.3.3-15.** Ejemplo de configuración del método applicable_formats()
- **Figura 3.3.3-16.** Ejemplo de definición de clase lista.
- **Figura 3.3.3-17.** Ejemplo de configuración del método get_content() para un bloque tipo.
- **Figura 4.1-1.** Acceso de un usuario al sistema.
- **Figura 4.1.2-1.** Vista previa del tema formal_white.
- **Figura 4.1.3-1.** Modelo entidad-relación (DER).
- **Figura 4.1.4-1.** Recuperar datos enviados desde un formulario.
- **Figura 4.1.4-2.** Contenido del constructor structuredforum_workspace().
- **Figura 4.1.4-3.** Ejemplo de descripción de la clase structuredforum_workspace desde la API.
- **Figura 4.2-1.** Arquitectura del foro estructurado.
- **Figura 4.2.3-1.** Gráfica de utilización del foro a través de la librería *PHPLot*.

- **Figura 4.2.4-1.** Estructura de directorios.
- **Figura 4.2.4-2.** Estructura de directorios de datos.

Índice general

1. Introducción 6

1.1. Marco y propósito 6
1.2. Objetivos 7
1.3. Alcance del proyecto 8
1.4. Resultados esperados 8

2. Organización del proyecto 9

2.1. Relación de actividades 9
2.2. Estimación de la duración de las tareas 12
2.3 Planificación temporal 12
2.4. Hitos de control 12
2.5. Otras consideraciones 12

3. Estudio de la plataforma Moodle 14

3.1. Definición y funcionalidades 14
3.1.1. Definición 14
3.1.2. Características principales 15

3.2. Arquitectura 19
3.2.1. Módulos de actividades 19
3.2.2. Recursos 27
3.2.3. Bloques 28
3.2.4. Tecnología 31
3.2.5. API principal 32

3.3. Especificación de componentes 34
3.3.1. Normas y convenios 34
3.3.2. Especificación de módulos de actividades 38
3.3.3. Especificación de bloques 39
3.3.4. Especificación de temas 44
3.3.5. Especificación de esquemas de base de datos 44

4. Implementación del foro estructurado 45

4.1. Revisión de la especificación y el diseño 45
4.1.1. Nomeclatura 46
4.1.2. Interfaz 47
4.3.3. Modelo de base de datos 48
4.3.4. Clases y objetos 53

4.2. Implementación 57
4.2.1. Capa de persistencia 57
4.2.2. Capa lógica 60
4.2.3. Capa de presentación 62
4.2.4. Estructura de directorio 63

5. Conclusiones 66

6. Glosario 68

Bibliografía 70

1. Introducción

1.1. Marco y propósito

En los estudios de la *Universitat Oberta de Catalunya* (UOC) se imparten asignaturas que se basan en el trabajo en grupo a través de aplicaciones Web (gestión de los espacios de trabajo, gestión de usuarios, planificación, versionado de aplicativos y documentación, etc.). De entre todas estas aplicaciones destacan los foros de debate a través de los cuales los participantes tienen la posibilidad de colaborar, razonar y proponer posibles soluciones sobre diversos temas.

Un foro estructurado es algo más que un foro de debate, es un proceso de discusión basado en tres tipos de contribuciones o etapas:

- Especificación. El tutor o grupo coordinador inicia el proceso de debate definiendo la actividad y objetivos del grupo.

- Elaboración. Los participantes (en nuestro caso estudiantes) participan proponiendo posibles soluciones a la actividad propuesta.

- Consenso. Cuando los participantes alcanzan en común una propuesta de solución correcta, el tutor o grupo coordinador cierra el debate. Acto seguido se extraen las conclusiones pertinentes sobre la actividad, solución propuesta y participación alcanzada.

El funcionamiento de este tipo de foros está descrito en un estudio previo que detalla la especificación y el diseño del mismo, mostrando además un prototipo que implementa parcialmente algunas de sus funcionalidades.

Recientemente se está utilizando en gran medida una plataforma de libre distribución llamada *Moodle* que cubre la necesidad de disponer de una entorno de aprendizaje virtual abierto, gratuito y basado en “pedagogía constructivista social” (concentrarse en las experiencias que podrían ser mejores para aprender desde el punto de vista de los estudiantes, en lugar de limitarse simplemente a proporcionarles la información que creen que necesitan saber).

Por lo que se desprende del estudio previo del foro estructurado parece que este tipo de aplicación Web encaja dentro de la filosofía Moodle gracias al conocimiento que se genera durante el proceso de discusión que permite el autoaprendizaje de los estudiantes. Eso sí, si queremos añadir esta funcionalidad a *Moodle* deberemos revisar la especificación de la aplicación Web ya que ésta está fuertemente ligada a una plataforma genérica.

En concreto, con este proyecto se pretende:

- Estudiar las posibilidades que ofrece *Moodle* y *Moodle*Cat como soporte de trabajo colaborativo.

- Implementar un foro estructurado sobre plataforma *Moodle* basado en tres tipos de contribuciones o etapas: especificación, elaboración y consenso.

1.2. Objetivos

Se pretende realizar un estudio exhaustivo de la arquitectura y funcionalidades principales que ofrece *Moodle* como entorno de aprendizaje orientado al trabajo en grupo y desarrollo de aplicaciones Web a medida, dando a conocer los principios pedagógicos constructivistas que promueve en un entorno de trabajo colaborativo. A su vez, se pretende implementar un foro de debate estructurado basado en tres tipos de contribuciones o etapas:

- Especificación. El tutor o grupo coordinador inicia el proceso de debate definiendo la actividad y objetivos del grupo.

- Elaboración. Los participantes (en nuestro caso estudiantes) participan proponiendo posibles soluciones a la actividad propuesta.

- Consenso. Cuando los participantes alcanzan en común una propuesta de solución correcta, el tutor o grupo coordinador cierra el debate. Acto seguido se extraen las conclusiones pertinentes sobre la actividad, solución propuesta y participación alcanzada.

Por lo tanto, el objetivo es múltiple:

- Dar a conocer las posibilidades que ofrece *Moodle* y para el trabajo colaborativo en grupo.

- Dar a conocer las posibilidades que ofrece *Moodle* en el desarrollo de aplicaciones Web a medida.

- Dar a conocer la comunidad *MoodleCat* como soporte a la introducción del entorno de aprendizaje virtual *Moodle* en las escuelas catalanas.

- Dar a conocer las ventajas y limitaciones que aporta un foro de debate estructurado como herramienta orientada a favorecer el aprendizaje individual y en grupo, permitiendo a la vez realizar un seguimiento del progreso y evaluación del mismo.

1.3. Alcance del proyecto

El resultado final de este PFC es un producto informático diseñado para su aplicación dentro del contexto de la *Universitat Oberta de Catalunya*. La aplicación de esta nueva herramienta en el ámbito de la UOC tendrá una utilidad práctica. Por un lado permitirá a los estudiantes colaborar, razonar y proponer posibles soluciones sobre diversos temas en un entorno colaborativo en grupo. Por otro lado, aportará a los profesores y coordinadores de grupo de un sistema de seguimiento de la actividad en grupo.

El producto desarrollado en este proyecto no podrá ser incorporado directamente al campus virtual de la UOC; será necesario disponer de un entorno de aprendizaje específico cuyo estudio es otro de los objetivos de este PFC.

1.4. Resultados esperados

Con este proyecto se espera obtener un conocimiento amplio de la plataforma *Moodle*. En concreto, se espera obtener:

- La elaboración de un documento que refleje el estudio de la plataforma *Moodle* como entorno de aprendizaje orientado al trabajo en grupo y desarrollo de aplicaciones Web a medida. Este documento deberá reflejar la arquitectura y funcionalidades de la plataforma, entorno de producción necesario para su puesta en funcionamiento y referencia al proceso de instalación y utilización basado en la documentación oficial.

- La implementación de un foro de debate estructurado basado en los tres tipos de contribuciones o etapas comentados.

- Integrar el foro estructurado sobre la plataforma *Moodle* a través de un módulo independiente.

- Generar un proceso y guía de instalación del foro estructurado.

- Generar un manual de usuario referente al funcionamiento del foro estructurado.

2. Organización del proyecto

2.1. Relación de actividades

A continuación se muestra la relación de actividades que han sido necesarias para llevar a cabo este proyecto.

T1. Definición del proyecto

El proyecto se define en el presente documento el cual hace referencia al plan general del proyecto.

T2. Especificación de requerimientos

Lectura y recogida de información inicial

Consiste en recapitular toda la información inicial, requerimientos funcionales y aspectos relacionados con el entorno de desarrollo y producción (lenguaje de programación, base de datos, servidores de aplicaciones, herramientas de desarrollo,…). En concreto, podemos distinguir dos bloques:

1. Para el estudio de la plataforma *Moodle* será necesario recapitular toda la información disponible referente a la arquitectura y funcionalidades de la misma, así como de la especificación que deben cumplir las aplicaciones o módulos integrados en este sistema.

2. Para la implementación del foro estructurado partimos de un trabajo previo sobre la especificación y diseño del mismo, es posible que será necesario adaptarlo con el fin de integrarlo en el sistema.

   Referencia: Caballé S. and Xhafa F. (2005). *A Web-Based Structured Discussion Forum for Collaborative Learning Environments*. Master's Final Project (PFC). Open University of Catalonia (UOC).

Instalación de la plataforma

Se llevará a cabo la instalación de la plataforma *Moodle* en un entorno real como soporte previo al estudio de la misma e implementación del foro. Para ello se seguirá la documentación descrita en la Web oficial: www.*Moodle*.org.

Glosario de términos

Se elaborará un glosario general sobre los términos específicos del proyecto.

Documentación

Se elaborará un documento respecto este apartado.

T3. Estudio de la plataforma *Moodle*

El estudio de la plataforma *Moodle* servirá como análisis previo para la implementación del foro de debate estructurado del cual disponemos de su especificación y diseño.

Definición y características

Se llevará a cabo un primer estudio sobre la definición y características generales de *Moodle* como plataforma de soporte al trabajo en grupo y al desarrollo de aplicaciones a medida para trabajo colaborativo.

Arquitectura

Consistirá en realizar un estudio detallado de la arquitectura del sistema detallado la tecnología empleada, componentes, estructura de la base de datos, seguridad e interfaz.

Especificación de componentes

Consistirá en estudiar la especificación que deberán cumplir los componentes referentes al sistema y a los módulos independientes. A su vez, deberá especificarse las nomenclaturas y convenios establecidos en el entorno de desarrollo.

Documentación

Se elaborará un documento referente al estudio de la plataforma que incorporará la documentación previa elaborada en la especificación de requerimientos.

T4. Revisión de la especificación y diseño del foro

Revisión de la especificación

Una vez realizado el estudio, será necesario revisar la especificación del foro estructurado para poder adaptarlo a la plataforma. Como ya se ha comentado, partimos de un trabajo previo.

Revisión de los subsistemas

Será necesario realizar una revisión de todos los subsistemas que componen el mismo tras la revisión de la especificación antes comentada.

Revisión de la base de datos

Deberemos revisar la estructura de la base de datos del foro, así como la nomenclatura utilizada en los nombres de las tablas y campos, para que ésta cumpla las especificaciones de la plataforma.

Diseño de la interficie de usuario

Aunque en el estudio previo se propone una interficie sencilla de usuario en HTML para el foro, ésta deberá ser adaptada para su integración en la plataforma.

Documentación

Se generará la documentación necesaria obtenida en este apartado.

T5. Implementación del foro

Instalación del entorno de desarrollo

Se llevará a cabo la instalación del entorno de trabajo necesario para llevar a cabo el desarrollo y pruebas necesarias. La plataforma ya habrá sido instalada en la etapa del estudio, aquí se trata se instalar las herramientas de desarrollo necesarias (editores de código, editores visuales, entorno de trabajo...).

Codificación de los subsistemas

Se llevará a cabo la codificación del foro estructurado según los subsistemas obtenidos en la fase de revisión del diseño.

Pruebas

Se deberán realizar pruebas de cada subsistema de manera individual y posteriormente de todo el aplicativo integrado en el sistema.

Documentación

Se generará la documentación necesaria obtenida en este apartado.

T6. Puesta en funcionamiento

Se elaborará un proceso y guía de instalación del foro estructurado implementado sobre un sistema real. Deberá venir acompañado de un manual de usuario sobre la utilización del mismo.

T7. Elaboración de la memoria final

La memoria final estará compuesta de los diferentes informes que se habrán ido elaborando durante el proyecto con las conclusiones y consideraciones finales.

T8. Elaboración de la presentación del proyecto

En último lugar, se deberá elaborar una presentación breve del proyecto que ayude junto a la memoria a la evaluación del mismo.

2.2. Estimación de la duración de las tareas

La estimación de la duración de las tareas se ha realizado en función del esfuerzo requerido, horas y días disponibles y fechas fijadas en el calendario del PFC. Se propone un calendario laboral de 5 horas al día de lunes a viernes de 16:00 a 21:00 horas con los siguientes días festivos (según el calendario laboral oficial de la zona de residencia del autor - Baleares): 13/04/2006 (jueves santo), 14/04/2006 (viernes santo) y 01/05/2006 (día del trabajador).

En el apartado siguiente se muestra en detalle esta información.

2.3. Planificación temporal

En el siguiente *diagrama de Gantt* se muestra la estimación en horas de las tareas, la planificación temporal y los hitos de control que se establecen.

	Nombre de tarea	Duración	Comienzo	Fin
1	Inicio del proyecto	0 días	vie 03/03/06	vie 03/03/0
2	**T1 Plan general del proyecto**	20 horas	vie 03/03/06	mié 08/03/0
3		0 días	mié 08/03/06	mié 08/03/0
4	⊟ **T2 Especificación de requerimientos**	**3,6 días**	**jue 09/03/06**	**mar 14/03/0**
5	Lectura y recogida de información inicial	10 horas	jue 09/03/06	vie 10/03/0
6	Instalación de la plataforma	5 horas	lun 13/03/06	lun 13/03/0
7	Glosario de términos	1 hora	mar 14/03/06	mar 14/03/0
8	Documentación	2 horas	mar 14/03/06	mar 14/03/0
9	⊟ **T3 Estudio de la plataforma**	**9 días**	**mar 14/03/06**	**lun 27/03/0**
10	Definición y características	5 horas	mar 14/03/06	mié 15/03/0
11	Arquitectura	8 horas	mié 15/03/06	vie 17/03/0
12	Especificación de componentes	10 horas	vie 17/03/06	mar 21/03/0
13	Documentación	22 horas	mar 21/03/06	lun 27/03/0
14		0 días	lun 27/03/06	lun 27/03/0
15	⊟ **T4 Revisión de la especificación y diseño del foro**	**7,6 días**	**lun 27/03/06**	**jue 06/04/0**
16	Revisión de la especificación	6 horas	lun 27/03/06	mar 28/03/0
17	Revisión de los subsistemas	10 horas	mar 28/03/06	jue 30/03/0
18	Revisión de la base de datos	2 horas	jue 30/03/06	vie 31/03/0
19	Diseño de la interficie de usuario	12 horas	vie 31/03/06	mar 04/04/0
20	Documentación	8 horas	mar 04/04/06	jue 06/04/0
21		0 días	jue 06/04/06	jue 06/04/0
22	⊟ **T5 Implementación del foro**	**23,2 días**	**jue 06/04/06**	**vie 12/05/0**
23	Instalación del entorno de desarrollo	5 horas	jue 06/04/06	vie 07/04/0
24	Codificación de los subsistemas	100 horas	vie 07/04/06	mié 10/05/0
25	Pruebas	6 horas	mié 10/05/06	jue 11/05/0
26	Documentación	5 horas	jue 11/05/06	vie 12/05/0
27	**T6 Puesta en funcionamiento**	6 horas	vie 12/05/06	lun 15/05/0
28		0 horas	lun 15/05/06	lun 15/05/0
29	**T7 Elaboración de la memoria final**	50 horas	lun 15/05/06	lun 29/05/0
30	**T8 Elaboración de la presentación del proyecto**	20 horas	lun 29/05/06	vie 02/06/0
31	**Entrega y finalización**	0 horas	vie 02/06/06	vie 02/06/0

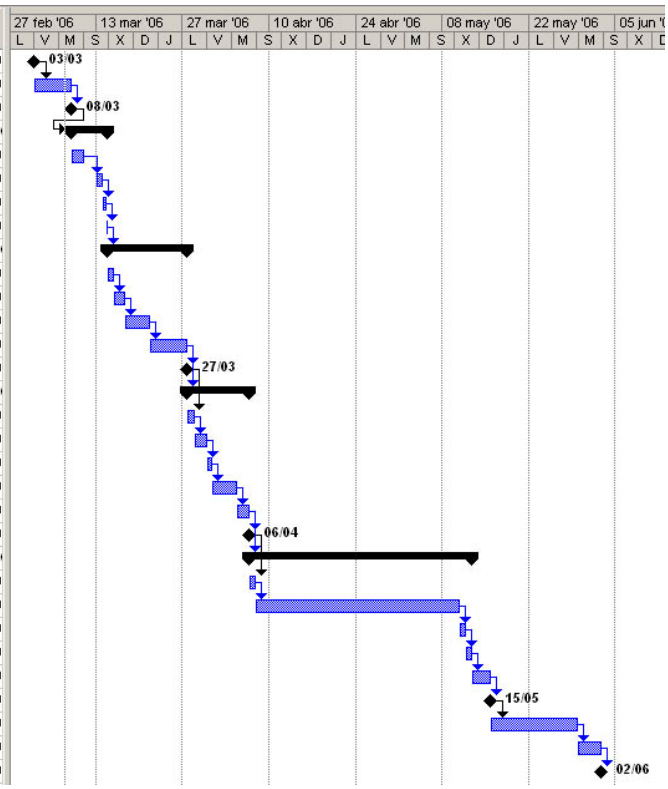


Figura 2.3-1. Diagrama de Gantt.

