Universidad Zaragoza
1542

Proyecto Fin de Carrera

Puesta en marcha de plataformas multi-sitio tipo campus virtual mediante la extensión de Moodle

Autor

Eduardo Ramos Ibáñez

Directora

Lucía Fernández Marco

Ponente

Sergio Ilarri Artigas

Escuela de Ingeniería y Arquitectura (EINA)

2014

Gracias a mis compañeros de trabajo por su apoyo, consejos e ideas aportadas durante el desarrollo de este proyecto.

Gracias a mis profesores, que tanto me han enseñado a lo largo de los años.

Y en especial a mi familia y amigos por la larga espera para ver este proyecto finalizado.

PUESTA EN MARCHA DE PLATAFORMAS MULTI-SITIO TIPO CAMPUS VIRTUAL MEDIANTE LA EXTENSIÓN DE MOODLE

RESUMEN

Moodle es una plataforma de aprendizaje on-line de software libre que cuenta con un alto nivel de madurez, extensibilidad y utilización en todo tipo de entornos educativos como colegios, universidades y otros centros de enseñanza superior.

Este proyecto surge a partir de la necesidad de la empresa SEAS de ampliar sus posibilidades en el campo de formación online mediante algunos cursos que no se adaptan a su plataforma de aprendizaje ya desarrollada y en funcionamiento en la actualidad.

Dicha empresa trabaja en múltiples ámbitos de formación muy diferenciados, llamados líneas de negocio, y su plataforma está desarrollada desde la base para facilitar la separación de ellos. Estos ámbitos o contextos permiten, mediante el mantenimiento de una única plataforma, ofrecer una personalización del aspecto, funcionalidades y administración separada de cada línea de negocio como si se tratase de una plataforma exclusiva.

Sin embargo, Moodle no está diseñado para contener varios ámbitos en un solo sitio web, de forma que es necesario desplegar varios sitios Moodle, uno por línea de negocio. Pero la gestión y mantenimiento de varios sitios Moodle puede llegar a ser una tarea manual repetitiva, poco eficiente y propensa a errores, especialmente cuando la cantidad de sitios es elevada.

Con este proyecto pretendemos ofrecer una solución automatizada general para el despliegue y gestión de varias plataformas Moodle a modo de multi-sitio, así como ofrecer una serie de complementos de utilidad común en todos los sitios Moodle, formen o no parte de un multi-sitio.

De este modo, el proyecto se compone de dos módulos principales técnicamente independientes, la aplicación de gestión del multi-sitio y un tema para Moodle con características de personalización avanzadas.

Además, ha sido necesario desarrollar otros módulos adicionales durante la evolución del proyecto y de sus requisitos, tratándose principalmente de módulos dirigidos a la integración de Moodle con la plataforma de SEAS.

Tabla de contenidos

Índice de tablas

Índice de ilustraciones

1. Introducción

1.1 Objetivos, motivaciones y viabilidad del proyecto

Este proyecto se ha realizado como proyecto final de carrera para Ingeniería Informática en la escuela universitaria EINA y el título elegido es "Puesta en marcha de plataformas multi-sitio tipo campus virtual mediante la extensión de Moodle".

Se pretende publicar con una licencia de software libre (GNU GPL v3[1]) algunas de las partes de este proyecto. Al momento de terminar esta memoria han sido publicados los componentes de este proyecto relativos a la personalización de los sitios que forman el multi-sitio, ya que como veremos más adelante se tratan de las partes más reutilizables potencialmente por otras personas de la comunidad Moodle.

El proyecto fue inicialmente propuesto y diseñado a finales del año 2013 como una posible solución a ciertas necesidades de **SEAS** (la empresa cliente) de gestionar una serie de cursos, generalmente de pequeña magnitud y corta duración, que no tienen cabida en su intranet y campus virtual específicamente construidos para su oferta formativa más estable y ampliamente establecida.

SEAS (Estudios superiores abiertos) es una empresa centrada en formación superior así como otros tipos de cursos en diversas áreas divididas en líneas de negocio.

Así pues, el uso de **Moodle, en conjunto con las herramientas desarrolladas en este proyecto** forma una alternativa a la ya existente plataforma de formación a distancia de SEAS.

La plataforma propia de SEAS ha sido desarrollada por la empresa **Hiberus Tecnología** (compañía especializada en la consultoría de negocio y la prestación de servicios tecnológicos), y esta propuesta de proyecto fue realizada en el contexto del mismo equipo que ha desarrollado y mantiene la plataforma, tratando la propuesta como una posible mejora y ampliación de la versatilidad de ésta.

El desarrollo de las herramientas y complementos para Moodle ha sido realizado teniendo en mente las necesidades de integración de Moodle con la plataforma de SEAS, pero a la vez, en todo momento se ha mantenido el diseño y arquitectura de dichos elementos independiente de cualquier característica o concepto específico de la empresa cliente y/o su plataforma de formación. En resumen, la plataforma de SEAS constituye una solución muy específica y adaptada a sus propias necesidades (a modo de *Servicio*), pero tanto el propio Moodle como todo lo incluido en este proyecto se tratan de soluciones generales a diversas necesidades (a modo de *Producto*).

La razón para realizar el proyecto de esta manera se fundamenta en cumplir el objetivo principal y motivación personal del autor del proyecto de llevar a cabo un sistema formado por una serie de herramientas que puedan ser de utilidad para cualquier persona, organización o empresa que utilice Moodle y así maximizar la utilidad y posibles aplicaciones del proyecto. Por otra parte, esto es totalmente

[1] Licencia de software libre que garantiza a los usuarios la libertad de usar, estudiar, compartir y modificar el software. Ver http://gnu.org/copyleft/gpl.html

necesario si se desea publicar el software de un proyecto mediante una licencia de software libre, como es el caso.

Gracias al cumplimiento de estos objetivos, a junio de 2014, se ha podido aprovechar el desarrollo de partes de este proyecto desde Hiberus para ser ofrecidas tanto a SEAS como a otros clientes de diversa índole.

Es necesario destacar que la propuesta inicial contenía una serie de características que SEAS necesitaría en cualquiera de sus plataformas de formación, así como otras características que se consideraron necesarias por Hiberus o por el autor del proyecto, a falta de una aceptación y definición con exactitud de los requisitos del proyecto por parte de la empresa cliente. Debido a esto se dispuso de cierta libertad de desarrollo.

Por último es conveniente indicar que, por decisión directa del autor del proyecto, se ha puesto mayor énfasis y esfuerzo en una de las partes principales del proyecto (la personalización de los sitios Moodle individuales), como se expone más adelante (ver sección 2 del tercer capítulo de esta memoria).

1.2 Descripción general del alcance del proyecto

El sistema propuesto busca crear una plataforma **multi-sitio** tipo **campus virtual** que engloba una serie de sitios ágilmente replicables, todos ellos basadas en Moodle. Cada sitio del multi-sitio tendrá una lista idéntica de características disponibles, pero deberá ser fácilmente desplegado y personalizado mediante una interfaz de gestión única de tales sitios.

La facilidad de replicación es un aspecto fundamental para SEAS, ya que integra en su plataforma diferentes contextos de trabajo mediante el concepto de líneas de negocio. Es decir, mediante una separación lógica de los datos en su intranet son capaces de crear plataformas independientes para diferentes centros educativos, escuelas, acciones formativas en empresas, etc.

Esta separación lógica permite mantener una sola base de datos, con las ventajas de integración y estandarización que ello supone, pero a la vez ser capaz de contener varias plataformas independientes e individualizadas.

Como Moodle no ofrece ningún tipo de separación de contextos, se debe utilizar un sitio web diferente para cada línea de negocio, con su propia base de datos. De lo contrario, no habría forma posible de adaptar las configuraciones generales, aspecto, visibilidad de cursos, usuarios, permisos, etc. del sitio dependiendo de la línea de negocio a la que pertenecen los alumnos y el personal docente.

Por tanto, para facilitar la gestión y mantenimiento de varios sitios Moodle, uno por línea de negocio, así como de agilizar el proceso de puesta en marcha y personalización del aspecto y características de cada sitio, se ha diseñado e implementado un sistema basado en **dos herramientas independientes: una aplicación gestor multi-sitio y un tema para Moodle**.

El gestor multi-sitio es una aplicación web que contiene, organiza, instala y mantiene los diferentes sitios Moodle. Su propósito es **automatizar** la configuración de **aspectos técnicos** como bases de datos, subdominios o distribución de ficheros para hacer el sistema lo más transparente y de fácil uso que sea

posible. Es importante destacar que se trata de una herramienta construida alrededor de Moodle, no dentro de Moodle.

En cambio, el tema para Moodle es un complemento o extensión (*plugin*) que se integra en cada sitio individual. Su principal responsabilidad es la **personalización** del aspecto de cada sitio. Combina una interfaz gráfica de ayuda con la edición avanzada de los estilos.

Además, aunque Moodle ofrece muchas de las funcionalidades básicas que se requieren en el campus virtual, éstas pueden no adaptarse completamente a las necesidades de SEAS. Algunas de estas funcionalidades son la visualización de calificaciones y estadísticas de acceso en el expediente del alumno o la matriculación masiva de alumnos.

En los casos en los que no se cumplen las necesidades, se han implementado una serie de mecanismos de integración entre Moodle y la intranet de SEAS, de forma que se permita, por ejemplo, gestionar automáticamente los usuarios de alumnos y profesores asociados a Moodle y su autenticación transparente desde el campus virtual de SEAS o ver el expediente de un alumno de Moodle desde la intranet.

Para conseguir esta comunicación e integración, se ha utilizado el soporte ya existente de servicios web en Moodle, y varias de las funciones básicas que contiene toda instalación de Moodle.

Todas las funcionalidades adicionales requeridas se han diseñado y construido como un complemento (*plugin*), ya que Moodle tiene como una de sus principales ventajas el soporte para el desarrollo de una gran variedad de complementos (temas, bloques, sistemas de autenticación, actividades, servicios web…).

Es deseable seguir este modelo de desarrollo con el objetivo de no modificar el núcleo de Moodle ya que:

1. Facilita enormemente la actualización del sistema base en el futuro o incluso durante el desarrollo. Esto es debido a que se puede reemplazar todo el núcleo de Moodle por uno más reciente sin miedo a sobrescribir modificaciones propias del mismo.
2. Evita la introducción de nuevos fallos/bugs en Moodle. Se trata de un proyecto de dimensiones considerables, y no se debería modificar su base sin tener un conocimiento en profundidad de su diseño al completo.

1.3 Esquema de los módulos principales del proyecto

Una vez han sido presentadas y aclaradas las diferentes motivaciones y necesidades que han dado forma a este proyecto podemos pasar a describir los principales problemas que soluciona, por qué se ha decidido tratar estos problemas y, en líneas generales, cómo han sido abordados técnicamente.

1.3.1 Replicación y configuración de sitios Moodle automatizada

La **primera parte** de desarrollo del proyecto consistió en la solución de este conjunto de funcionalidades, y se realizó a modo de una **aplicación web independiente** llamada "*MMoodle*" (MultiMoodle) que contiene, gestiona y controla el acceso a los diferentes sitios Moodle que forman un multi-sitio.

Las características y funcionalidades más destacables que lo definen son:

1. Utilización de un sistema de **plantillas de sitios Moodle** para crear sitios a partir de ellas.
2. Creación de la base de datos y ficheros a partir de una plantilla indicada, todo realizado **automáticamente al crear un sitio**.
3. Configuración y posibilidad de modificación del dominio accesible asociado al sitio, título y descripción de forma automática.
4. **Auto-encaminamiento** del dominio escogido al sitio Moodle final sin necesidad de ningún tipo de configuración por parte del usuario ni de reiniciar el servidor web.

El auto-encaminamiento es una característica importante, ya que evita la configuración manual del enlace entre la dirección web del nuevo sitio (dominio) y los archivos de Moodle correspondientes. Sin esta funcionalidad, no podríamos afirmar que el gestor multi-sitio automatiza el proceso de crear un sitio completamente.

Por ejemplo, un sitio creado con el dominio misitio.es sería accesible simplemente introduciendo http://misitio.es en un navegador web (asumiendo que somos propietarios del dominio, y éste direcciona a nuestro servidor).

Además, partiendo de que por ejemplo la aplicación multi-sitio esté en el dominio mmoodle.com, nuestro sitio también será accesible desde misitio.es.mmoodle.com. Esto, además de ser conveniente para realizar pruebas en el caso de que todavía no seamos propietarios del dominio real, nos permitiría crear un multi-sitio utilizando solamente un dominio y múltiples subdominios como el del ejemplo.

Es importante recalcar que para utilizar el resto de funcionalidades del proyecto no es en absoluto necesario la instalación de este módulo. Sólo se trata de una herramienta para la agilización de la gestión de múltiples sitios Moodle.