2.4. Hitos de control

- 8 de marzo. Plan general del proyecto.
- 27 de marzo. Estudio de la arquitectura y funcionalidades de la plataforma *Moodle*.
- 6 de abril. Revisión de la especificación y diseño del foro.
- 15 de mayo. Implementación y puesta en funcionamiento.
- 2 de junio. Entrega final.

Título	Entrega
PAC1: Planificación y definición de la propuesta	9 de marzo
PAC2: Estudio de la plataforma	28 de marzo
PAC3: Desarrollo	16 de mayo
Entrega final	2 de junio

2.5. Otras consideraciones

El coste del proyecto se presupone nulo debido a que en todo momento se prevé utilizar de forma adecuada software gratuito y de libre distribución; tanto para el entorno de producción (base de datos, tecnología y servidor de aplicaciones) como para el entorno de desarrollo (editores de código, visuales, entorno de trabajo...). Las horas dedicadas a la elaboración y seguimiento del mismo forman parte del calendario del PFC asignado.

Queda fuera del alcance del proyecto el mantenimiento y posibles mejoras que puedan realizarse tanto del estudio de la plataforma como de la implementación del foro estructurado.

3. Estudio de la plataforma Moodle

3.1. Definición y funcionalidades

3.1.1. Definición

Moodle fue iniciado por *Martin Dougiamas* a finales de los años noventa quien trabajó como administrador de *WebCT* en la la *Curtin University of Technology*. El proyecto surge como un reto personal del autor debido a la frustración por los productos software comerciales existentes dedicados al aprendizaje virtual. Actualmente el autor continúa dirigiendo el proyecto aunque en él participa un gran número de colaboradores repartidos por todo el mundo que han evolucionando el producto de tal manera que existen más de 2.600 sitios Web en más de 100 países y está traducido a más de 50 idiomas.

Moodle es el acrónimo de *Modular Object-Oriented Dynamic Learning Environment* (Entorno/plataforma de Aprendizaje Dinámico Orientado a Objetos y Modular). Es un paquete software de libre distribución escrito en PHP destinado a la creación de sitios Web de aprendizaje en entornos de trabajo colaborativos (también conocidos como *Learning Content Management System* - LCMS). Puede utilizarse en cualquier comunidad de usuarios pero sobretodo está dirigido a escuelas, institutos, universidades, centros de formación profesional, negocios, academias, hospitales, librerías y agencias de empleo.

La principal característica que diferencia *Moodle* de otros sistemas e-learning (como *Dokeos, ATutor*, *Claroline*, *ILIAS*, *Webct*, *Blackboard*,..) son los **principios pedagógicos constructivistas** en los cuales está basado. Estos principios promueven que el alumno se implique activamente en su propio aprendizaje, analizando, investigando, colaborando, compartiendo, construyendo y generando en razón a lo que ya sabe; es decir, los alumnos aprenden de ellos mismos promoviendo la construcción de nuevas ideas, mientras que el profesor pasa de ser la fuente de conocimiento principal a ser un simple árbitro dentro del proceso de aprendizaje. Estos principios marcan profundamente la organización de los cursos ofertados a través de tres modelos de iteración en línea:

- Semanal. El curso gira alrededor de la asignación de actividades semanales.
- Temático. El curso se organiza en base a los temas propuestos.
- Social. El curso está basado en debates a través de foros de discusión. Es un modelo muy orientado al trabajo colaborativo basado en la interacción entre los participantes.

MoodleCat es un servicio tecnológico dedicado a la implantación de *Moodle* en las escuelas que se encuentran dentro del territorio de habla catalana. Su objetivo es poner a disposición de éstas la infraestructura, soporte y asesoramiento tecnológico necesario para llevar a cabo esta tarea. Ofrece cursos de formación destinados a profesores para que éstos sean capaces de dinamizar a los estudiantes. También ofrece un servicio de soporte para la creación de cursos personalizados de calidad.

3.1.2. Características principales

A grandes rasgos, *Moodle* se caracteriza por una serie de aspectos relacionados con el diseño, la administración del sitio, los usuarios y el formato de los cursos:

Diseño:

- Promueve una pedagogía constructivista social que trata de concentrarse en las experiencias que podrían ser mejores para aprender desde el punto de vista de los estudiantes, en lugar de limitarse simplemente a proporcionarles la información que cree que necesitan saber.

- Es adecuada sobretodo para cursos en línea aunque también puede ser útil como refuerzo para el aprendizaje presencial.

- El usuario (profesor/estudiante/administrador) sólo necesita un navegador estándar para acceder a ella, por lo tanto puede ser accedido desde cualquier entorno o sistema operativo.

- Ha sido creada mediante una abstracción completa de base de datos. Esto permite que pueda utilizarse cualquier sistema gestor de base de datos (SGBD), aunque en la actualidad sólo hay desarrollados esquemas para *MySQL* y *PostgreSQL*.

- Es relativamente sencilla de instalar y actualizar.

- Muestra una lista descriptiva de todos los cursos ofertados.

- Los cursos se organizan por categorías facilitando así su búsqueda.

Administración del sitio:

- Es un paquete software escrito en PHP de libre distribución, totalmente abierto y gratuito. Utiliza licencia GNU que implica la posibilidad de copiar, utilizar y modificar *Moodle* siempre y cuando se acepte proporcionar el código fuente a otros, no modificar o eliminar la licencia original y los derechos de autor y aplicar esta misma licencia a cualquier trabajo derivado de él.

- Existe un único usuario administrador encargado de administrar todo el sitio. Este usuario se define durante el proceso de instalación.

- Esta diseñado de manera modular, permitiendo agregar en cualquier momento contenidos que motiven al estudiante. Es posible añadir módulos no estándar no pertenecientes a la distribución oficial.

- Puede ejecutarse en una amplia variedad de plataformas.

- Puede utilizarse junto a otros sistemas.

- Es multi-idioma. Actualmente soporta más de 40 idiomas diferentes, entre ellos el catalán y el castellano.

- Es un sistema totalmente personalizable a través de temas.

- Existen módulos no estándar que pueden ser añadidos al sistema.

- Dispone de una API extensa y un manual de desarrolladores con el fin de ampliar o modificar las funcionalidades.

Administración de usuarios:

- Los objetivos son reducir al mínimo el trabajo del administrador Éste sólo ha de controlar la creación de cursos y determinar los profesores asignados a los cursos.

- Los usuarios se dan de alta a través de un formulario en el cual han de indicar una dirección de correo electrónico válida.

- Los profesores tienen la posibilidad de dar de alta o baja manualmente a los estudiantes.

- Soporta diferentes tipos de autentificación: manual, LDAP, IMAP, POP3, NNTP y a través de una base de datos externa.

- Permite conectarse de forma segura utilizando certificados SSL y TLS.
- El tipo de acceso (invitado, administrador, profesor o estudiante) determinará los privilegios del usuario.

- Una persona necesita una única cuenta para acceder al sistema, aunque puede tener diferentes tipos de acceso.

- Un usuario tiene la posibilidad de personalizar la zona horaria, idioma de la interfaz y perfil personal (indicando una descripción de si mismo y/o adjuntando una foto). También existe la posibilidad de que un profesor limite en un curso concreto el uso de un idioma.

Administración de cursos:

- Existe la posibilidad de restringir el acceso a diversos cursos y determinados usuarios a través de una clave de acceso.

- Existe la posibilidad de permitir el acceso anónimo a diversos cursos.

- Para un profesor existen diferentes niveles de acceso a las opciones de un curso. Sólo el profesor principal del mismo tendrá control completo sobre todos los elementos, incluyendo posibles restricciones a otros profesores.

- Un curso se puede ofertar en diferentes formatos: semanal, temático o social basado en debates.

- Los cursos se componen de diferentes actividades dependiendo de las opciones que haya escogido el profesor. En concreto, un curso suele componerse de foros de debate, glosarios, cuestionarios, recursos, consultas, encuestas, tareas, chats y talleres.

- Permite un registro y seguimiento completo de la actividad de cada estudiante.

- Las calificaciones de todas las actividades de un curso se engloban y gestionan desde una única página.

- Los profesores pueden definir sus propias escalas que utilizarán para calificar las diferentes actividades.

- Ciertas actividades (como los foros y comentarios) están integradas con el correo electrónico de los participantes.

- El contenido de los cursos puede ser reutilizable para otras sesiones e incluso para cursos diferentes.

3.2. Arquitectura

3.2.1. Módulo de actividades

La plataforma ofrece una amplia lista de actividades que pueden ser desarrolladas a lo largo de un mismo curso. A continuación se muestra una breve descripción de los módulos principales de actividades:

- **Tareas**. Permiten al profesor asignar tareas (ensayos, monografías, redacciones,..) a los estudiantes. Opcionalmente, se puede especificar una fecha final de entrega y una calificación máxima que se podrá asignar al estudiante. Existen tres tipos de tareas:

  - Actividad fuera de línea. La tarea se realiza fuera de la plataforma. Sólo se mostrará una descripción de la misma. El profesor puede calificarla desde la plataforma y el estudiante ver la calificación obtenida.

  - Subir un único archivo. El estudiante ha de elaborar un documento (texto, imagen, archivo comprimido,…) y subirlo a la plataforma. La calificación se realiza en línea.

  - Tarea de texto en línea. El estudiante ha de editar un texto utilizando las herramientas proporcionadas por la plataforma. Los profesores pueden calificar e incluir comentarios. En versiones antiguas este tipo de tareas se englobaban en el *modulo Diario*.

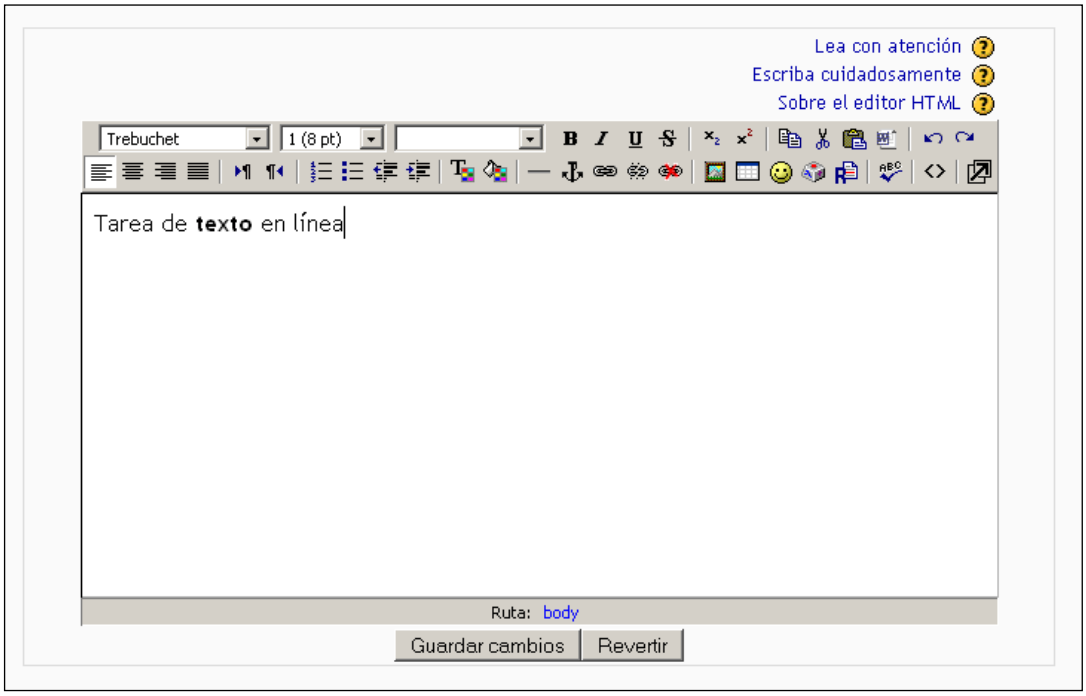


Figura 3.2.1-1. Ejemplo de tarea en línea.

- **Chats**. Permiten la comunicación en tiempo real de todos los participantes de un curso. Se pueden crear diferentes canales de conversación y utilizar emoticonos, direcciones URL e imágenes. Aunque se pueden registrar las sesiones, no es una actividad calificable.

- **Consultas**. Los profesores realizan una pregunta y muestran una lista de posibles respuestas (de 2 a 6) entre las cuales los estudiantes deberán elegirán sólo una. Es útil para la creación de encuestas rápidas (votaciones) o para que los estudiantes decidan sobre cualquier tema.

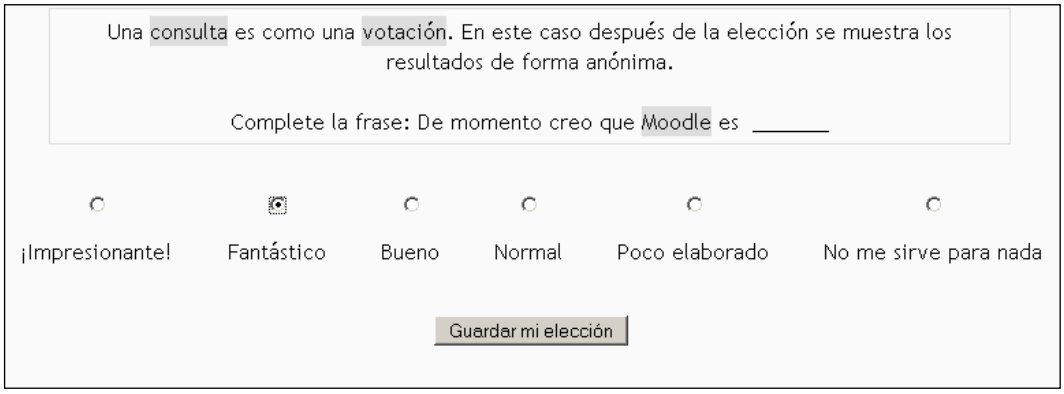


Figura 3.2.1-2. Ejemplo de consulta rápida.

- **Foros**. Se componen de un nombre, descripción y una lista de temas abiertos. Permite el intercambio de mensajes públicos del grupo sobre un tema compartido. La participación en foros puede ser una parte integral de la experiencia de aprendizaje, ayuda a los alumnos a aclarar y desarrollar su comprensión del tema.

  Existen foros estándar de uso general (abierto a cualquiera que quiera iniciar un nuevo tema de debate) y específicos de alguna sección concreta de un curso. Existe la posibilidad de activar la opción RSS para poder leer los mensajes a través de un lector (por ejemplo, en la página personalizada del buscador *Google*). Es un intercambio de ideas sobre un tema y todo en una misma página.

  Como parte de los objetivos de este proyecto está la implementación de un foro estructurado que permitirá separar la especificación, elaboración y consenso de los temas debatidos.

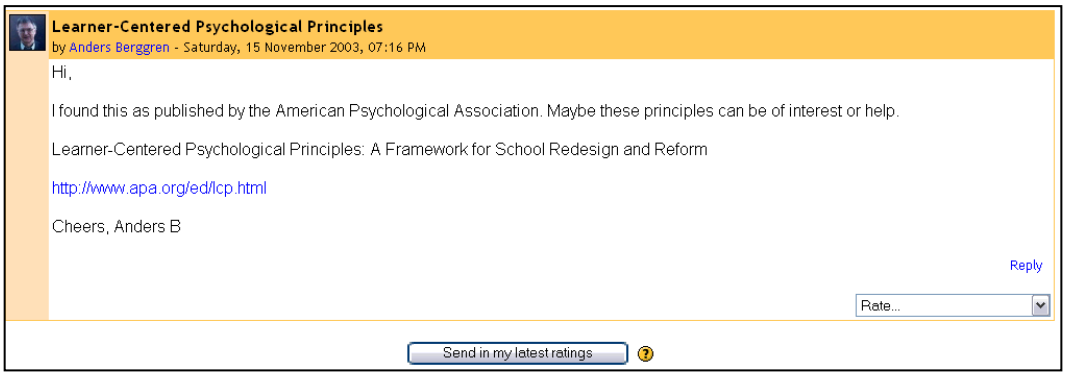


Figura 3.2.1-3. Ejemplo de mensaje de un foro.

- **Glosarios**. Son una recopilación de términos o palabras definidas que suelen ser conceptos relevantes de un curso. Se pueden exportar de un curso a otro (mediante XML), añadir enlaces y categorizar. Son accesibles a través del menú principal o través de una etiqueta que haya sido definida. Pueden ser creados tanto por el profesor como por los estudiantes.

- **Lecciones**. Proporcionan contenidos al curso a través de páginas web. Cada página suele proponer una pregunta y un número de respuestas posibles. Según la respuesta elegida por el estudiante, éste podrá avanzar en la lección o bien volver atrás hasta que responda correctamente a la pregunta. El profesor califica el proceso seguido por el estudiante, una vez haya finalizado la lección.

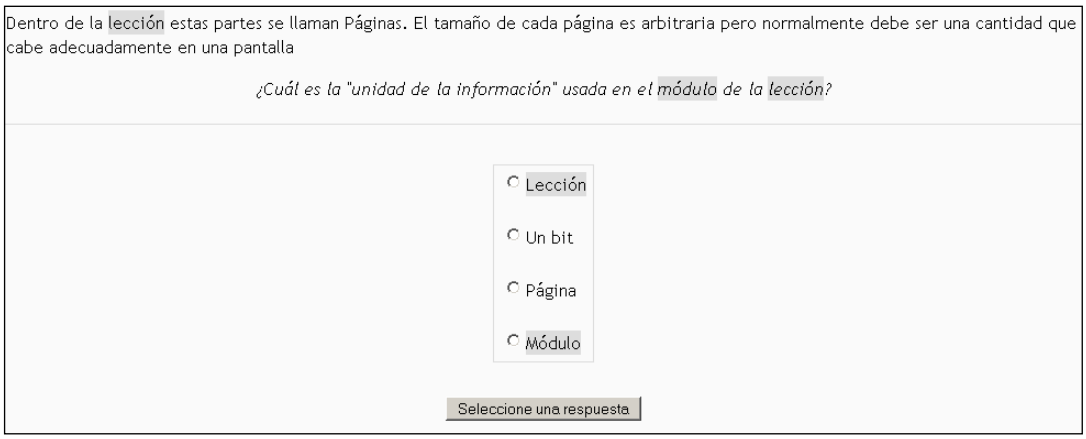
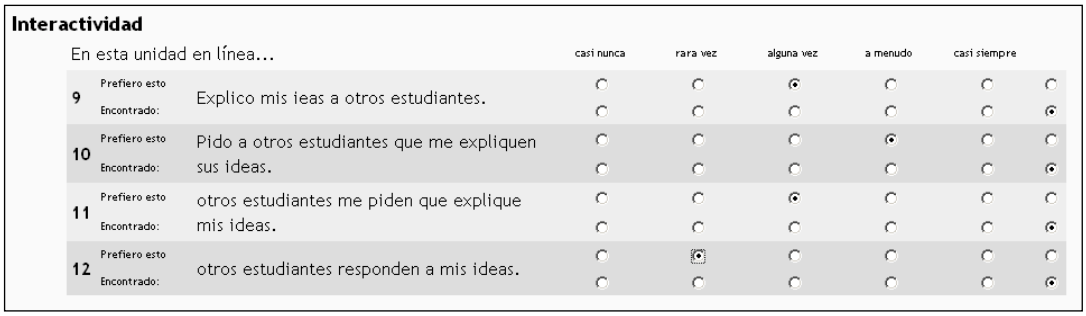


Figura 3.2.1-4. Ejemplo de lección.

- **Encuestas**. Sirven al profesor para recapitular información acerca de sus estudiantes con el fin de mejorar el proceso de enseñanza. Sólo interesa su opinión, no existen respuestas correctas o incorrectas, no debe afectar a las calificaciones, los informes estarán siempre disponibles a través de gráficos y tablas. También existen encuestas ya preparadas (llamadas COLLES y ATTLS).



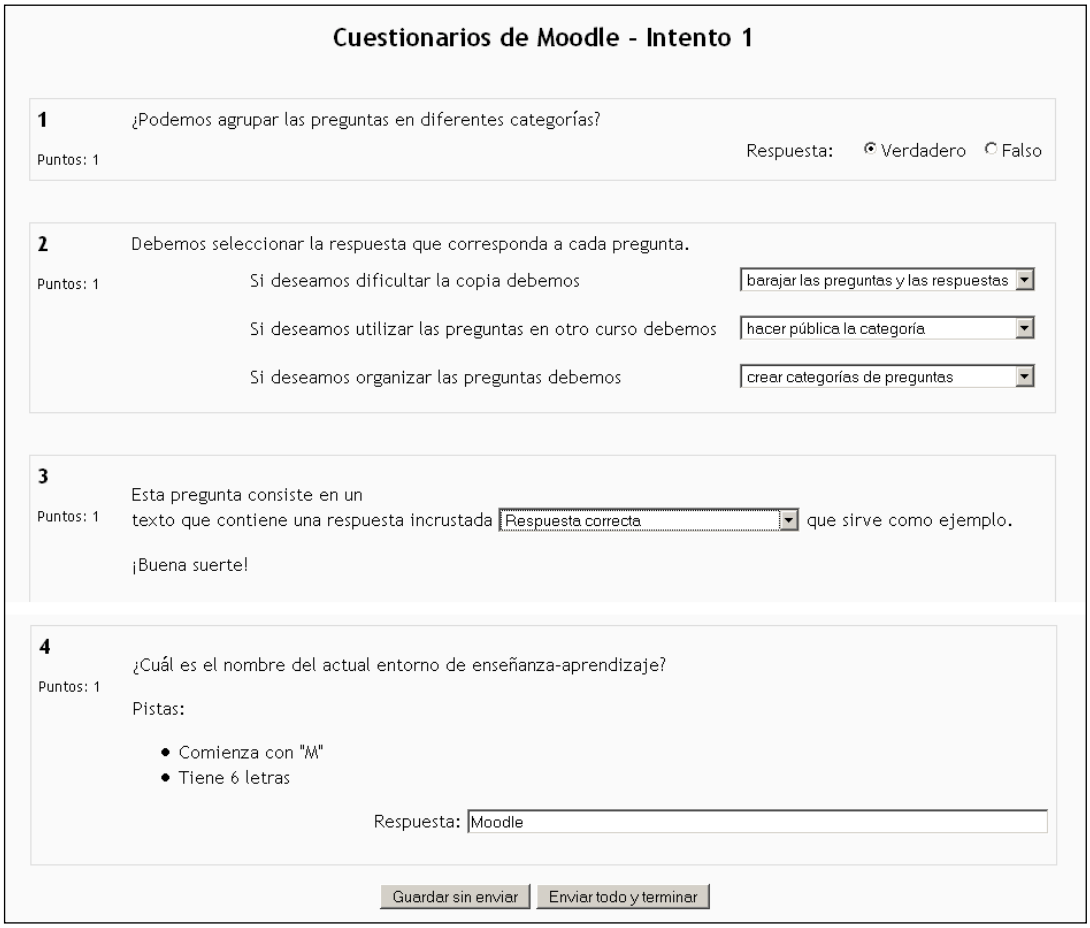
Interactividad

En esta unidad en línea...

		casi nunca	rara vez	alguna vez	a menudo	casi siempre	
9	Prefiero esto			●			
	Encontrado:						●
10	Prefiero esto				●		
	Encontrado:						●
11	Prefiero esto			●			
	Encontrado:						●
12	Prefiero esto		●				
	Encontrado:						●

Figura 3.2.1-5. Ejemplo de encuesta.

- **Cuestionarios**. También llamados exámenes. Consisten en un conjunto de preguntas (opción múltiple, respuesta corta, verdadero-falso, emparejar, preguntas aleatorias, numéricas, incrustadas en texto y texto y gráficos descriptivos) que sirven al profesor para calificar el proceso de aprendizaje de cada estudiante. Se pueden categorizar, reutilizar, configurar para que se muestren las respuestas correctas o no, configurar para poder ser realizados varias veces o no y establecer un límite de tiempo para la realización del mismo.



Cuestionarios de Moodle - Intento 1

1 — Puntos: 1
¿Podemos agrupar las preguntas en diferentes categorías?
Respuesta: Verdadero Falso

2 — Puntos: 1
Debemos seleccionar la respuesta que corresponda a cada pregunta.
Si deseamos dificultar la copia debemos — barajar las preguntas y las respuestas
Si deseamos utilizar las preguntas en otro curso debemos — hacer pública la categoría
Si deseamos organizar las preguntas debemos — crear categorías de preguntas

3 — Puntos: 1
Esta pregunta consiste en un texto que contiene una respuesta incrustada [Respuesta correcta] que sirve como ejemplo.
¡Buena suerte!

4 — Puntos: 1
¿Cuál es el nombre del actual entorno de enseñanza-aprendizaje?
Pistas:
- Comienza con "M"
- Tiene 6 letras

Respuesta: Moodle

Guardar sin enviar | Enviar todo y terminar

Figura 3.2.1-6. Ejemplo de cuestionario en el cual aparecen cuatro tipos de respuestas (verdadero-falso, emparejar, respuesta incrustada y respuesta corta).

- **Hot Potatoes**. Permiten añadir otros tipos de cuestionarios diferentes a los ofrecidos por el *módulo cuestionarios*

Existen varios tipos: *JQuiz* (cuestionario interactivo), *JCloze* (completar frases), *JCross* (palabras cruzadas), *Sequitur* (edición de texto) y *RhuBard* (rellenar huecos).

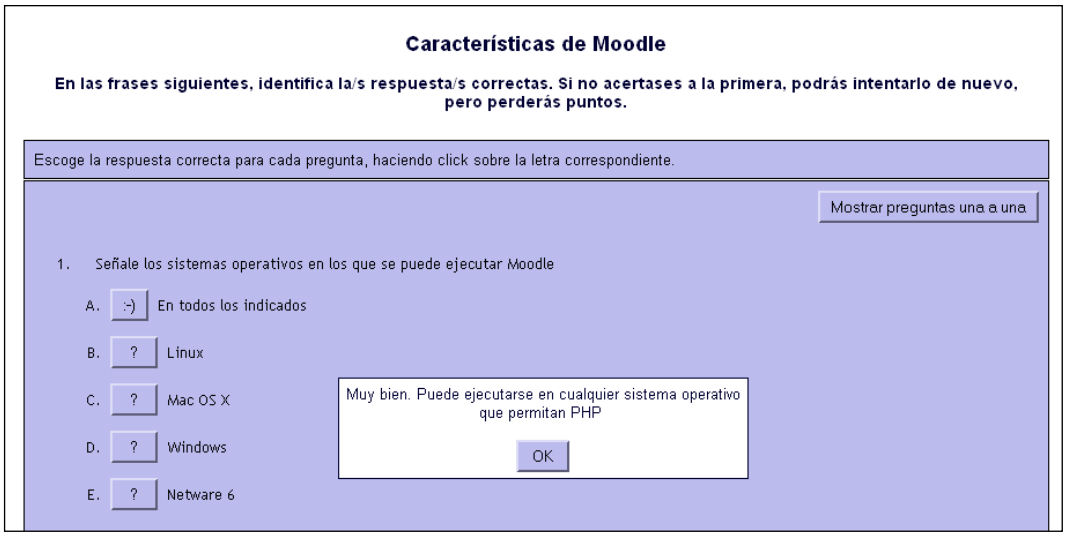


Figura 3.2.1-7. Ejemplo del *Hot Potatoe JQuiz*.

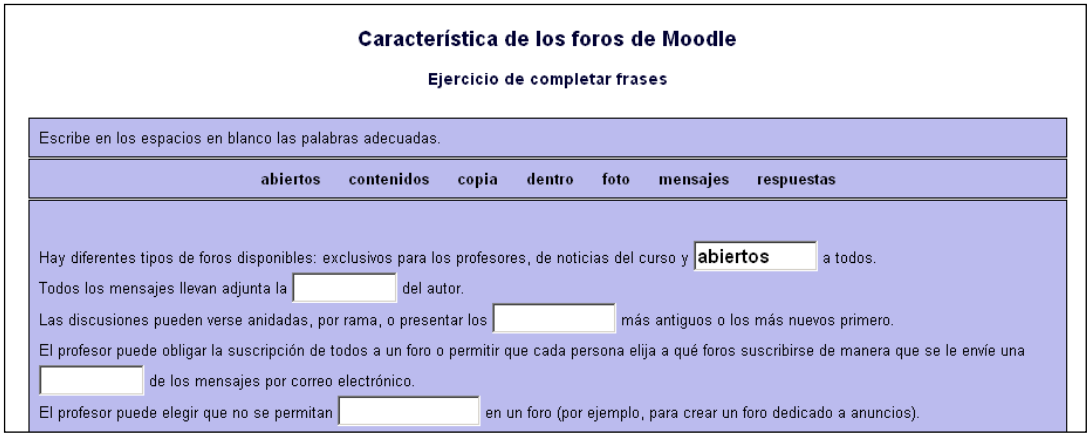


Figura 3.2.1-8. Ejemplo del *Hot Potatoe JCloze*.

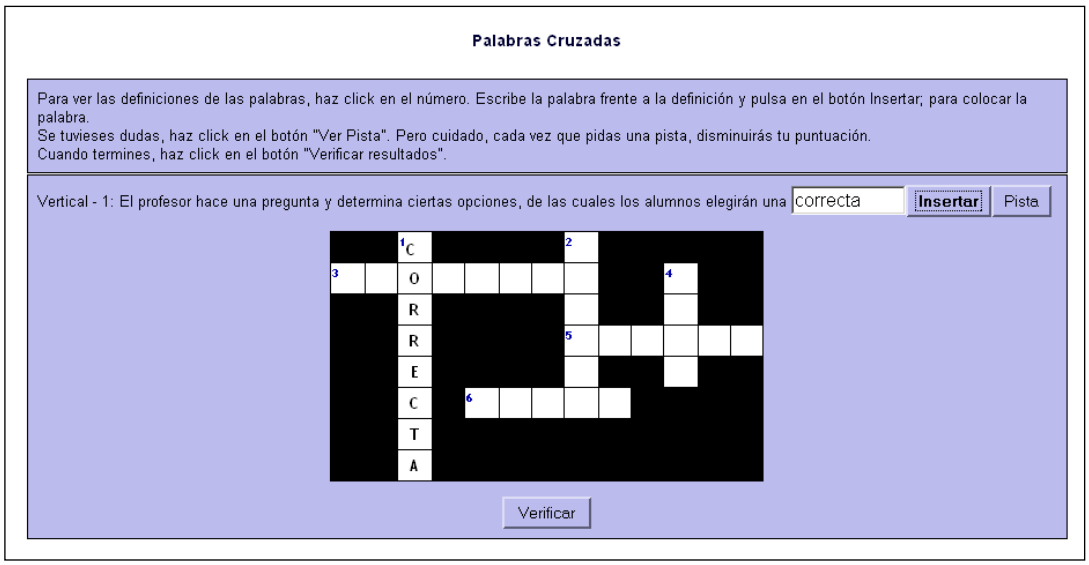


Figura 3.2.1-9. Ejemplo del *Hot Potatoe JCross*.

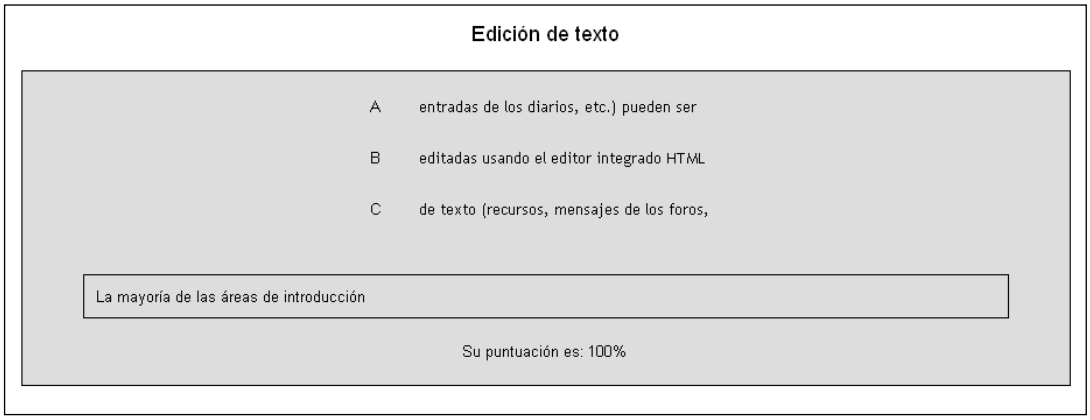


Figura 3.2.1-10. Ejemplo del *Hot Potatoe Sequitur*.

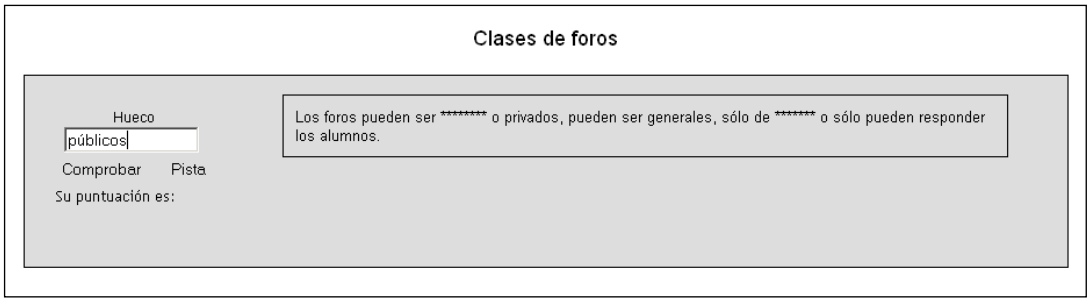


Figura 3.2.1-11. Ejemplo del *Hot Potatoe RhuBard*.

- **SCORMs**. También llamados Recursos. Material Web empaquetado (páginas, gráficos, clips multimedia,…) que puede ser exportado o añadido a un curso y que ofrece información sobre de las consultas realizadas Los estados pueden ser: navegado (flecha azul), incompleto (naranja), completado (aspa verde) y no se ha intentado (cuadro sin rellenar).