1.3.2 Personalización de sitios Moodle

Por decisión y motivación del autor del proyecto, se trata de la parte principal del proyecto, al ser la más **potencialmente reutilizable** por otros usuarios en la comunidad Moodle. Por lo tanto es la parte que se propuso publicar libremente de forma prioritaria.

Para llevar a cabo los objetivos propuestos, se ha implementado la solución a modo de un **tema para Moodle**, al tratarse de la forma más común de personalizar el aspecto y comportamiento de la plataforma prácticamente al completo.

El tema ha sido desarrollado desde la base para ser altamente personalizable mediante una interfaz de usuario de fácil uso que permite pre-visualizar las personalizaciones realizadas navegando por el sitio Moodle real para, posteriormente, guardar y aplicar el aspecto personalizado.

Las características más destacables que ofrece son:

1. Diseño web moderno y adaptativo[2] (*responsive*) para la utilización de Moodle desde ordenadores de escritorio, tabletas y teléfonos móviles. Pueden visualizarse varios ejemplos en la sección 1 del capítulo 2 del anexo A.
2. Potente herramienta de personalización que permite:
    a. Gestión visual y clasificada de los elementos que forman Moodle de forma interactiva.
    b. Tres diseños base desde los que partir (básico, casi-plano, con degradado).
    c. Identificar y restaurar los valores modificados.
    d. Importación y exportación de estilos.
    e. Inclusión de CSS personalizado.
3. Se incluyen otra serie de características que enriquecen el tema:
    a. Ajustes generales como los textos e imágenes de fondo para el sitio, cabecera y pie de página, favicon[3]…
    b. Personalizar los elementos del menú de navegación principal del sitio.
    c. Carrusel de diapositivas y varios spots publicitarios en la página principal.
    d. Selección de las secciones que forman la página de autenticación.
    e. Gestión de enlaces a redes sociales y aplicaciones móviles.
4. Diseño base cuidado y utilizable directamente sin necesidad de ser personalizado anteriormente.

En cuanto al diseño web adaptativo, cabe aclarar que no consiste en la realización de diferentes estructuras de la página para cada tipo de dispositivo. Para simplificar el diseño, se crea una única estructura de página que se adapta al ancho de pantalla disponible en el momento de visualización.

Las técnicas principales que ayudan a realizar esta adaptación son el redimensionamiento y reordenación de bloques, la ocultación de ciertas partes de la página en dispositivos con un tamaño de pantalla considerado pequeño y el cambio del aspecto de partes muy específicas (por ejemplo el menú de navegación) para mejorar la usabilidad de ellas y ahorrar espacio en pantalla.

1.3.3 Integración de Moodle con otras plataformas y funcionalidades complementarias

Para la solución de otros problemas de menor complejidad que surgieron durante el proyecto se han implementado una serie de complementos adicionales para Moodle cuyos detalles se exponen en la sección 3 del tercer capítulo de este documento.

[2] Conjunto de técnicas de diseño web que permiten que el contenido se adapte al tamaño de pantalla disponible en cada dispositivo. Ver http://es.wikipedia.org/wiki/Dise%C3%B1o_web_adaptable
[3] Un favicon, también conocido como icono de página, es una pequeña imagen asociada con una página o sitio web en particular. Los navegadores web suelen mostrar el favicon de la página visitada, si ésta lo posee, en la barra de direcciones y en el encabezado de la pestaña correspondiente. Ver http://es.wikipedia.org/wiki/Favicon

1.4 Resumen de los posteriores capítulos

Esta memoria se ha estructurado en tres capítulos principales:

1. Gestión y planificación – Expone la planificación y avance del proyecto a lo largo de sus diferentes fases. También define el periodo de duración y tiempo de trabajo invertido en cada una de las fases, componentes y tareas.
2. Trabajo realizado – Explica en mayor detalle el contexto tecnológico de cada uno de los componentes del proyecto, su funcionamiento, la solución adoptada, alternativas consideradas y problemas destacables durante el desarrollo.
3. Conclusiones y trabajo futuro – Pone fin a la memoria concretando las conclusiones recabadas durante y al final del proyecto, indica el trabajo que queda por hacer y termina con una valoración personal del autor del proyecto.

1.5 Anexos a esta memoria

Anexo A, Demostración y manual de usuario – Lectura recomendada. Este anexo se centra en demostrar y ejemplificar en profundidad todas las funcionalidades y características de las partes de este proyecto, en secciones claramente diferenciadas.

A la vez que se demuestran las funcionalidades, se explica cómo utilizarlas y los resultados que proporcionan, a modo de manual de usuario.

Este anexo resulta especialmente interesante si se desean ver imágenes de los resultados de este proyecto, al contener gran cantidad de ellas. Por tanto, al contar con este anexo, el número de imágenes de los resultados en la memoria es reducido.

2. Gestión y planificación

2.1 Planificación y avance natural del proyecto

La propuesta, desarrollo y puesta en marcha de este proyecto se ha realizado en tres fases diferenciadas.

Vamos a ver qué tareas se realizaron en cada una de ellas.

2.1.1 Fase previa de análisis, investigación y elaboración de requisitos

Esta fase, **previa a la propuesta del proyecto final de carrera**, consistió en la obtención de unos requisitos previos de la empresa cliente para la utilización de Moodle de forma integrada con su plataforma existente, así como el análisis del cumplimiento en Moodle de necesidades generales en la futura implementación del proyecto, como pueden ser la posibilidad de incluir todo tipo de archivos multimedia, la gestión masiva de matriculaciones de alumnos o la importación de cursos y otro tipo de recursos dedicados al aprendizaje online.

Esta primera fase previa a la iniciación del proyecto fue de **carácter investigativo**, es decir, desde Hiberus Tecnología se anticipó que la empresa cliente, entre otras, iba a necesitar Moodle para la impartición de determinados cursos, y se quiso estar preparados ante ello con anterioridad. Se realizó un esfuerzo adicional no solicitado por el cliente.

Se escogió Moodle como plataforma de formación para este proyecto debido a su madurez demostrada, y su gran nivel de utilización en múltiples sectores, tanto a nivel académico como empresarial, así como por su diseño modular que facilita la extensión del propio sistema.

Así pues, se realizó un análisis en profundidad de las posibilidades que ofrece Moodle, su posible utilización por parte de SEAS, y se aprovechó este mismo análisis para **mejorar el conocimiento** en Hiberus **de la tecnología Moodle** para ser capaces de ofrecer soluciones personalizadas, adecuadas y de forma ágil a todo tipo de futuros clientes mediante la definición e implementación de una serie de requisitos de utilidad general para todos ellos.