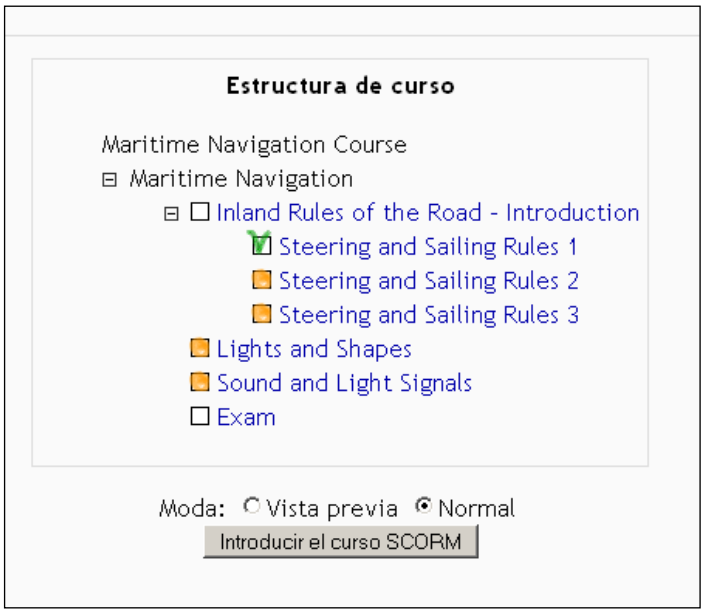
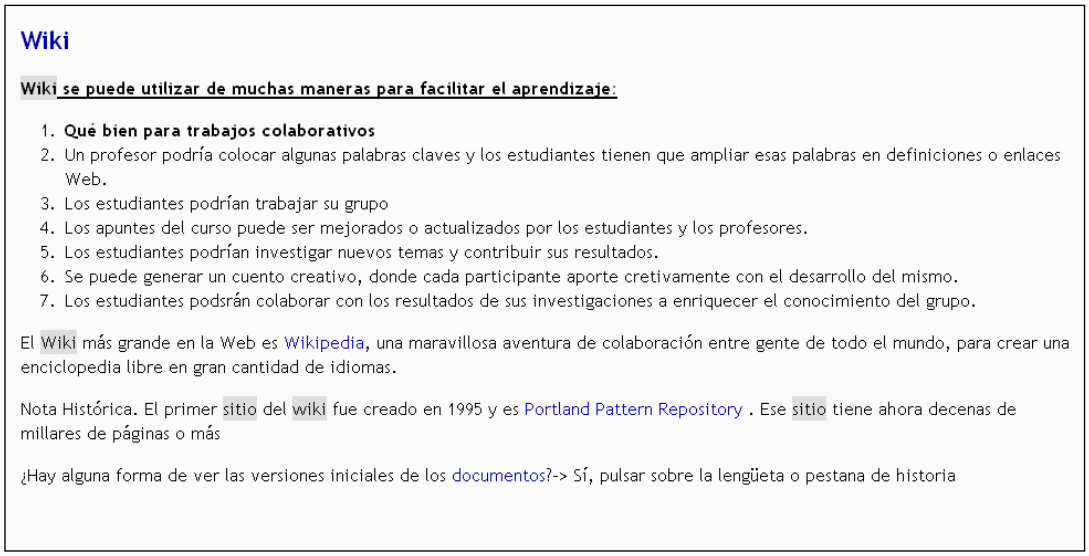


Figura 3.2.1-12. Ejemplo del SCORM.

- **Talleres**. Junto a los foros de debate, es uno de los módulos de actividades más orientado al trabajo colaborativo en grupo. Permite a los estudiantes crear proyectos-prototipo y evaluar los de otros. El profesor califica los trabajos y las evaluaciones realizadas por los estudiantes. El profesor puede suministrar documentos de ejemplo a los estudiantes para practicar la evaluación.

- **Base de datos**. Espacio para compartir archivos (texto, imágenes, archivos, enlaces) accesible tanto para estudiantes como para profesores. Está disponible a partir de la versión 1.6 de *Moodle*.

- **Wikis**. Permiten crear, expandir o modificar contenidos de manera conjunta a través de documentos basados en marcas. Normalmente no se realizan revisiones antes de aceptar las modificaciones ya que son los propios participantes los encargados de su revisión.



Wiki

Wiki se puede utilizar de muchas maneras para facilitar el aprendizaje:

1. **Qué bien para trabajos colaborativos**
2. Un profesor podría colocar algunas palabras claves y los estudiantes tienen que ampliar esas palabras en definiciones o enlaces Web.
3. Los estudiantes podrían trabajar su grupo
4. Los apuntes del curso puede ser mejorados o actualizados por los estudiantes y los profesores.
5. Los estudiantes podrían investigar nuevos temas y contribuir sus resultados.
6. Se puede generar un cuento creativo, donde cada participante aporte cretivamente con el desarrollo del mismo.
7. Los estudiantes podsrán colaborar con los resultados de sus investigaciones a enriquecer el conocimiento del grupo.

El Wiki más grande en la Web es Wikipedia, una maravillosa aventura de colaboración entre gente de todo el mundo, para crear una enciclopedia libre en gran cantidad de idiomas.

Nota Histórica. El primer sitio del wiki fue creado en 1995 y es Portland Pattern Repository . Ese sitio tiene ahora decenas de millares de páginas o más

¿Hay alguna forma de ver las versiones iniciales de los documentos?-> Sí, pulsar sobre la lengüeta o pestana de historia

Figura 3.2.1-13. Ejemplo de Wiki.

Existen otros tipos de módulos de actividades no estándar que amplían las capacidades de los módulos principales. Éstos han de ser descargados aparte ya que no vienen en la distribución oficial. Algunos de estos módulos son:

- **Base de datos de imágenes**, **Flash y objetos**.

- **Libros**. Material sencillo compuesto por múltiples páginas con estructura de libro. Permite imprimir los contenidos de forma completa o por capítulos.

- **DFWiki**. Módulo alternativo al *Wiki* stándard con mejoras en la edición de contenidos y en la existencia de bloques de ayuda.
- **Dialogue**. Permiten enviar mensajes privados con retroalimentación. Es semejante al correo electrónico.

- **Ejercicios**. Tarea práctica evaluable por el profesor que permite además al estudiante autoevaluarse antes de enviarla. Se compone de tres fases:

    1. Configurar el ejercicio creando la tarea y seleccionando el criterio de evaluación.
    2. Permitir evaluaciones y envíos de los estudiantes.
    3. Mostrar las calificaciones generales.

- **Jmol resource**. Visualización interactiva de estructuras moleculares en 3D.

- **Correo**. Similar al módulo *Dialogue* que como ventaja permite enviar mensajes masivos.

- ***Moodle* for Móviles Quiz**. Permite realizar cuestionarios *Moodle* a través de teléfonos móviles con i-node.

- **Presentación**. Permite a los profesores crear recursos con apariencia y comportamiento de presentaciones a través de diapositivas.

- **Cuestionario phpESP**. Módulo alternativo al *módulo de encuestas* basado en phpESP que es una herramienta de libre distribución para elaborar encuestas.

- **Scheduler**. Permite al profesor programar citas para entrevistarse presencialmente con los estudiantes especificando el día y la duración de las mismas. También permite registrar la asistencia a dichas citas.

- **Módulo Test de Identificación Inconsciente (Tui)**. Permite realizar test psicológicos.

- **Netpublish**. Permite a los usuarios (profesores, estudiantes, administrador) crear doucmentos en común de múltiples páginas (revistas, prensa,…) que pueden ser publicadas de forma pública. Es posible calificar cada artículo.

- **Return folder**. Permite crear espacios a través de los cuales los estudiantes puedan subir archivos que pueden ser explorados por el profesor. No es una actividad calificable.

3.2.2. Recursos

Los recursos (no confundir con el módulo de actividades *SCORMs* - Recursos) son herramientas que permiten al profesor facilitar la información necesaria a los estudiantes para poder llevar a cabo las actividades del curso. Existen diferentes tipos de recursos:

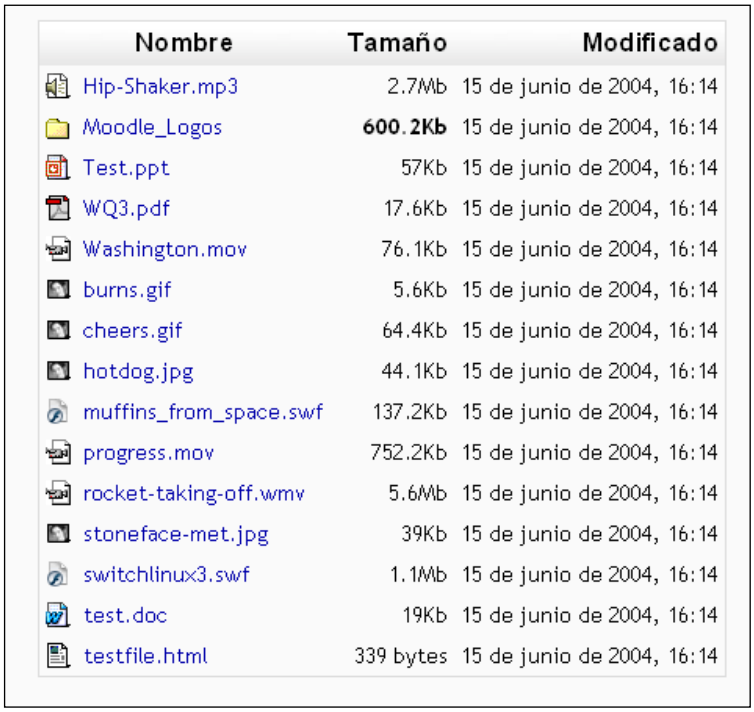
- **Páginas de texto**. Son páginas escritas en texto plano que pueden ser convertidas a formato de página Web.

- **Páginas Web**. Permite la creación de páginas web dentro de la plataforma a través de un editor HTML. Es posible incluir código *Javascript*.

- **Enlaces a archivos o Webs**. Permite añadir contenidos externos de forma rápida en diferentes formatos (contenido incrustado, ventanas emergentes, marcos,..). Los archivos multimedia se tratan de modo inteligente permitiendo mostrar reproductores de audio, vídeo y animaciones Flash.

- **Etiquetas**. Permiten añadir comentarios e imágenes en los módulos de actividades.

- **Mostrar un Directorio**. Sirve para incluir contenidos navegables de un curso (texto, archivos, enlaces, Wiki, HTML o una referencia bibliográfica). Los estudiantes pueden navegar y descargar todos los archivos de un directorio.



Nombre	Tamaño	Modificado
Hip-Shaker.mp3	2.7Mb	15 de junio de 2004, 16:14
Moodle_Logos	**600.2Kb**	15 de junio de 2004, 16:14
Test.ppt	57Kb	15 de junio de 2004, 16:14
WQ3.pdf	17.6Kb	15 de junio de 2004, 16:14
Washington.mov	76.1Kb	15 de junio de 2004, 16:14
burns.gif	5.6Kb	15 de junio de 2004, 16:14
cheers.gif	64.4Kb	15 de junio de 2004, 16:14
hotdog.jpg	44.1Kb	15 de junio de 2004, 16:14
muffins_from_space.swf	137.2Kb	15 de junio de 2004, 16:14
progress.mov	752.2Kb	15 de junio de 2004, 16:14
rocket-taking-off.wmv	5.6Mb	15 de junio de 2004, 16:14
stoneface-met.jpg	39Kb	15 de junio de 2004, 16:14
switchlinux3.swf	1.1Mb	15 de junio de 2004, 16:14
test.doc	19Kb	15 de junio de 2004, 16:14
testfile.html	339 bytes	15 de junio de 2004, 16:14

Figura 3.2.2-1. Ejemplo de visualización de un directorio.

3.2.3. Bloques

Los bloques son módulos (actividades, recursos, herramientas,..) que un profesor puede añadir, modificar o borrar de las columnas izquierda y derecha de la página principal de un curso. Existen varios tipos de bloques:

- **Actividades**. Lista todas las categorías de actividades disponibles de un curso (ver sección *3.1 Módulos de actividades*). Cuando se crea un curso nuevo, este bloque sólo muestra la actividad *Foros*, representando al foro novedades del curso. A medida que se añaden nuevas actividades al curso, el bloque irá mostrándolas.

Figura 3.2.3-1. Visualización del bloque de actividades.

- **Administración**. Permite a los estudiantes consultar las calificaciones obtenidas hasta el momento, obtener un informe de actividades realizadas, editar su perfil, cambiar la contraseña y desapuntarse de un curso.

- **Calendario**. Muestra eventos relevantes. Estos eventos pueden ser:

    - Del sitio. Son creados por el administrador y visibles para todos los grupos.

    - De un curso Son creados por el profesor y visibles para todos los participantes del curso.

    - De grupo Son creados por el profesor y visibles para todos los miembros de un grupo de usuarios.

    - De usuario. Son creados y visibles sólo por el usuario.

- **Cursos**. Lista todos los cursos en los cuales está matriculado o es profesor un usuario.

- **Descripción de un curso o del sitio**. Muestra una breve descripción de un curso a través del texto descriptivo de la configuración del mismo.

- **HTML**. Permite agregar contenido estático (HTML + *Javascript*).

- **Novedades**. Muestra las últimas noticias o mensajes aparecidos en el foro de novedades de un curso.

- **Mensajes**. *Moodle* dispone de un sistema de mensajería interna entre los usuarios de la plataforma. El uso didáctico de este bloque es muy importante ya que sirve de comunicación privada entre profesor y estudiantes y como advertencia sobre la evolución de la actitud, progresión o solicitud de requisitos o justificaciones de carácter personal.

- **Usuarios en línea**. Muestra la lista de los usuarios del curso (profesores o estudiantes) conectados en un período de tiempo fijado por el administrador.

- **Personas**. Muestra una lista completa de todos los participantes de un curso. Al pulsar sobre el enlace de cualquier participante nos muestra su información personal así como sus datos de contacto.

- **Resultados de cuestionario**. Permite mostrar públicamente un listado de los estudiantes que han obtenido mejor resultado en un cuestionario. Por seguridad y respeto al nivel de resultados obtenido, no debería hacerse público el resultado de aquellos cuestionarios formales o de aprendizaje.

- **Entrada aleatoria del Glosario**. Muestra de forma aleatoria la definición de un término del glosario del curso.

- **Actividad reciente**. Muestra información de los eventos próximos del calendario. SI el evento es una actividad puede ser enlazada directamente.

- **Canales RSS remotos.** Permite mostrar contenido RSS de sitios web externos.

- **Buscar en los foros**. Buscador de mensajes en los foros mediante palabras clave. Mostrará diferente resultado en función del tipo de acceso (invitado, estudiante, profesor o administrador).

- **Enlaces de sección**. Permite navegar de manera cómoda de una sección o semana particular del curso a otra dependiendo del formato del curso (sólo para los formatos semanal o temático).
- **Bloques no estándar**. Son bloques que han de ser descargados aparte ya que no vienen en la distribución oficial. De entre todos los bloques disponibles destacan:

    - **Course menú**. Permite mostrar las secciones/semanas de un curso en forma de carpetas.

- **Glossaries Search**. Permite realizar búsquedas en todos los glosarios de un curso.
- **Multiple Search**. Permite realizar búsquedas en varios buscadores de internet.
- **Marking**. Resumen de las actividades pendientes de ser calificadas y de los estudiantes que no han superado una cierta calificación.
- **Personal Bookmarks**. Lista personal de favoritos (enlaces).

Este listado es sólo un pequeño esbozo de los bloques disponibles para la plataforma. Pueden obtenerse muchos de ellos desde la dirección de Internet http://cvs.sourceforge.net/viewcvs.py/*Moodle*/contrib/

3.2.4. Tecnología

Moodle es una aplicación Web que para poder ejecutarse necesita estar hospedada en un servidor Web que tenga soporte para lenguaje PHP y un **sistema gestor de base de datos (SGBD)** para gestionar los datos que maneja. Es un sistema multiplataforma que ha sido testeado en diversos entornos (Microsoft Windows, Linux/Unix, MacOS X y Netware 6) aunque lo normal es instalarlo en entornos **LAMP** (Linux + Apache + MySQL + PHP).

En concreto, necesita el siguiente entorno de ejecución:

- Un servidor Web que soporte PHP. Usualmente suele utilizarse **Apache**, aunque también puede utilizarse otros servidores como *Internet Information Server* (IIS) para plataformas *Microsoft Windows*.

- Una instalación de **PHP** (versión 4.3.0 o posterior). PHP5 está soportado a partir de *Moodle* 1.4.

- Un SGBD. La utilización de la librería ADOdb permite la abstracción de base de datos lo que significa que *Moodle* puede usar más de diez marcas diferentes de bases de datos. No obstante, en la actualidad sólo están implementados diagramas para **MySQL** o **PostgreSQL**.

La página oficial ofrece varias alternativas para la instalación de los requerimientos dependiendo del entorno utilizado. Éstos pueden instalarse cada uno por separado u optar por alguna de las soluciones propuestas:

- MacOs X. La forma más sencilla consiste en utilizar el servidor web Apache que proporciona *Apple* y añadir PHP y MySQL por separado.

- RedHat Linux. Instalar todos los paquetes RPM disponibles para Apache, PHP y MySQL.

- Microsoft Windows. La forma más sencilla consiste en utilizar uno de los paquetes completos de instalación disponibles. De forma alternativa, existen paquetes (como *EasyPHP*, *XAMPP*, *AppServ*...) que incluyen todo lo necesario para la instalación de los requerimientos.

Para la realización de este estudio se ha optado por instalar *Moodle* bajo plataforma **WAMP** (Windows + Apache + MySQL + PHP). Para ello se ha utilizado el paquete software **AppServ** (versión 2.5.5) que proporciona una instalación y configuración completa de los requerimientos. En concreto, la citada versión configura e instala los siguientes componentes:

- Apache 2.0.55
- MySQL 5.0.15
- PHP 5.1.1
- phpMyAdmin-2.6.4-pl4 (Front-end de gestión para MySQL)

En el manual de administrador del foro estructurado se muestra una guía de instalación rápida de *Moodle*. En este manual se describe los pasos de instalación para una plataforma WAMP utilizando el paquete *AppServ*; el proceso es muy parecido en otros entornos y no debería reportar mayor dificultad.

3.2.5. API principal

Moodle cuenta con una extensa **API** (interfaz de programación de aplicaciones) para describir la lista existente de especificaciones y comunicación entre componentes. Esta API está organizada a través de packages y cada uno de ellos contiene una lista de clases con métodos y atributos.

Los principales packages que componen la API son:

- `moodlecore`. Se describen las clases principales del sistema.
- `blocks`. Definiciones de bloques. La clase `block_base()` es muy utilizada para crear bloques de texto sencillos.
- `moodleauth`.y `auth`. Clases relacionadas con la autentificación de usuarios.

- `Xmlschema` y `axmls`. Clases de soporte para XML.
- `Spreadsheet_WriteExcel`
- `pages`. Clases para la gestión de páginas.
- `PHPMailer`. Clases de correo electrónico.
- `TYPO3`. Clases relacionadas con el CMS.

La lista completa de clases, métodos y atributos de estos packages es demasiado extensa como para exponerla en este documento. Ésta puede consultarse directamente desde la dirección http://*Moodle*.org/mod/resource/view.php?id=1267

3.3. Especificación de componentes

3.3.1. Normas y convenios

Antes de empezar a desarrollar nuevos componentes (módulos, bloques, recursos,...), hay que tener en cuenta que *Moodle* es un proyecto de desarrollo colaborativo (no confundir con entorno colaborativo, que también lo es) y por tanto es necesario establecer una serie de normas y convenios respecto al código, estilo, estructura de la base de datos y seguridad para ofrecer consistencia y estabilidad.

A **nivel general** hay que tener en cuenta las siguientes reglas:

1. Todos los archivos con código han de utilizar la extensión .php.

2. Todas las plantillas han de utilizar la extensión .html.

3. Todos los archivos de texto han de utilizar el formato de texto Unix.

4. Las etiquetas PHP han de ser completas <?php?>, no se permite <??>.

5. Todos los avisos de *copyright* han de ser mantenidos. Como ya se ha comentado, *Moodle* se basa en licencia GNU.

6. Todos los archivos han de incluir el archivo principal config.php.

7. Cada archivo ha de comprobar que el usuario está autenticado correctamente.

8. Todos los accesos a la base de datos deberán utilizar las funciones definidas en la librería `lib/datalib.php` por compatibilidad con otros SGBD.

9. Sólo se han de utilizar variables globales estándar.

10. Todas las variables han de ser inicializadas o comprobado su existencia.

11. Todas las cadenas y ficheros de ayuda han de ser traducibles.

12. Toda la información obtenida desde archivos o base de datos ha de ser tratada antes de insertarla en la base de datos.

13. Todos los textos, especialmente aquellos que han sido introducidos por los usuarios, deben ser filtrados y limpiados de caracteres extraños.

Respecto al **estilo de código** se han de cumplir las siguientes normas:

1. El sangrado del texto debe ser siempre de cuatro espacios, nunca tabuladores.

2. Los nombres de las variables han de ser fáciles de leer, procurando que sean palabras en minúsculas con significado en inglés. Si se necesita más de una palabra, se han de poner juntas de forma breve. Se han de utilizar nombres en plural para las matrices de objetos.

3. Las constantes han de definirse siempre en mayúsculas, y empezar siempre por el nombre del módulo al que pertenecen. Han de tener las palabras separadas por guiones bajos.

4. Los nombres de las funciones han de ser palabras sencillas en minúsculas y en inglés, y empezar con el nombre del módulo al que pertenecen para evitar conflictos. Las palabras han de separarse por guiones bajos. Los parámetros, si es posible, tendrán valores por defecto. No ha de haber espacios entre el nombre de la función y el siguiente paréntesis.

5. Los bloques de código siempre deben estar encerrados por llaves, aunque sólo consten de una línea.

6. Las cadenas tienen que ser definidas utilizando comillas simples siempre que sea posible, para obtener un mejor rendimiento.

7. Los comentarios deben ser añadidos de forma que resulten prácticos, para explicar el flujo del código y el propósito de las funciones y variables.

8. Cada función y clase han de utilizar el popular formato **phpDoc**. Esto permite que la documentación sea generada automáticamente.

9. Los comentarios en línea deberían utilizar los caracteres //, alineados con cuidado por encima de las líneas de código que comenta.

10. Generalmente, ha de haber un espacio entre llaves y líneas normales y ninguno entre llaves y variables o funciones.

Respecto a la **estructura de la base de datos** se ha de tener en cuenta las siguientes normas:

1. Cada tabla ha de tener un campo auto numérico *id* (INT10 – entero de longitud 10) como clave primaria.

2. La tabla principal que contiene instancias de cada módulo debe tener el mismo nombre que el módulo y contener, por lo menos, los siguientes campos:

    a. id. Clave primaria.
    b. course. Identificador del curso al que la instancia pertenece.
    c. name. Nombre completo de la instancia.

3. El resto de las tablas asociadas han de tener como nombre el prefijo del módulo al cual pertenecen más el propio nombre de la tabla.

4. Los nombres de los campos han de ser sencillos y cortos, siguiendo las mismas reglas que los nombres de las variables.

5. Cuando sea posible, los campos que contengan una referencia al campo id de otra tabla (por ejemplo, modulo) debería ser llamado *moduloid*.

6. Los campos lógicos serán implementados como enteros cortos (por ejemplo, INT4) con los valores 0 o 1, para permitir la futura expansión de los valores si fuera necesario.

7. La mayoría de las tablas tienen que tener un campo *timemodified* (INT10) que será actualizado con la fecha actual.

Respecto al **estilo de la interfaz** se establecen las siguientes recomendaciones:

1. Hay que conseguir una interfaz lo más sencilla posible.

2. Es necesario utilizar una plantilla de página imprimiendo las cabeceras con la función `print_heading()` y las cajas usando `print_simple_box()` (package *Moodle*core).

3. Es necesario utilizar una plantilla de formulario, mostrando las opciones más importantes en la parte superior, indicando una etiqueta y/o archivo de ayuda para cada entrada y desglosando las opciones en parámetros necesarios, opcionales, extra o avanzados.

4. Todas las páginas han de llamar a la función print_header(), y suministrar una ruta de navegación estándar (CURSO >> ÍNDICE >> INSTANCIA >> SUBPAGINAS.).

5. Las páginas incluidas en módulos de actividad han de llamar a la función navmenu() para generar el menú de navegación apropiado.

6. Las direcciones URL deben ser lo más cortas posible.

7. No utilizar subrayado en nombres de parámetros o nombres de archivos.

8. Nunca utilizar dos palabras cuando una sea suficiente.

9. Los botones de ayuda deben estar a la derecha del objeto (excepcionalmente pueden estar a la izquierda, si el objeto está alineado a la derecha).

10. En las tablas hay que utilizar la función `print_table()` cuando sea posible.

Por último, se establecen las siguientes **normas de seguridad**:

1. Cada variable debe ser correctamente inicializada en cada fichero de código.

2. No se ha de utilizar la función `optional_variable()`. En su lugar, se ha de utilizar la función `optional_param()`.

3. No se ha de utilizar la función `require_variable()`. En su lugar, se ha de utilizar la función `required_param()`.

4. Para recuperar valores enviados desde formularios (métodos GET y POST) se ha de utilizar las funciones `required_param()` o `optional_param()` apropiadas.

5. No se ha de utilizar funciones en accione comparativas, sino que primero se ha de asignar el valor de la función a una variable y después realizar la comparación.

6. Se ha de agrupar todas las llamadas a `required_param()`, `optional_param()` y el resto de inicialización de variables en el principio de cada fichero (o función).

7. Todos los nombres de ficheros deben ser limpiados de caracteres extraños utilizando la función `clean_filename()`.

8. Cuando sea posible, la información que se almacenará en la base de datos debe venir de peticiones POST en lugar de peticiones GET.

9. La información enviada a la base de datos ha sido filtrada mediante la función `clean_param()` y `addslashes()`.

10. Hay que comprobar la falta de comillas en las variables utilizadas en sentencias SQL. Es un punto de entrada para ataques de tipo *SQL Injection* (introducción de sentencias SQL en campos de usuario que si no es tratado puede provocar errores en la aplicación).

3.3.2. Especificación de módulos de actividades

Los módulos de actividades se encuentran en el directorio `/mod/` de la instalación de *Moodle*. Cada módulo ha de estar en un directorio diferente y ha de contener los siguientes elementos:

- mod.html. Formulario para asignar o actualizar una instancia del módulo.

- version.php. Define alguna meta-información y proporciona código de actualización.

- icon.gif. Icono 16x16 representativo del módulo.

- db/. Directorio donde se guardarán ficheros con sentencias SQL de todas las tablas y datos requeridos de una base de datos (para cada tipo de base de datos).

- index.php. Página para presentar la lista de todas las instancias del módulo en un curso.

- view.php. Página para mostrar una instancia en particular.

- lib.php. Fichero donde se definen todas las funciones del módulo. Si el módulo se llama structuredforum, entonces deben existir como mínimo las siguientes funciones:

    - `structuredforum_add_instance()`. Código para añadir una nueva instancia del módulo.
    - `structuredforum_update_instance()`. Código para actualizar una instancia existente.
    - `structuredforum_delete_instance()`. Código para eliminar una instancia existente.
    - `structuredforum_user_outline()`. Dada una instancia, devuelve un resumen de una contribución de un usuario.

- structuredforum_user_complete(). Dada una instancia, imprime detalles sobre la contribución de un usuario.

- Para evitar posibles conflictos, cualquiera de las funciones de un módulo debe ser nombrada comenzando por el nombre del módulo en minúsculas más un guión bajo (en nuestro caso structuredforum_) y cualquier constante definida de forma idéntica pero en mayúscula (en nuestro caso STRUCTUREDFORUM_).

- Finalmente, será necesario crear los ficheros de idioma de las cadenas de texto a mostrar dentro del directorio /lang/<lenguaje>/structuredforum.php, donde <lenguaje> corresponde al identificador del idioma (en castellano correspondría al directorio es/).

La página oficial ofrece una plantilla para facilitar la creación de nuevos módulos en la cual sólo hay que sustituir las apariciones de “NEWMODULE” por el nombre del módulo a desarrollar. A partir de este punto puede modificarle dicha plantilla al antojo del desarrollador.

3.3.3. Especificación de bloques

A la hora de especificar un nuevo componente en forma de bloque hemos de seguir los siguientes pasos:

1. Asignar al bloque un nombre representativo (por ejemplo `simplehtml`).
2. Crear un directorio específico dentro del directorio `/blocks/` de la instalación de *Moodle* (por ejemplo `/blocks/simplehtml/`).
3. Crear un fichero principal en el cual definiremos la clase principal del bloque (por ejemplo `/blocks/simplehtml/block_simplehtml.php`).
4. Definir la clase como extensión de la clase `block_base`.
5. Crear el método `init()` en el cual definamos un título descriptivo y la versión actual del bloque.

```
class block_simplehtml extends block_base {
    function init() {
        $this->title = get_string('simplehtml','block_simplehtml');
        $this->version = 2004111200;
    }
}
```

Figura 3.3.3-1. Ejemplo de definición de clase con método `init()`.

6. Crear el método `get_content()` que servirá para mostrar información por pantalla.

```
function get_content() {
    if ($this->content !== NULL) {
        return $this->content;
    }
    $this->content = new stdClass;
    $this->content->text = '¡El contenido del bloque SimpleHTML!';
    $this->content->footer = 'Pie de página aquí...';

    return $this->content;
}
```

Figura 3.3.3-2. Ejemplo de método `get_content()`.

7. Para que un profesor pueda configurar el bloque hemos de crear el método `instante_allow_config()` que mostrará un enlace de edición en la cabecera.

```
function instance_allow_config() {
    return true;
}
```

Figura 3.3.3-3. Configuración del método `instance_allow_config()`.

Será necesario además crear el fichero `config_instance.html` en el cual le indiquemos el tipo de configuración del bloque.

```
<table cellpadding="9" cellspacing="0">
 <tr valign="top">
   <td align="right">
     <?php print_string('configcontent', 'block_simplehtml'); ?>:
   </td>
   <td>
     <?php print_textarea(true, 10, 50, 0, 0, 'text', $this-
>config->text); ?>
   </td>
 </tr>
 <tr>
   <td colspan="2" align="center">
     <input type="submit"
       value="<?php print_string('savechanges') ?>" />
     </td>
 </tr>
 </table>
 <?php use_html_editor(); ?>
```

Figura 3.3.3-4. Ejemplo de configuración a través de un editor WYSIWYG.

8. Para que el bloque muestre la configuración guardada es necesario modificar el método `get_content()` de la clase e indicarle que muestre el contenido de la configuración.

```
$this->content = new stdClass;
$this->content->text = $this->config->text;
$this->content->footer = 'Pie de página aquí...';
```

Figura 3.3.3-5. Modificación del método `get_content()` para mostrar la configuración del bloque.

9. Existe la opción de permitir mostrar y/o ocultar el bloque según el contexto en el que se encuentre. Para ello podemos modificar el método `get_content()` para que no muestre nada en el caso que dicho bloque no tenga contenido.
10. Existe la opción de permitir varias instancias del mismo bloque deberemos añadir el método `instante_allow_multiple()` a la clase. En este caso, no es necesario el método `instante_allow_config()` ya que por defecto todas las instancias podrán ser configuradas.

```
function instance_allow_multiple() {
    return true;
}
```

Figura 3.3.3-6. Configuración del método `instance_allow_multiple()`.

11. Existe la opción de permitir cambiar la configuración global de todas las instancias de un mismo bloque. Para ello deberemos configurar primero el método `has_config()`.

```
function has_config() {
    return true;
}
```

Figura 3.3.3-7. Configuración del método `has_config()`.

Deberemos además crear el fichero `config_global.html` (en versiones antiguas se llamaba simplemente `config.html`) para mostrar la pantalla de configuración.

```
<div style="text-align: center;">
 <input type="hidden" name="block_simplehtml_strict" value="0" />
 <input type="checkbox" name="block_simplehtml_strict" value="1"

<?php
    if(!empty($CFG->block_simplehtml_strict)){
      echo 'checked="checked"';
    }
?> />
 <?php print_string('donotallowhtml', 'block_simplehtml'); ?>
```

```
 <p><input type="submit" value="<?php print_string('savechanges');
?>" /></p>
 </div>
```

Figura 3.3.3-8. Ejemplo de configuración global.

12. Existe la opción de restringir los valores de la configuración introducidos configurando adecuadamente el método `config_save()` (en versiones antiguas se llamaba `handle_config()`).

```
function config_save($data) {
    // por defecto graba todas las variables como propiedades $CFG
    foreach ($data as $name => $value) {
        set_config($name, $value);
    }
    return true;
}
```

Figura 3.3.3-9. Ejemplo de configuración del método `config_save()`.

13. Existe la opción de ocultar la cabecera de un bloque configurando el método `hide_header()`, aunque es necesario configurar el título en el método `init()`.

```
function hide_header() {
    return true;
}
```

Figura 3.3.3-10. Configuración del método `hide_header()`.

14. Existe la opción de indicar el ancho en píxeles del bloque configurando el método `preferred_width()`.

```
function preferred_width() {
    // El valor preferido está en píxeles
    return 200;
}
```

Figura 3.3.3-11. Ejemplo de configuración del método `prederred_width()`.

15. Existe la opción de asignar hojas de estilos CSS y comportamientos a los bloques configurando el método `html_attibutes()`.

```
function html_attributes() {
    return array(
        'class'       => 'sideblock block_'. $this->name(),
        'onmouseover' => 'alert("Mouseover on our block!");'
    );
}
```

Figura 3.3.3-12. Ejemplo de configuración del método `html_attibutes()`.

16. Existe la opción de restringir el formato de los bloques configurando adecuadamente el método `aplicable_formats()`. En versiones antiguas sólo se permitían en los cursos.

```
function applicable_formats() {
    return array('site' => true);
}
```

Figura 3.3.3-13. Ejemplo de configuración del método `applicable_formats()` restringiendo la aparición del bloque únicamente a la página frontal.

```
function applicable_formats() {
    return array('course-view' => true,
                 'course-view-social' => false);
}
```

Figura 3.3.3-14. Ejemplo de configuración del método `applicable_formats()` permitiendo la aparición del bloque sólo en los cursos en cualquier formato excepto social.

```
function applicable_formats() {
    return array('site-index' => true,
                 'course-view' => true,
                 'course-view-social' =>false,
                 'mod' => true,
                 'mod-quiz' => false);
}
```

Figura 3.3.3-15. Ejemplo de configuración del método `applicable_formats()` permitiendo la aparición del bloque en la página frontal, en los cursos (excepto social) y en culalquier módulo de actividad excepto *quiz*).