Durante este período de investigación, se llegó a la conclusión de que **era necesario el desarrollo de dos herramientas imprescindibles** si se deseaba simplificar y automatizar en la medida de lo posible la puesta en marcha de sitios Moodle:

1. Un **gestor multi-sitio que facilitase todas las tareas repetitivas** del ámbito de operaciones y sistemas que son necesarias para la creación de un sitio Moodle: gestión de directorios, archivos y base de datos de cada sitio, enlazado de dominios a cada sitio e instalación de Moodle básico, principalmente.
2. **Un editor potente e interactivo del aspecto, estilo e imagen de marca de los sitios individuales** del multi-sitio. Para posibilitar esto, se determinó que la opción óptima consistía en desarrollar un tema para Moodle que permitiese ser auto-personalizado de forma muy flexible, en lugar de ser un módulo del gestor multi-sitio.

El hecho de que el editor de estilos (y por tanto el tema) sea independiente del multi-sitio, permite integrarlo en cualquier otro entorno Moodle, además de convertirlo en una de las partes más interesantes de este proyecto y poder ser publicado en la comunidad de Moodle.

Esta fase comprendió los meses de **mayo a septiembre de 2013**.

2.1.2 Fase de implementación del proyecto

Se trata de la fase más prolongada de este proyecto. A falta de una confirmación de los requisitos por parte de la empresa cliente, el autor del proyecto decidió comenzar a trabajar en las soluciones que permitirían cumplir los requisitos previstos.

También es necesario recalcar que, aunque los requisitos fueron elaborados y presentados a la empresa cliente, debido a la naturaleza investigativa del proyecto y el haber sido **planificado como una inversión extra por parte del equipo** de Hiberus Tecnología para el beneficio de sus clientes, su desarrollo se realizó íntegramente por el autor del proyecto y fuera de las horas de trabajo.

Primer componente – Gestor multi-sitio

Se decidió comenzar el desarrollo por el gestor multi-sitio y su conjunto de funcionalidades esenciales e interesantes para un proyecto final de carrera. Es decir, se implementaron todas las partes imprescindibles para el funcionamiento del multi-sitio, y se dejaron para más adelante otro tipo de funcionalidades secundarias y de poco interés como pueden ser la gestión de usuarios, roles y permisos (queda para trabajo futuro, ver sección 2 del cuarto capítulo de este documento).

El desarrollo del gestor multi-sitio comprende:

- La creación de un sitio web para la herramienta.
- La programación del proceso capaz de generar un sitio a partir de otro sitio que actúa como plantilla.
- La visualización en tiempo real del proceso (que dependiendo de las opciones escogidas puede tardar de uno a varios minutos).
- El auto-encaminamiento de sitios a partir de peticiones al servidor según el dominio de la petición, sin ninguna configuración adicional necesaria.

El proceso de generación de sitios y el sistema de auto-encaminamiento fueron las dos partes principales de este componente del proyecto, y de duración similar. Cabe destacar que el auto-encaminamiento de sitios necesitó de un tiempo de aprendizaje y familiarización con el funcionamiento avanzado de reglas del servidor Apache (ver sección 1.3 del tercer capítulo de este documento para más detalle).

Segundo componente – Tema para Moodle

El desarrollo continuó con la implementación de la herramienta que permitiese personalizar cada sitio individual con facilidad, en esencia, creando un nuevo tema para Moodle construido desde el primer momento con este objetivo. El desarrollo de este tema, en general, supuso una **labor de investigación**

mayor que la del gestor multi-sitio, que fue una labor principalmente de programación. Aunque esto no quiere decir que el tiempo de programación del tema fuera menor, también fue más alargado.

En resumen, se trata de la parte principal del proyecto, y en la que más horas se ha invertido, como puede verse en la tabla 3 del segundo capítulo de este mismo documento.

El desarrollo del tema incluye:

- Investigación y aprendizaje de Moodle, su sistema de complementos y entorno de programación de temas y otras características.
- Creación del tema desde cero: esqueleto de la aplicación, diseño de los diferentes tipos de páginas (siempre adaptativo) de un sitio Moodle, parámetros de configuración generales y la estructuración del sitio que facilita la construcción del gestor visual de estilos.
- Investigación del funcionamiento de las tecnologías a utilizar para el gestor visual de estilos (principalmente LESS, ver explicación en la sección 2 del tercer capítulo de este documento).
- Implementación de las funcionalidades del gestor visual de estilos, y posteriormente, de una potente interfaz de usuario para su utilización.
- Revisión de la implementación de la compilación de estilos con LESS para mejorar rendimiento, flexibilidad y facilidad de actualización del sistema a las últimas especificaciones de LESS.

Esta fase comprendió los meses de **octubre de 2013 a abril de 2014**.

2.1.3 Fase de pruebas, ajustes y despliegue del proyecto

Una vez terminada la fase de desarrollo principal, surgió la necesidad de demostrar a SEAS las posibilidades del sistema desarrollado con Moodle, y poco después, de integrar Moodle con la plataforma de SEAS a un mayor nivel (matriculación automática, expediente de alumno sincronizado, acceso transparente al sitio Moodle desde el campus de SEAS…).

Por tanto se comenzó a utilizar por primera vez el tema de Moodle en un sitio real. Gracias a esto, se fueron encontrando **problemas para los que se desarrollaron mejoras y correcciones** que permitieron enriquecer el resultado final de este módulo del proyecto.

Posteriormente se utilizó el tema para otros clientes diferentes. En el Anexo A se pueden visualizar algunas capturas de varios resultados reales de la puesta en marcha de este proyecto.

También se realizaron pruebas de despliegue del gestor multi-sitio en diversas máquinas, aunque todavía no esté siendo utilizado, ya que la cantidad de sitios manejada todavía es pequeña y una herramienta multi-sitio no resulta imprescindible por el momento.

Las pruebas realizadas se limitaron a la utilización de los componentes del proyecto para la preparación y puesta en marcha de varios sitios. Durante este proceso de preparación se pudieron detectar problemas y carencias que fueron solucionadas en esta misma fase.

Esta fase comprendió los meses de **mayo a julio de 2014**.

2.2 Horas invertidas en el proyecto

En las siguientes tablas se puede consultar una estimación de las horas invertidas en cada fase del proyecto y un desglose de las horas invertidas en cada una de las tareas que forman cada fase.