17. Por último, podemos crear bloques en forma de lista en lugar del estándar bloque de texto. Para ello deberemos extender la clase `block_list` en lugar de la clase `block_base`.

```
class block_my_menu extends block_list {
     // El método init() no necesita cambiar para nada
}
```

Figura 3.3.3-16. Ejemplo de definición de clase lista.

Será necesario además modificar el método `get_content()` para añadir la lista y los iconos asociados a los elementos de la misma.

```
function get_content() {
    if ($this->content !== NULL) {
        return $this->content;
    }

    $this->content = new stdClass;
    $this->content->items = array();
    $this->content->icons = array();
    $this->content->footer = 'Pie de página aquí...';

    $this->content->items[] = '<a href="some_file.php">Menu Option
1</a>';
    $this->content->icons[] = '<img src="images/icons/1.gif"
width="16" height="16" alt="" />';

    // Agrega más ítems a la lista aquí

    return $this->content;
}
```

Figura 3.3.3-17. Ejemplo de configuración del método `get_content()` para un bloque tipo.

3.3.4. Especificación de temas

Los temas definen la apariencia de un sitio. Estos se almacenan dentro del directorio `/theme/` y ha de contener como mínimo los siguientes archivos:

- `config.php`. Define los colores del tema que se usan en todo el sitio.
- `styles.php`. Contiene las definiciones de la hoja de estilos CSS.
- `header.html`. Cabecera del sito. Debería ser incluido al principio de cada página.
- `footer.html`. Pie de cada página.

3.3.5. Especificación de esquemas de base de datos

Existe un problema en la creación automática de nuevas tablas en una base de datos en la instalación inicial. Actualmente sólo están soportados de forma total MySQL y PostgreSQl aunque es posible definir nuevos esquemas en los archivos del directorio `lib/db` y dentro del subdirectorio `db/` de cada módulo (si existe).

4. Implementación del foro estructurado

4.1. Revisión de la especificación y diseño

La especificación del foro estructurado se basa en un estudio previo. Nos encontramos por un lado que dicha especificación está fuertemente ligada a una plataforma genérica implementada en el lenguaje orientado a objetos JAVA a través de la librería *Collaborative Learning Purpose Library* (CLPL). Por otro lado, *Moodle* está implementado a través del lenguaje programación interpretado PHP, usado generalmente para la creación de contenido para sitios Web. Esta diferencia de tecnología, plataforma y paradigma de programación obliga a tener que realizar importantes cambios de adaptación al entorno.

En primer lugar, el foro estructurado encaja en la plataforma *Moodle* como un módulo de actividad para un curso no presencial. La especificación está orientada a un entorno centralizado desde el cual el tutor gestiona los espacios de trabajo, grupos y usuarios/estudiantes de todos los cursos. Esta centralización no sirve en la plataforma ya que los usuarios son gestionados desde la página principal de cada curso y no desde los módulos de actividad. Hay que tener en cuenta además que *Moodle* permite para un mismo curso tener varias instancias de un mismo módulo de actividad con configuraciones diferentes. Por estas razones y con el fin de respetar en la medida de lo posible la especificación se ha decidido restringir la creación de instancias a una por curso, sin embargo la gestión de usuarios se deberá seguir gestionando desde la página principal del curso ya que sino entraría en conflicto con el resto de la plataforma. Por lo tanto, la gestión de usuarios del foro propuesta se restringe únicamente a la asignación de los mismos a grupos, **no se permite la creación, modificación y/o eliminación de usuarios desde otro sitio diferente de la página principal del curso**.

En segundo lugar, y siguiendo la filosofía comentada en el párrafo anterior, el registro y acceso de los usuarios al foro deberá llamar al sistema de acceso de la plataforma. Esta modificación supone un cambio importante en el funcionamiento del subsistema de seguridad del foro (*Security subsystem*) propuesto en la especificación. Por lo tanto, el acceso de los usuarios al sistema se llevará a cabo desde la página del sitio o del curso al que quieren acceder a través del sistema de seguridad de la plataforma.

En tercer lugar, es necesario destacar que aunque la especificación contempla que un usuario puede pertenecer a varios grupos, el modelo de base de datos propuesto no tiene en cuenta está relación. En la implementación sí se ha tenido en cuenta y por lo tanto un usuario puede pertenecer a varios grupos sean o no de un mismo curso.

Figura 4.1-1. Acceso de un usuario al sistema.

Por último, siguiendo la nomenclatura del apartado *Normas y convenios* de la sección 3 se han tenido que realizar cambios en los nombres de las clases, funciones, tablas y campos de las mismas para adaptarlos al entorno.

4.1.1. Nomenclatura

La aplicación de las normas y convenios obliga a tener que realizar cambios en varios contextos:

- **Base de datos:**

1. El identificador de cada tabla ha de ser un campo auto numérico llamado *id* (INT10 – entero de longitud 10). Deberemos modificar todos los identificadores principales de todas las tablas del módulo para cumplir esta norma ya que el formato propuesto *dftabla-code* no se ajusta a la norma.

2. Los módulos de actividad han de tener como mínimo una tabla principal en la que almacenar las diferentes instancias; esta tabla ha de tener el mismo nombre que el módulo, en nuestro caso tendremos que añadir la tabla `structuredforum`.

3. El resto de las tablas asociadas han de tener como nombre el prefijo del módulo al cual pertenece más el propio nombre de la tabla. En nuestro caso modificaremos todos los nombres de tablas de la especificación para que contengan el prefijo `structuredforum_` en lugar del acrónimo DF.

4. Todo campo que contenga una referencia al campo identificador de otra tabla (por ejemplo, *modulo*) han de ser llamado con el nombre de la tabla referencia (por ejemplo, *moduloid*).

5. Se recomienda que todas las tablas contengan un campo *timemodified* para almacenar la fecha y hora de la última actualización de un registro. No añadiremos este campo en todas ya que no es necesario, sólo lo haremos en aquellas tablas con campos de fecha del modelo ER propuesto.

Durante la instalación de la plataforma se puede indicar opcionalmente un prefijo a las tablas (además del prefijo del propio módulo). No es necesario que tengamos en cuenta este prefijo a la hora de referenciarnos a dichas tablas ya que el sistema lo detectará automáticamente. Como anécdota comentar que durante la instalación del entorno de desarrollo hemos escogido el prefijo `mdl_` para las tablas de la plataforma, por lo tanto físicamente las tablas del módulo tienen el prefijo `mdl_structuredforum_` aunque esto no afecta a futuras instalaciones del mismo.

- **Funciones, clases y objetos:**

Siguiendo el convenio establecido para las funciones y nombres de variables, deberemos modificar el acrónimo DF del nombre de las clases por el prefijo *structuredforum_* y los campos *identifier* por *id* para hacer referencia al identificador así como los nombres de los campos que hagan referencia al identificador de otra tabla. Dentro de una clase este prefijo no afectará a las funcionen que la componen.

4.1.2. Interfaz

Moodle utiliza temas que permiten personalizar el diseño de toda la plataforma. No sería recomendable utilizar un diseño específico para un módulo de actividad ya que crearía inconsistencia con el resto del sitio.

En la implementación del foro no se ha seguido la interfaz propuesta en el prototipo ni en el resto del estudio. En su lugar se utiliza la hoja de estilos estándar que variará en función del tema; éste puede ser escogido por el administrador del sitio desde el bloque administración→Configuración→Temas en la página principal de la plataforma. Aún así, se ha intentado respetar en la medida de lo posible la ubicación de los elementos y la lista de opciones de los menús. Para la elaboración de las figuras del presente documento se ha escogido el tema “formal_white” tal y como muestra la *figura 4.1.2-1*.

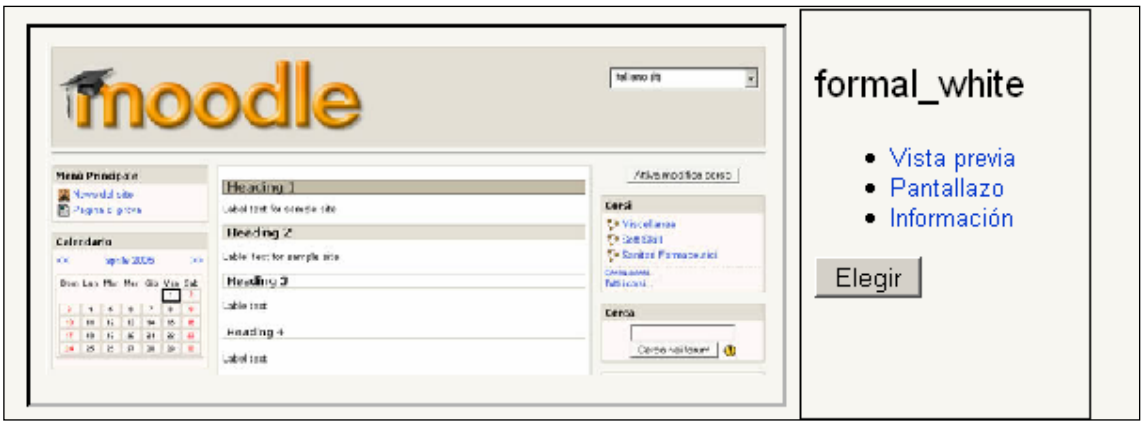


Figura 4.1.2-1. Vista previa del tema `formal_white`.

Según el convenio se ha de añadir una ruta de navegación que permita acceder de forma sencilla a los diferentes cursos, módulos de actividad y secciones (Curso >> Módulo >> Instancia>> Sección).

Todas los textos mostrados en la interfaz han sido tipificados para español internacional (es) a través del fichero de idioma `structucturedforum.php` (dentro del directorio `lang/es/`) y de las ventanas de ayuda para los campos de formulario (dentro del directorio `lang/es/help/structuredforum/`). Es posible traducir el módulo a otros idiomas definiendo el fichero de idioma y las ventanas de ayuda en el directorio correspondiente. El idioma a mostrar depende del perfil del usuario aunque puede forzarse en la definición del curso.

4.1.3. Modelo de base de datos

Debido a la adaptabilidad al nuevo entrono el modelo de base de datos ha sufrido importantes cambios:

1. Se han realizado los cambios de nomenclatura comentados.

2. La tabla `DFUser` se ha substituido por la tabla `USER` que es el espacio donde Moodle almacena los datos de todos los usuarios del sitio. Esta tabla se relaciona con las tablas `user_students` y `user_teachers` que definen el rol del usuario en el curso (estudiante, profesor).

3. Se han eliminado las tablas `DFAgenda` y `DFCalendar`, almacenando todos los datos del organizador en la tabla `structuredforum_organizer`. Se ha tomado esta decisión por

eficiencia ya que una agenda y un calendario son conceptos muy similares y la separación de estos sólo tenía sentido cuando estaba ligado a la librería CLPL.

4. Se han añadido las tablas `structuredforum_explorer` y `structuredforum_document` para la gestión de documentos. Estas tablas deberían aparecer en el modelo propuesto que no es del todo completo al estar orientado a la elaboración del prototipo.

5. Había un error en el modelo propuesto referente a la asignación de usuarios a grupos. La relación debería ser N-M en lugar de 1-N para permitir que un usuario pueda pertenecer a varios grupos. Se ha añadido la tabla `structuredforum_usergroup` que tiene en cuenta esta nueva relación.

6. Por eficiencia se ha decidido eliminar las tablas `DFChart`, `DFlag`, `DFStadistics` y `DFKnowledge` ya que tanto las estadísticas como el conocimiento se pueden calcular a través de las tablas `structuredforum_logfile` y `structuredforum_interaction`.

Actualmente Moodle soporta los SGBD *MySQL* y *PostgreSQL*. Para la implementación del foro se ha definido el esquema de base de datos en MySQL utilizado integridad referencial a través de las tablas INNODB. De esta manera nos aseguramos que los registros de tablas relacionadas son válidos y que no se borren o cambien datos relacionados de forma accidental produciendo errores de integridad.

A continuación se muestra el diagrama entidad-relación (ER) así como la descripción de los atributos de las tablas que lo componen:

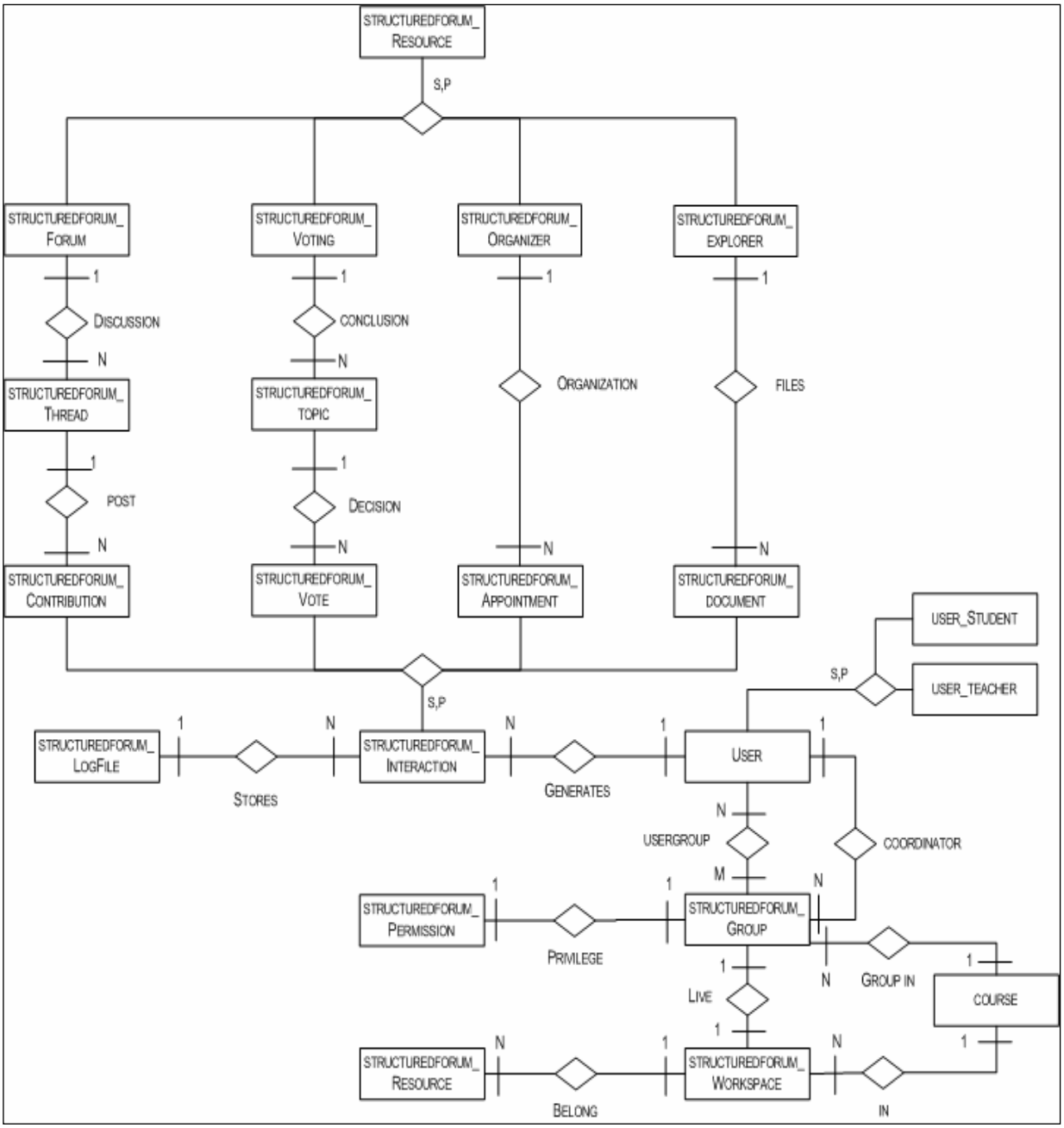


Figura 4.1.3-1. Modelo entidad-relación (DER).

STRUCTUREDFORUM_RESOURCE

id, name, type, workspaceid

donde {workspaceid} referencia a STRUCTUREDFORUM_WORKSPACE

STRUCTUREDFORUM_FORUM

id, workspaceid, moderatorid, threads, contributions

donde {workspaceid} referencia a STRUCTUREDFORUM_WORKSPACE

STRUCTUREDFORUM_VOTING

id, workspaceid, topics, votes

donde {workspaceid} referencia a STRUCTUREDFORUM_WORKSPACE

STRUCTUREDFORUM_ORGANIZER

id, workspaceid, appointments

donde {workspaceid} referencia a STRUCTUREDFORUM_WORKSPACE

STRUCTUREDFORUM_EXPLORER

id, workspaceid, documents

donde {workspaceid} referencia a STRUCTUREDFORUM_WORKSPACE

STRUCTUREDFORUM_THREAD

id, name, timemodified, forumid, firstid, lastid, contributions

donde {forumid} referencia a STRUCTUREDFORUM_FORUM

donde {firstid} referencia a STRUCTUREDFORUM_CONTRIBUTION

donde {lastid} referencia a STRUCTUREDFORUM_CONTRIBUTION

STRUCTUREDFORUM_TOPIC

id, name, description, votingid

donde {votingid} referencia a STRUCTUREDFORUM_VOTING

STRUCTUREDFORUM_CONTRIBUTION

id, name, category, participantid, timemodified, content, referenceid, threadid, attachment

donde {participantid} referencia a USER

donde {referenceid} referencia a STRUCTUREDFORUM_CONTRIBUTION

donde {threadid} referencia a STRUCTUREDFORUM_THREAD

STRUCTUREDFORUM_VOTE

id, participantid, timemodified, content, topicid

donde {participantid} referencia a USER

donde {topicid} referencia a STRUCTUREDFORUM_TOPIC

STRUCTUREDFORUM_APPOINTMENT

id, participantid, timemodified, content, organizerid, daymodified

donde {participantid} referencia a USER

donde {organizerid} referencia a STRUCTUREDFORUM_ORGANIZER

STRUCTUREDFORUM_DOCUMENT

id, description, size, explorerid, timemodified, attachment, participantid

donde {participantid} referencia a USER

donde {explorerid} referencia a STRUCTUREDFORUM_EXPLORER

STRUCTUREDFORUM_LOGFILE

id, timemodified, workspaceid

donde {workspaceid} referencia a STRUCTUREDFORUM_WORKSPACE

STRUCTUREDFORUM_INTERACTION

id, senderid, receiverid, content, channel, content, referid, logfileid

donde {senderid} referencia a USER

donde {referid} referencia a STRUCTUREDFORUM_APPOINTMENT or VOTE or DOCUMENT or CONTRIBUTION (depende del campo channel)

donde {receiverid} referencia a STRUCTUREDFORUM_GROUP

donde {logfileid} referencia a STRUCTUREDFORUM_LOGFILE

STRUCTUREDFORUM_PERMISSION

id, groupid, pread, pwrite, pdelete, pmove

donde {groupid} referencia a STRUCTUREDFORUM_GROUP

STRUCTUREDFORUM_GROUP

id, name, coordinatorid, workspaceid, courseid

donde {coordinatorid} referencia a USER

donde {workspaceid} referencia a STRUCTUREDFORUM_WORKSPACE

donde {courseid} referencia a COURSE

STRUCTUREDFORUM_USERGROUP

id, userid, groupid

donde {userid} referencia a USER

donde {groupid} referencia a STRUCTUREDFORUM_GROUP

STRUCTUREDFORUM_WORKSPACE

id, name, courseid, groupid, description, generic

donde {courseid} referencia a COURSE

donde {groupid} referencia a STRUCTUREDFORUM_GROUP

4.1.4. Clases y objetos

La especificación está pensada en el paradigma de la orientación a objetos. PHP permite trabajar con clases y objetos a partir de la versión 5 (la versión 4 no permitía aspectos tan básicos como la herencia de clases) aunque todavía existen ciertas limitaciones en su uso (por ejemplo, todavía no es posible utilizar packages). Se ha intentando respetar en la medida de lo posible la definición de clases y objetos aunque se han tenido que realizar importantes cambios.

1. En primer lugar, la especificación del foro está dividida en cinco subsistemas:

   - Subsistema de seguridad (*Forum Security Subsystem*).
   - Subsistema de colaboración (*Forum Collaboration Subsystem*).
   - Subsistema de comunicación (*Forum Communication Subsystem*).
   - Subsistema de gestión (*Forum Management Subsystem*).
   - Subsistema de conocimiento (*Forum Knowledge Subsystem).*

   Cada uno de estos subsistemas está definido en un package diferente. Como no existen packages en PHP se ha salvado la situación creando un conjunto de librerías de clases que permiten simular dentro de lo posible este comportamiento. Estas librerías están almacenadas en los ficheros: `securitylib.php`, `collaboration.php`, `communicationlib.php`, `managementlib.php` y `knowledgelib.php` respectivamente.

   Además de estos subsistemas ha sido necesario la creación de un sistema genérico que describa el módulo de actividad dentro de la plataforma así como las clases genéricas (como por ejemplo las clases abstractas `structuredforum_generic_disk` y `structuredforum_generic_management`). Este sistema genérico está almacenado en la librería `lib.php`.

2. En segundo lugar, ha habido que modificar todas las referencias a la librería CLPL. En la mayoría de estas referencias nos encontramos que las clases extendían la funcionalidad de alguna de las clases genéricas de la librería. El cambio ha consistido principalmente en definir las clases del foro como principales, añadiendo sólo los métodos y atributos necesarios; es decir, no han sido añadido todos los métodos de clase de manera indiscriminada sino sólo aquellos que han sido necesarios.

3. En tercer lugar, para mejorar la legibilidad y la abstracción de datos desde el programa principal los constructores de las clases persistentes recibirán en la mayoría de los casos un único objeto (normalmente llamado `$form`) que incluirá todos los datos recibidos desde un formulario HTML. Este objeto se crea a partir de la función `data_submitted()` (perteneciente a la API de la plataforma) tal y como se muestra a continuación:

```
// si se ha enviado datos desde un formulario los almacena
if ($form = data_submitted()) {
      …
//sino muestra el formulario
}else{
      …
}
```

Figura 4.1.4-1. Recuperar datos enviados desde un formulario.

Por ejemplo, para la gestión de espacios de trabajo crearemos una instancia de la clase `structuredforum_workspace` con el siguiente constructor:

```
structuredforum_workspace ($form);
```

En lugar del constructor propuesto en la especificación:

```
structuredforum_workspace ($pidentifier, $pname, $plocation);
```

El constructor tratará únicamente aquellos atributos del objeto `$form` (`$form->nombre_atributo`) que tengan sentido para la instancia. En el ejemplo que nos ocupa tenemos que crear una instancia de la clase persistente de un espacio de trabajo. El código interno del constructor (incluido dentro de la clase `structuredforum_workspace` perteneciente a la librería `managementlib.php`) es el siguiente:

```
/**
* crea una nueva instancia para un espacio de trabajo
* @param Object $form
* @access public
*/
```

```
    public function structuredforum_workspace($form='') {
      $this->id=$form->id;
      $this->name=$form->name;
      $this->courseid=$form->courseid;
      $this->description=$form->description;
      $this->groupid=$form->groupid;
      $this->generic=$form->generic;
    }
```

Figura 4.1.4-2. Contenido del constructor `structuredforum_workspace()`.

En el ejemplo puede observarse varias diferencias respecto al constructor original. Estas diferencias hacen referencia a la lista de atributos de la clase. En el constructor de la clase original tenemos sólo tres atributos (`$pidentifier`, `$pname`, `$plocation`) y en la implementada seis (`$name`, `$courseid`, `$description`, `$groupid`, `$generic`) que además no se corresponden. Esto es debido a que **en todas las clases persistentes implementadas el identificador** (`$id`) **es autogenerado**; el resto de atributos de la clase se han añadido por correspondencia con los campos de la tabla persistente.

4. En cuarto lugar, tal y como se ha comentado en la revisión del modelo de base de datos (sección 4.1.3), se ha decidido por eficiencia eliminar las tablas `DFChart`, `DFlag`, `DFStadistics` y `DFKnowledge` ya que tanto las estadísticas como el conocimiento se pueden calcular a través de las tablas `structuredforum_logfile` y `structuredforum_interaction`. Esta acción supone un cambio importante en el subsistema de conocimiento (*Knowledge subsystem*) ya que se han suprimido las clases persistentes y lógicas relacionadas con estas tablas.

5. En quinto lugar se han unido en una sola clase aquellas que hacen referencia a la creación (*`Creation`*), consulta (*`Consultation`*), modificación (*`Modification`*), eliminación (*`Elimination`*) y excepción (*`Exception`*) de un objeto. Estas operaciones las llevará a cabo el gestor de disco.

6. Por ultimo, en la sección *4.2 Implementación* se describe el listado completo de clases en relación a la capa a la cual pertenecen.

Resumiendo, el lector que compare la descripción de clases del estudio con la implementación encontrará notables diferencias que se corresponden principalmente con el cambio de nomenclatura y la separación de la dependencia con la librería CLPL. Adjunto a esta documentación se aporta una API donde se definen todas las clases utilizadas organizadas por packages, donde se incluye además la descripción de todos los atributos y métodos de las mismas. Se ha decidido no incluir el contenido de esta API en el presente documento ya que la mayoría de esta información ya aparece en la especificación realizada en el estudio previo. Para la realización de la API se ha seguido el formato de documentación *PHPDoc* y el programa *PHPDocumentor* para la generación de la misma.

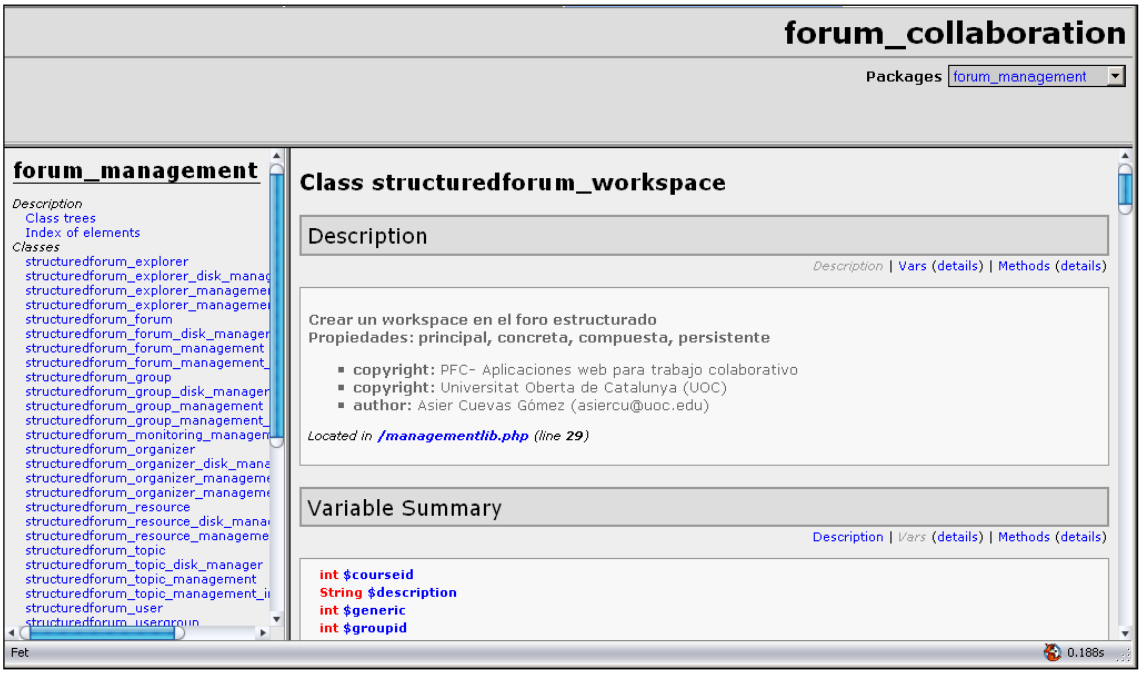


Figura 4.1.4-3. Ejemplo de descripción de la clase `structuredforum_workspace` desde la API.

4.2. Implementación

Para la implementación se ha definido una arquitectura basada en capas: persistencia, lógica y presentación. De esta manera se consigue mayor claridad y control sobre cada funcionalidad. La comunicación entre capas es bidireccional de tal manera que cuando llega una petición a una capa desde otra superior se realiza la operación demandada y acto seguido se devuelve el resultado a la capa que la había solicitado.

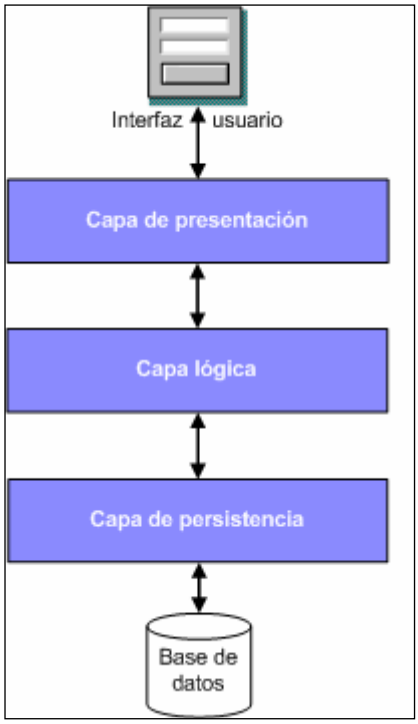


Figura 4.2-1. Arquitectura del foro estructurado.

4.2.1. Capa de persistencia

La gestión de persistencia con la base de datos se realiza a través de la clase abstracta `structuredforum_generic_disk` que actúa como gestor de disco. Esta clase es heredada por cada uno de los gestores de disco de las clases persistentes y gestiona todas las operaciones en disco sobre la correspondiente tabla. A continuación se muestra la lista completa de clases que gestionan todas las operaciones de disco (altas, bajas y modificaciones sobre las tablas de la base de datos, exceptuando la tabla USER que sólo es consultada ya que la gestión de usuarios y perfil se realiza desde el bloque administración de Moodle).

Clase	Descripción
structuredforum_thread_disk_manager	Clase puente para los datos persistentes de la tabla structuredforum_thread (tema o hilo de conversación).
structuredforum_contribution_disk_manager	Clase puente para los datos persistentes de la tabla structuredforum_contribution (mensaje).
structuredforum_vote_disk_manager	Clase puente para los datos persistentes de la tabla structuredforum_vote (voto).
structuredforum_appointment_disk_manager	Clase puente para los datos persistentes de la tabla structuredforum_appointment (cita).
structuredforum_document_disk_manager	Clase puente para los datos persistentes de la tabla structuredforum_document (documento).
structuredforum_interaction_disk_manager	Clase puente para los datos persistentes de la tabla structuredforum_interaction (interacción).
structuredforum_logfile_disk_manager	Clase puente para los datos persistentes de la tabla structuredforum_logfile (registro).
structuredforum_workspace_disk_manager	Clase puente para los datos persistentes de la tabla structuredforum_workspace (espacio de trabajo).
structuredforum_explorer_disk_manager	Clase puente para los datos persistentes de la tabla structuredforum_explorer (explorador de archivos).
structuredforum_forum_disk_manager	Clase puente para los datos persistentes de la tabla structuredforum_forum (foro de debate o de formación de grupo).
structuredforum_voting_disk_manager	Clase puente para los datos persistentes de la tabla structuredforum_voting (sistema de votación).
structuredforum_topic_disk_manager	Clase puente para los datos persistentes de la tabla structuredforum_topic (conclusión).
structuredforum_resource_disk_manager	Clase puente para los datos persistentes de la tabla structuredforum_resource (recurso).
structuredforum_group_disk_manager	Clase puente para los datos persistentes de la tabla structuredforum_group (grupo de estudiantes).

structuredforum_user_disk_manager	Clase puente para los datos persistentes de la tabla structuredforum_user (usuario).
structuredforum_organizer_disk_manager	Clase puente para los datos persistentes de la tabla structuredforum_organizer (organizador).
structuredforum_usergroup_disk_manager	Clase puente para los datos persistentes de la tabla structuredforum_usergroup (relación usuario-grupo).
structuredforum_permission_disk_manager	Clase puente para los datos persistentes de la tabla structuredforum_permission (permiso de grupo).

Estos gestores de disco actúan como clase puente entre los datos persistentes de la tabla a la cual hacen referencia y a la clase persistente correspondiente. Las clases persistentes son las siguientes.

Clase	**Descripción**
structuredforum_thread	Clase persistente de la tabla structuredforum_thread (tema o hilo de conversación).
structuredforum_contribution	Clase persistente de la tabla structuredforum_contribution (mensaje).
structuredforum_vote	Clase persistente de la tabla structuredforum_vote (voto).
structuredforum_appointment	Clase persistente de la tabla structuredforum_appointment (cita).
structuredforum_document	Clase persistente de la tabla structuredforum_document (documento).
structuredforum_interaction	Clase persistente de la tabla structuredforum_interaction (interacción).
structuredforum_logfile	Clase persistente de la tabla structuredforum_logfile (registro).
structuredforum_workspace	Clase persistente de la tabla structuredforum_workspace (espacio de trabajo).
structuredforum_explorer	Clase persistente de la tabla structuredforum_explorer (explorador de archivos).

structuredforum_forum	Clase persistente de la tabla structuredforum_forum (foro de debate o de formación de grupo).
structuredforum_voting	Clase persistente de la tabla structuredforum_voting (sistema de votación).
structuredforum_topic	Clase persistente de la tabla structuredforum_topic (conclusión).
structuredforum_resource	Clase persistente de la tabla structuredforum_resource (recurso).
structuredforum_group	Clase persistente de la tabla structuredforum_group (grupo de estudiantes).
structuredforum_user	Clase persistente de la tabla structuredforum_user (usuario).
structuredforum_organizer	Clase persistente de la tabla structuredforum_organizer (organizador).
structuredforum_usergroup	Clase persistente de la tabla structuredforum_usergroup (relación usuario-grupo).
structuredforum_permission	Clase persistente de la tabla structuredforum_permission (permiso de grupo).

4.2.2. Capa lógica

La lógica de gestión de las diferentes secciones se realiza a través de la clase abstracta `structuredforum_generic_management`. Esta clase es heredada por cada uno de los gestores de las clases persistentes y gestiona todas las operaciones lógicas que se realizan con la clase persistente; es decir, se encargan de crear las instancias de los objetos, acceder a los métodos de los mismos y de realizar las llamadas al gestor de disco (capa de persistencia). A continuación se muestra la lista completa de clases encargadas de la gestión de los objetos persistentes.