Fase del proyecto	Componente	Tarea	Horas invertidas
Fase previa al inicio del proyecto		Realización de primeras pruebas con Moodle y análisis del alcance de las funcionalidades ofrecidas	30
		Estudio de los recursos necesarios para cada sitio Moodle, definición de las operaciones necesarias para desplegar y mantener un sitio y elaboración de ideas para automatizar las mismas	35
		Comparación de las características requeridas por el campus de SEAS y las características ofrecidas por Moodle	15
		Análisis de la capacidad de extensión de Moodle e integración con otras plataformas mediante servicios web	25
		Propuesta de proyecto acorde a las características requeridas (aquellas ofrecidas por Moodle y las que debe desarrollar este proyecto)	10
		Total fase I	**115**

Tabla 1- Horas invertidas en fase previa al inicio del proyecto

Fase del proyecto	Componente	Tarea	Horas invertidas
Fase de implementación	Gestor multi-sitio	Base del módulo gestor multi-sitio: autenticación, base de datos, estructura estandarizada de directorios y ficheros	15
	Gestor multi-sitio	Desarrollo del sistema capaz de crear sitios Moodle a partir de un sitio plantilla con diferentes modos de replicado de plantillas	40
	Gestor multi-sitio	Interfaz de usuario para la creación de sitios y visualización en tiempo real del progreso	10
	Gestor multi-sitio	Sistema de enlazado automático de sitios a subdominios	45
		Total fase II parte 1 (Gestor multi-sitio)	**110**

Tabla 2 - Horas invertidas en fase de implementación (primera parte: gestor multi-sitio)

Fase del proyecto	Componente	Tarea	Horas invertidas
Fase de implementación	Tema para Moodle	Investigación y aprendizaje de APIs[4] de Moodle relativas a la creación de temas y fundamentales para el desarrollo en Moodle	15
	Tema para Moodle	Creación de un nuevo tema para Moodle desde cero, con diseño adaptativo y construido para ser personalizable mediante el gestor visual de estilos más adelante	80
	Tema para Moodle	Características generales de configuración y personalización del tema (logo, pie, favicon, diapositivas, bloques publicitarios…)	35
	Tema para Moodle – Gestor visual de estilos	Investigación del funcionamiento de las tecnologías a utilizar para el gestor visual de estilos	15
	Tema para Moodle – Gestor visual de estilos	Desarrollo de prueba de concepto del gestor visual inspirado en otras herramientas similares	15
	Tema para Moodle – Gestor visual de estilos	Desarrollo de funcionalidades esenciales (compilación, guardado, importación y exportación, visualización del sitio con los estilos compilados...)	70
	Tema para Moodle – Gestor visual de estilos	Interfaz de usuario avanzada y rica en funcionalidades para el gestor visual de estilos	65
	Tema para Moodle – Gestor visual de estilos	Búsqueda e implementación de una mejor solución a la compilación de estilos LESS	25
		Total fase II parte 2 (Tema para Moodle)	**320**

Tabla 3 - Horas invertidas en fase de implementación (segunda parte: tema para Moodle)

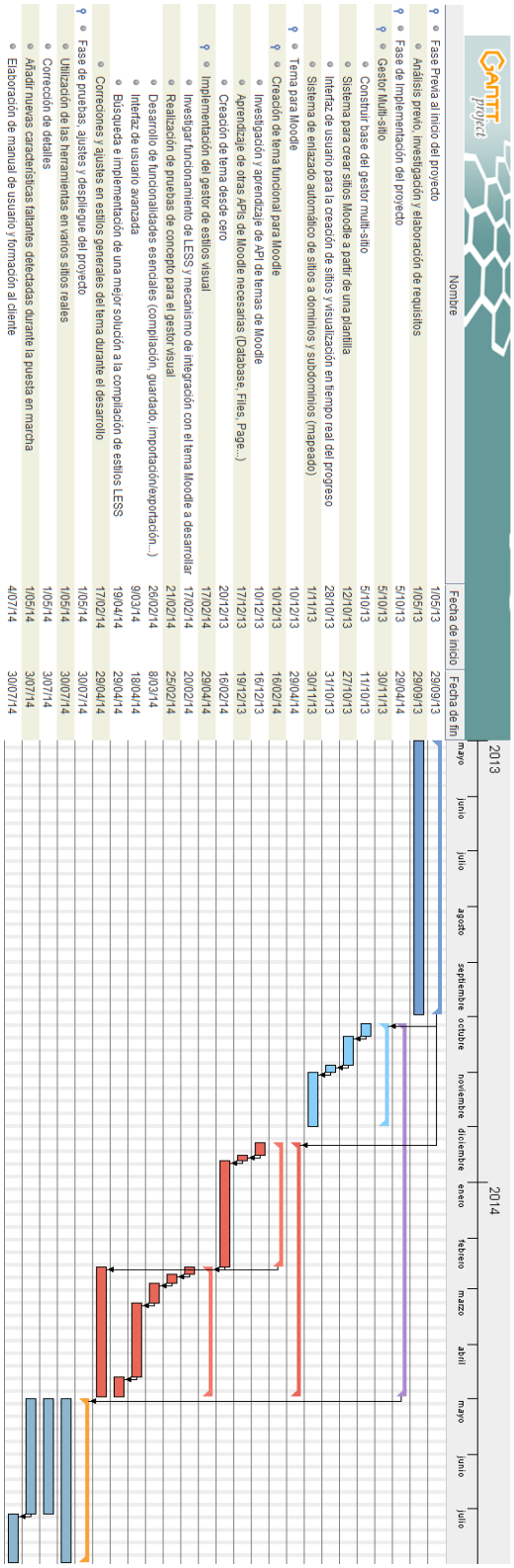
Fase del proyecto	Componente	Tarea	Horas invertidas
Fase de pruebas, ajustes y despliegue		Realización de pruebas de implantación y uso real del sistema para varios clientes	50
		Detección y corrección de errores, mejoras y ajustes	40
		Desarrollo de nuevas características no tenidas en cuenta durante la fase de implementación, necesarias para la integración con la plataforma de SEAS	70
		Elaboración de manuales de usuario y pautas para la correcta integración	20
		Total fase III	**180**

Tabla 4 - Horas invertidas en fase de pruebas, ajustes y despliegue

TOTAL PROYECTO: 725 Horas

[4] API: Interfaz de programación de aplicaciones; es el conjunto de funciones y procedimientos y métodos que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción

2.3 Diagrama de planificación del proyecto



En el diagrama GANTT de la figura 1 quedan representadas las diferentes fases del proyecto, junto con las tareas e hitos de mayor importancia que constituyen cada fase.

En dicho diagrama se muestra el periodo de tiempo que abarcó cada fase y cada tarea.

Figura 1 - Diagrama GANTT del proyecto

3. Trabajo realizado

Debido a la existencia de dos componentes principales independientes contenidos en este proyecto (gestor multi-sitio y tema para Moodle), este capítulo queda divido en dos grandes secciones dedicadas a cada uno de ellos y una tercera sección para otros desarrollos de menor calibre dentro del proyecto.

En cada una de las dos subsecciones principales se explica el funcionamiento y modo de implementación de sus funcionalidades principales, las técnicas utilizadas, soluciones adoptadas y problemas o dificultades destacables encontrados durante el desarrollo (con su correspondiente solución).

3.1 Gestor multi-sitio

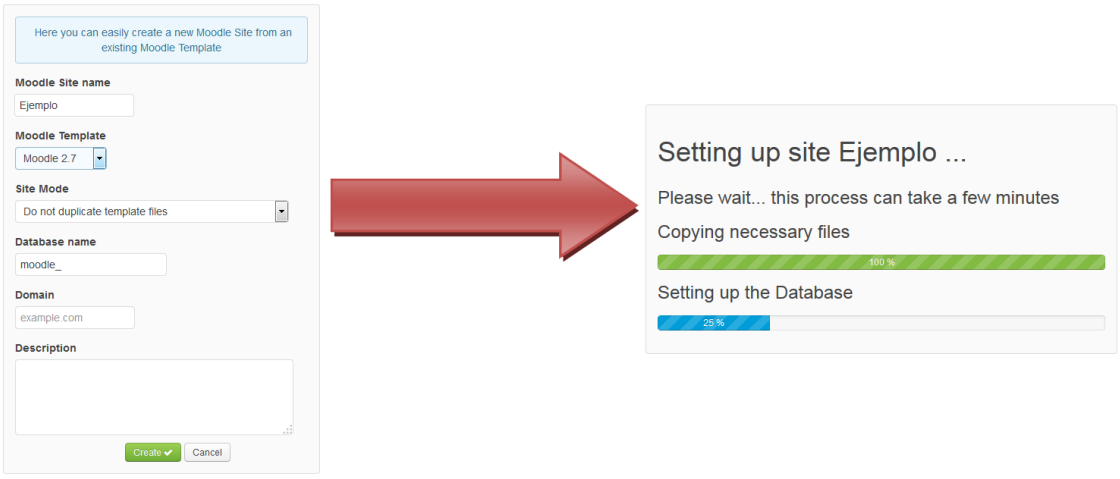


Figura 2 - Creación de un sitio en el gestor multi-sitio

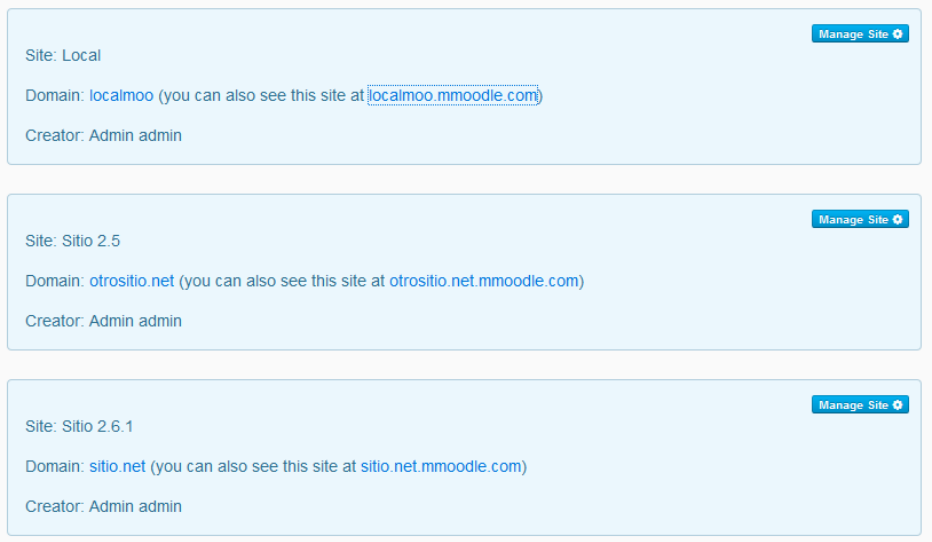


Figura 3 - Listado de sitios administrados por el gestor multi-sitio

3.1.1 Esquema técnico simple

La tecnología utilizada se trata de PHP, Apache y MySQL. PHP es un lenguaje de programación de uso general del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico. Apache es un servidor web de considerable madurez y flexibilidad, muy utilizado en conjunto con PHP. MySQL es un sistema gestor de bases de datos de libre distribución, utilizado en multitud de aplicaciones.

Al contrario que en Moodle, se utiliza un patrón MVC[5] facilitado por el uso de Zend Framework 1 para simplificar la estructura de la aplicación y para aprovechar la gestión de modelos de base de datos y formularios de Zend. Zend es un *framework* (marco de trabajo) ampliamente utilizado para el desarrollo de aplicaciones web con PHP. Ver http://framework.zend.com para más información.

La justificación de la elección de dichas tecnologías es, por una parte la familiaridad del autor del proyecto con ellas (y del resto del equipo en Hiberus Tecnología), y por otra parte por homogeneidad con el propio proyecto Moodle, que también utiliza PHP.

Zend Framework necesita de un aprendizaje inicial, pero al tratarse de un framework MVC típico, su uso básico es muy similar al de muchos otros framework web, y por tanto tiene una curva de aprendizaje poco pronunciada. El autor del proyecto ya cuenta con conocimientos avanzados y experiencia con Zend Framework 1, pero la documentación de referencia ha sido consultada en la referencia bibliográfica ***(1)***.

3.1.2 Planteamiento de soluciones adoptadas y alternativas

En el gestor multi-sitio se tomaron una serie de decisiones técnicas sin demasiadas alternativas posibles, pero las exponemos a continuación de todas formas.

Proceso de creación de un nuevo sitio

El hecho de basar la creación de sitios en el duplicado de otros sitios plantilla es una **solución simple a la replicación de plataformas Moodle** de forma sencilla y rápida. La instalación automatizada de Moodle sería una solución innecesariamente compleja e imposibilitaría la preparación de configuraciones base de una mínima complejidad en las plantillas.