Clase	Descripcion
structuredforum_thread_management	Abstrae parte de la lógica de la gestión de threads (tema o hilo de conversación).
structuredforum_contribution_management	Abstrae parte de la lógica de la gestión de contribution (mensaje).
structuredforum_vote_management	Abstrae parte de la lógica de la gestión de vote (voto).
structuredforum_appointment_management	Abstrae parte de la lógica de la gestión de appointment (cita).
structuredforum_document_management	Abstrae parte de la lógica de la gestión de document (documento).
structuredforum_interaction_management structuredforum_logfile_management	Abstrae parte de la lógica de la gestión de interaction (interacción). Abstrae parte de la lógica de la gestión de logfile (registro).
structuredforum_workspace_management	Abstrae parte de la lógica de la gestión de workspace (espacio de trabajo).
structuredforum_explorer_management	Abstrae parte de la lógica de la gestión de explorer (explorador de archivos)
structuredforum_resource_management	Abstrae parte de la lógica de la gestión de resource (recurso).
structuredforum_group_management	Abstrae parte de la lógica de la gestión de group (grupos de usuarios).
structuredforum_user_management	Abstrae parte de la lógica de la gestión de user (usuario).
structuredforum_usergroup_management	Abstrae parte de la lógica de la gestión de usergroup (relación usuario-grupo).
structuredforum_forum_management	Abstrae parte de la lógica de la gestión de forum (foro de debate y de formación de grupo),
structuredforum_voting_management	Abstrae parte de la lógica de la gestión de voting (sistema de votación).
structuredforum_topic_management	Abstrae parte de la lógica de la gestión de topic (conclusión).
structuredforum_organizer_management	Abstrae parte de la lógica de la gestión de organizer (organizador).

structuredforum_permission_management	Abstrae parte de la lógica de la gestión de permission (permisos de grupo).

4.2.3. Capa de presentación

En esta última se encarga de presentar al usuario el resultado final de las operaciones de la capa lógica. Esta presentación es gestionada a través de la clase `structuredforum_interface` y de las clases que la heredan. Éstas obtienen el resultado de la lógica de gestión y los transfieren a los documentos finales que verá el usuario. Estos documentos contienen código HTML, *JavaScript* y un reducido número de etiquetas PHP para imprimir los datos obtenidos. A continuación se muestra la lista completa de clases que definen la interfaz del usuario (sobretodo la distribución de los elementos ya que el aspecto gráfico es gestionado por el tema del sitio).

Clase	**Descripción**
structuredforum_workspace_management_interface	Interficie de la gestión de espacios de trabajo.
structuredforum_monitoring_management_interface	Interficie del seguimiento de grupos.
structuredforum_group_management_interface	Interficie de la gestión de grupos de usuarios.
structuredforum_topic_management_interface	Interficie de la gestión de conclusiones.
structuredforum_user_management_interface	Interficie de la gestión de usuarios (sólo para consultas y relación con grupos).
structuredforum_forum_management_interface	Interficie del foro de debate y de formación de grupo.
structuredforum_explorer_management_interface	Interficie del explorador de archivos.
structuredforum_voting_management_interface	Interficie del sistema de votación. Permite gestionar las conclusiones y votar sobre ellas.
structuredforum_organizer_management_interface	Interficie del organizador. Permite gestionar las citas.

Por último, comentar que la gráfica de utilización del foro de debate de la sección de seguimiento (`monitoring`) se crea y actualiza dinámicamente gracias a la utilización de la librería genérica *PHPLot* (librería creada en PHP y de libre distribución para la generación dinámica de gráficas).

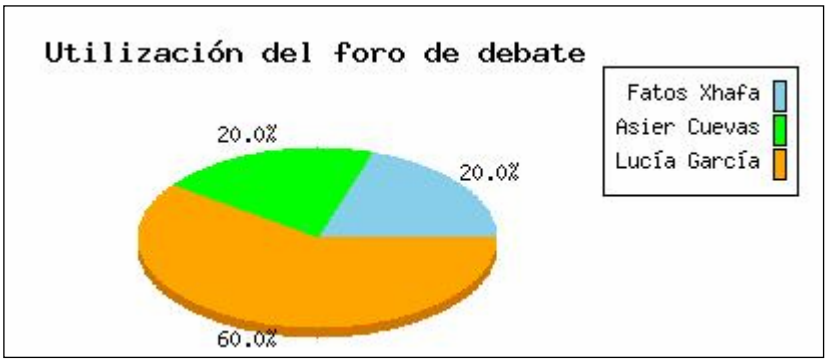


Figura 4.2.3-1. Gráfica de utilización del foro a través de la librería *PHPLot*.

4.2.4. Estructura de directorios

El foro estructurado es un módulo de actividad que deberá almacenarse dentro del directorio `/mod/` de la instalación de Moodle. Este módulo contiene la estructura de directorios definida en la figura 3.4.1

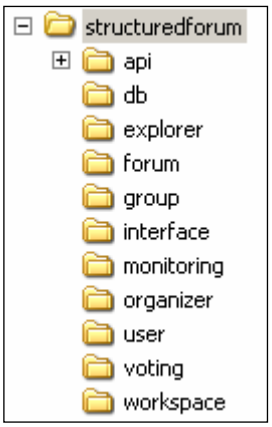


Figura 4.2.4-1. Estructura de directorios del módulo.

- `api/`. API donde se muestra la descripción de las clases con sus métodos y atributos.

- `db/`. Define el esquema de base de datos. Contiene los ficheros `mysql.php` y `mysql.sql` donde se definen las sentencias SQL para el SGBD MySQL que se ejecutarán durante la instalación del módulo.

- `explorer/`. Explorador de archivos. Contiene el programa principal (`index.php`) y los documentos HTML asociados para representar los botones, listado de documentos y ficha de documento (`buttons.html`, `list.html` y `modify.html`).

- `forum/`. Foro de debate o de formación de grupo. Contiene el programa principal (`index.php`) y los documentos HTML asociados para representar los botones, listado de temas, paginación y ficha de mensaje (`buttons.html`, `list.html`, `pages.html` y `modifypost.html`).

- `group/`. Grupo de usuarios. Contiene el programa principal (`index.php`) y los documentos HTML asociados para la ficha del grupo (`modify.html`).

- `interface/`. Interficie de usuario. Contiene los documentos HTML genéricos para el área de profesor (`tutorarea.html`), coordinador (`coordinatorarea.html`) y estudiante (`studentarea.html`) así como la página de confirmación para eliminar objetos (`confirm.html`) y la de consulta (`consult.html`).

- `monitoring/`. Seguimiento de la actividad de los grupos. Contiene el programa principal (`index.php`) y los documentos HTML asociados para representar el informe de seguimiento (`report.html`).

- `organizer/`. Organizador. Contiene el programa principal (`index.php`) y los documentos HTML asociados para representar los datos resumidos de una cita, cabecera y pie de organizador y ficha de cita (`appointment.html`, `header.html`, `footer.html` y `modify.html`).

- `user/`. Usuarios del curso. Contiene el programa principal (`index.php`) y los documentos HTML asociados para representar la ficha de usuario (`modify.html`). En la ficha de usuario se muestra los grupos de los que es miembro para el curso seleccionado, el perfil de usuario se muestra a través de la página `/user/view.php` (no confundir el directorio de usuario del foro `/mod/structuredforum/user/` con el de Moodle `/user`) a la cual deberemos pasar como parámetro el identificador del usuario (`id`) y el identificador del curso (`course`), por ejemplo `/user/view.php?id=3&course=8`.

- `voting/`. Sistema de votación. Contiene el programa principal (`index.php`) y los documentos HTML asociados para representar el listado de conclusiones (con su respectivo voto si este se ha llevado a cabo) y la ficha de conclusión (`list.html` y `modify.html`).

- `workspace/`. Espacios de trabajo. Contiene el programa principal (`index.php`) y los documentos HTML asociados para representar la ficha de espacio de trabajo (`modify.html`).

Por otro lado, es necesario disponer de una estructura de directorios donde almacenar los documentos del explorador de archivos y de los adjuntos del foro de debate y de formación de grupo.

Moodle almacena los ficheros de datos en el directorio `/moodledata/` (en nuestro caso `C:\AppServ\www\moodledata`). Este directorio no debería ser accesible desde el exterior, únicamente desde el código de los módulos. Para no mezclar datos este directorio está organizado a través de los identificadores de los cursos. Dentro de un curso pueden haber ficheros del programa principal y de los módulos; estos últimos se almacenan dentro del directorio `moddata/`. Cada módulo tendrá a su vez su propia estructura de directorios donde almacenar los datos; en nuestro caso al añadir un foro estructurado a un curso, se creará automáticamente un directorio llamado structuredforum/ y dentro de este directorio se irán creando directorios para los espacios de trabajo que vayamos definiendo.

En la siguiente figura podemos observar la estructura de directorios de datos para un curso con identificador 6 en el cual hay definidos dos espacios de trabajo (identificador 10 y 11 respectivamente). El espacio 11 tiene a su vez definido el recurso foro de debate (directorio `forum/`) en el cual existe una contribución (identificador 32) que contiene un fichero adjunto (si la contribución no contiene adjuntos no se crea el directorio). El sistema es muy parecido para el explorador de archivos y los documentos que lo componen.

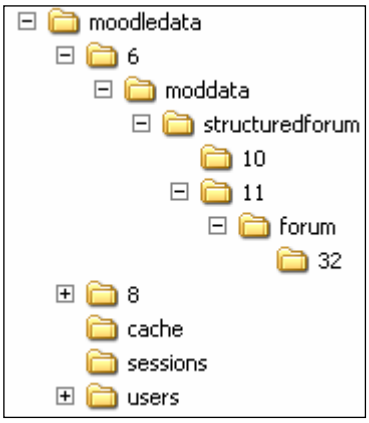


Figura 4.2.4-2. Estructura de directorios de datos.

5. Conclusiones

La selección de un entrono de aprendizaje para trabajo colaborativo no es una tarea sencilla ya que existen multitud de productos que cumplen este propósito. Los sistemas de libre distribución parecen ser la opción más interesante gracias a su bajo coste (sólo tenemos que preocuparnos del hardware, no de las licencias) y a las posibilidades que ofrecen (éstas son básicamente las mismas que las ofrecidas por los productos comerciales).

De entre todos los sistemas de libre distribución existentes *Moodle* se adapta mejor al trabajo colaborativo en grupo en base al modelo pedagógico constructivista social en el cual está basado. Los estudiantes aprenden de ellos mismos y de la interacción de otros participantes compartiendo, construyendo nuevas ideas. *Moodle* es un sistema de libe distribución modular y totalmente abierto. Esto permite poder añadir nuevos contenidos que motiven al estudiante, ya sea a través de los módulos de actividades, los bloques e incluso los temas.

Sin embargo, el sistema presenta alguna desventaja con respecto a otros productos ya que quizás no sea la opción más adecuada para entornos basados en el auto-estudio en los cuales se utilice los elementos de colaboración y comunicación únicamente como apoyo al proceso de aprendizaje. Por otro lado, hay que comentar que la realización de las actividades puede resultar a veces demasiado mecánica.

En la segunda parte de este proyecto hemos implementado un foro estructurado basado en tres tipos de contribuciones o etapas (especificación, elaboración y consenso). Este foro ha sido integrado en la plataforma *Moodle* como módulo de actividad. Para ello hemos partido de un trabajo previo sobre la especificación y diseño del mismo.

La dependencia de la especificación del foro estructurado con la librería CLPL ha obligado a tener que realizar importantes cambios de adaptación al entorno, sin embargo los cambios más importantes han sido a causa de la propia especificación de *Moodle*. Esta especificación estaba pensada para un entorno centralizado sin embargo como módulo de actividad sólo tenía que ser posible gestionar los datos del propio módulo, dejando la gestión de usuarios y cursos para el módulo de administración de la plataforma.

La arquitectura de la aplicación basada en capas ha permitido separar la lógica de la presentación así como de la gestión de los datos persistentes de la base de datos. Así mismo la presentación se ha llevado a cabo utilizando hojas de estilos genéricas que son utilizadas por el tema de la interfaz, de esta manera el diseño del foro está perfectamente integrado con el resto de secciones del sitio y puede ser modificado de forma sencilla por el administrador.

Ha sido de importante ayuda la utilización de la librería *PHPLot* para la generación dinámica de la gráfica de utilización del foro de debate. Esta gráfica aporta información al profesor y/o al coordinador del grupo sobre el seguimiento de la actividad.

Toda la documentación del código ha sido realizada utilizando el formato *PHPDoc*. Junto a esta memoria se aporta la API generada con la definición de clases, métodos y atributos del foro estructurado.

Por último, comentar que no han surgido problemas importantes a la hora de realizar la instalación del entorno de desarrollo ni de la plataforma Moodle. Para ello se han seguido los pasos comentados en el manual del administrador (adjunto a esta memoria. Quizás sí habría que mencionar el volumen de trabajo que ha conllevado la implementación del foro debido sobretodo a los cambios que ha habido que realizar en la especificación motivados por la plataforma *Moodle*, aunque este volumen de trabajo no ha impedido la entrega del proyecto en las fechas señaladas en la planificación del mismo.

6. Glosario

A continuación de describe una recopilación de los términos más relevantes del presente estudio.

- **Apache**. Servidor Web de código abierto, modular y multiplataforma. En la actualidad es el servidor Web más utilizado.

- **API**. Interfaz de Programación de Aplicaciones. Suministra documentación sobre un conjunto de especificaciones de comunicación entre componentes.

- **AppServ**. Paquete software que nos permite instalar de forma fácil un entrono WAMP con el que poder ejecutar la plataforma.

- **DER**. Modelo entidad relación para el esquema de la base de datos.

- **Entorno LAMP**. Sistema operativo Linux, con servidor web Apache, SGBD MySQL y soporte para lenguaje PHP. Es el entorno más común para la instalación de *Moodle*.

- **Entorno WAMP**. Sistema operativo Microsoft Windows, con servidor web Apache, SGBD MySQL y soporte para lenguaje PHP.

- **Moodle**. Plataforma de libre distribución (bajo licencia GNU) que cubre la necesidad de disponer de una entorno de aprendizaje virtual, abierto y gratuito como soporte para el trabajo colaborativo en grupo.

- **MoodleCat**. Servicio tecnológico para la implantación de *Moodle* en las escuelas catalanas.

- **MySQL**. SGBD de código abierto y multiplataforma. Es el más popular y utilizado en plataformas web y en concreto también en *Moodle*.

- **PHP**. Lenguaje de programación interpretado de código abierto, de alto nivel, embebido en páginas HTML y ejecutado en el servidor.

- **PHPDoc**. Formato de documentación de libre distribución similar al *JavaDoc* empleado en entornos Java.

- **PostgreSQL**. SGBD alternativo también de código abierto y multiplataforma. Su uso es recomendable en entornos en los cuales se maneje gran cantidad de datos.

- **Principios pedagógicos constructivistas**. Es una filosofía de aprendizaje y una forma pensar que se basa en la experiencia para aprender desde el punto de vista de los estudiantes, en lugar de limitarse simplemente a proporcionarles la información que cree que necesitan saber.

- **Sistema gestor de base de datos (SGBD)**. Servidor que permite almacenar, acceder y actualizar información en bases de datos. Un SGBD puede gestionar muchas bases de datos a la vez.

- **SQL Injection**. Es un tipo de ataque que consiste en introducir sentencias SQL en campos de usuario con el fin de producir errores impredecibles en la aplicación. Ara evitarla se ha de llevar a cabo una correcta gestión de errores.

- **WebCT.** *Web Course Tools*, o Herramientas para Cursos Web) es un sistema comercial de aprendizaje virtual el cual es usado principalmente por instituciones educativas para el aprendizaje a través de Internet.

Bibliografía

A continuación se muestra una lista de las fuentes de información consultadas para la realización de este estudio:

[1] **Moodle**. Página Web oficial de la plataforma. http://*Moodle*.org/

[2] **MoodleCat**. Página de la comunidad *MoodleCat*. http://www.*Moodle*cat.com/

[3] **Cole, Jason (2005)**. *Using Moodle – Teaching with the Popular Open Source Course Management System*. O'Reilly

[4] **Caballé S. and Xhafa F. (2005)**. *A Web-Based Structured Discussion Forum for Collaborative Learning Environments*. Master's Final Project (PFC). Open University of Catalonia (UOC).

[5] **API principal de Moodle**. Describe las funciones públicas que pueden ser utilizadas desde los módulos y bloques de la plataforma. Esta API está en formato PHPDoc que es le mismo utilizado para la generación de la API del foro estructurado. http://moodle.sourceforge.net/dhawes-phpdoc/

[6] **Wikipedia**, definición de *Moodle*. http://es.wikipedia.org/wiki/*Moodle*

[7] **Apache**. Servidor Web. http://httpd.apache.org/

[8] **PHP.** Lenguaje de programación http://www.php.net

[9] **MySQL**. SGBD. http://www.mysql.com/

[10] **AppServ**. Paquete de aplicaciones para servidor Web. http://www.appservnetwork.com/

[11] **PHPDoc**. Paquete de libre distribución para generar documentación en PHP. http://www.phpdoc.de/

[12] **PHPLot**. Paquete de libre distribución para generar gráficas vectoriales. http://sourceforge.net/projects/phplot/

[13] **Securiteam**. Página Web sobre anuncios y recomendaciones de seguridad en la web. http://www.securiteam.com/securityreviews/5DP0N1P76E.html

[14] **Alternativas libres**. Página dedicada a la definición de alternativas software de libre distribución. http://alts.homelinux.net/index.php

[15] **Edutools.** Estudio de comparación de plataformas e-learning. http://www.edutools.info/item_list.jsp?pj=8