Por tanto, el copiado de los archivos necesarios junto con la base de datos es una solución práctica y ampliamente probada, ya que en esencia migrar un sitio Moodle a otro servidor consiste en realizar este mismo proceso manualmente. Nuestra herramienta **automatiza y estandariza este proceso** para permitir contener cada sitio en el multi-sitio de la misma forma, y permite el ahorro de espacio en disco mediante un modo de reutilización de plantillas, en lugar del modo que las duplica al completo.

[5] El modelo–vista–controlador (MVC) es un patrón de arquitectura de software que separa los datos y la lógica de negocio de una aplicación de la interfaz de usuario y el módulo encargado de gestionar los eventos y las comunicaciones.

Servidor web y encaminamiento automático de sitios Moodle

En cuanto a la elección de Apache como servidor web, la razón principal es la experiencia del autor del proyecto con este servidor. **La principal alternativa** es el servidor **nginx**, pero la experiencia del autor con este servidor es nula. Por otra parte Apache es el servidor más utilizado con diferencia, y permite configuraciones de encaminamiento de dominios y hosts de varias formas bastante flexibles que suplen nuestras necesidades.

Para el encaminamiento (*routing*) automático de dominios se utiliza una estrategia basada en la combinación de las siguientes funcionalidades de Apache: ***VirtualHosts, RewriteEngine*** **y** ***RewriteMap*** mediante ficheros de texto plano para el mapeado de dominios a los directorios adecuados. Se puede encontrar buena documentación relativa a estas funcionalidades en la página web de la organización *Apache software foundation*. Específicamente, se consultaron las páginas de las referencias bibliográficas ***(2)*** y ***(3)***.

La utilización del fichero de texto plano para el mapeado de dominios es imprescindible si queremos posibilitar un auto-encaminamiento de sitios a directorios con una estructura compleja y/o dependiente de información externa adicional al nombre del sitio, como es nuestro caso (como hemos visto antes, un sitio puede escoger si duplica o no los ficheros de una plantilla en su propio directorio).

Resumen de los procesos y estructuras de datos del gestor multi-sitio

En la figura 4 se puede observar un diagrama general de las clases principales que implementan el proceso de creación y sincronización de un sitio, junto con una pequeña explicación de su cometido.

Dicho diagrama pretende mostrar una visión simplificada, o de alto nivel, de las clases que lo forman y cómo interactúan entre ellas. Se ha considerado que introducir demasiado detalle lo haría inmanejable y de poca utilidad en esta memoria. Por lo tanto no debe tomarse como un diagrama restringido y acotado por las especificaciones de la notación UML.

Veamos una explicación de las clases del diagrama. Los modelos (representaciones de datos) son:

1. MoodleTemplate (Plantilla de sitio) – Incluye la ruta de los archivos de la plantilla y la base de datos.
2. MoodleSite (Sitio Moodle) – Incluye la información básica del sitio, su dominio, base de datos y la plantilla utilizada.
3. MoodleSiteSetupProgress (Progreso de configuración del sitio) – Contiene información de los porcentajes de realización de los subprocesos asociados a la creación de un sitio.
4. User (Usuario) – Representa un usuario del gestor multi-sitio.

Las clases de eventos (Event), entidades encargadas de enviar eventos (EventSender) y entidades encargadas de recibir y procesar eventos (EventListener) constituyen un pequeño sistema de comunicación de acciones realizadas y datos producidos en determinados procesos que posteriormente deben ser tratados por otros procesos. Por ejemplo, el proceso que crea el sitio debe recibir información del progreso de copiado de archivos y la base de datos del sitio para poder actualizar el porcentaje de realización de éstos.

Los procesos principales son:

1. ConfigureMoodleSite (Configurar sitio Moodle) – Sincroniza la información de la base de datos y el fichero de configuración del sitio Moodle. Utilizado al crear y modificar un sitio. Además ejecuta el subproceso ConfigureDomainMappingsFile, que se encarga de actualizar el fichero de mapeos de dominios a directorios.
2. CreateMoodleSiteFromTemplate (Crear sitio a partir de plantilla) – Realiza todas las operaciones necesarias para crear un sitio nuevo a partir de una plantilla. Los subprocesos incluidos son:
    a. CopyDirectory – Copiar un directorio (utilizado para duplicar los archivos privados y el código fuente si es necesario).
    b. CopyDatabase – Copiar la base de datos del sitio Moodle plantilla.

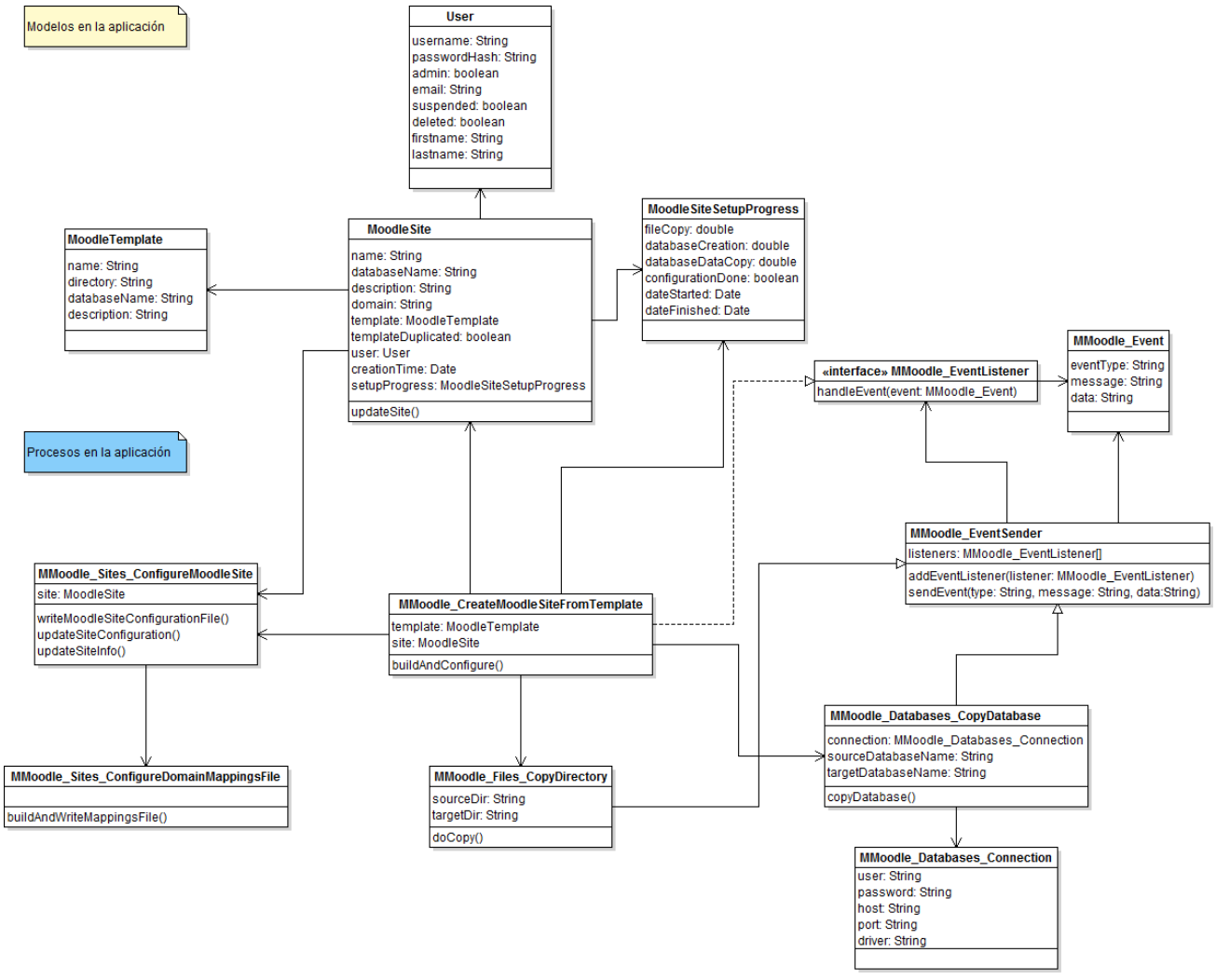


Figura 4 - Diagrama de clases del Gestor multi-sitio

3.1.3 Funcionamiento técnico de la herramienta

En esta sección vamos a ver una explicación de cada uno de los procesos que forman MMoodle, y los fundamentos de su diseño.

Configuración de la aplicación

Al momento de escribir esta memoria, las plantillas de MMoodle necesitan una configuración manual, tanto la base de datos como la colocación de los archivos de Moodle en su lugar apropiado. Al ser este un proceso sencillo y que se suele realizar una sola vez, carece de interés automatizar la subida de archivos en el ámbito de este proyecto, ya que aumentaría la complejidad de la aplicación sin una gran ventaja aparente.

Lo mismo sucede con los usuarios y permisos, el sistema soporta varios usuarios, pero no existe todavía una gestión de éstos desde la propia aplicación. Queda como tarea pendiente para completar la plataforma.

Creación de nuevos sitios a partir de plantillas

Los archivos que forman plantillas y sitios son alojados en dos carpetas diferentes:

1. Archivos de código fuente – Son los necesarios para la ejecución del sitio web Moodle, por lo tanto son **públicos**.
2. Otros archivos de Moodle: caches, alojamiento de archivos adjuntos, imágenes del sitio, etc. Son archivos **privados**, y su acceso es controlado por Moodle.

Partiendo de estas suposiciones, cada plantilla tiene asignada un subdirectorio público y otro privado, y lo mismo ocurre con los sitios Moodle. Gracias a esta organización interna de rutas, la aplicación puede averiguar en todo momento dónde se encontrarán los archivos de las plantillas y sitios, y poder duplicarlos para asignarlos a un nuevo sitio si es necesario.

Dicho esto, el proceso de crear un sitio sigue unos pasos relativamente sencillos:

1. Se hace una copia de los archivos privados de la plantilla y se dejan en el directorio privado del nuevo sitio.
2. Si el sitio lo requiere, se duplican los archivos de código fuente de la plantilla y se dejan en el directorio público del nuevo sitio.
3. Se hace una copia completa de la base de datos de la plantilla y se asocia al nuevo sitio.
4. Se ejecuta un proceso general de **sincronización** del sitio que:
    a. Modifica el nombre y descripción del sitio Moodle en su base de datos.
    b. Determina qué sitios deben ser accesibles a través de la carpeta pública de código fuente.
    c. **Adapta el archivo de configuración** del código fuente que utiliza el sitio o sitios (puede ser el código de la plantilla o el código propio duplicado) para aceptar peticiones de acceso al sitio o sitios a través de los dominios necesarios y enlazarlas a la base de datos apropiada según la petición.

d. **Regenera el archivo de mapeado de sitios a directorios** – Apache es capaz de detectar que el archivo ha cambiado y lo vuelve a cargar y cachear *en caliente*, sin necesidad de reiniciar el servidor y sin perder nada de rendimiento (de acuerdo a ***(3)***). De esta forma el encaminamiento del nuevo sitio es efectivo al instante.

Si se modifica o elimina un sitio, basta con volver a ejecutar el proceso de sincronización, de forma que se consigue simplificar y centralizar todo el proceso en un solo lugar.

Al crear un nuevo sitio, MMoodle ofrece dos formas diferentes de utilizar una plantilla:

1. Reutilizar el mismo código de Moodle de la plantilla para diferentes sitios.
2. Duplicar todo el código de Moodle de la plantilla para el nuevo sitio.

Dependiendo de la opción que elijamos, el proceso realiza el duplicado de algunos archivos o todos los archivos, y el proceso de sincronización genera un fichero de mapeado de sitios acorde a los directorios de código fuente que el sitio va a utilizar.

La tabla 5 muestra una comparación de las ventajas y desventajas de ambos modos.

Método	Ventajas	Desventajas
Reutilizar plantilla	Ahorro de espacio en disco Proceso de creación de sitio más rápido Facilidad de actualización de todos los sitios que utilizan la plantilla, así como de instalar complementos en todos ellos	No permite tener sitios con diferentes versiones de Moodle o de los complementos instalados
Duplicar plantilla	Permite tener sitios independientes con diferentes versiones de Moodle, así como un conjunto de complementos propio	Mayor uso de espacio en disco Proceso de creación de sitio más lento Al actualizar el código de Moodle (y sus complementos) de la plantilla no se actualizará el código del sitio La instalación de complementos en varios sitios requerirá la repetición del proceso en cada uno de los sitios

Tabla 5 - Comparación de los modos de utilizar una plantilla de sitio Moodle

Encaminamiento automático de los sitios Moodle

Al hablar de encaminamiento automático de los sitios, nos referimos a la carga de diferentes sitios web Moodle a partir de los dominios de las peticiones web, basándonos en un conjunto de **reglas generales que no precisen ser modificadas cada vez que se añade un nuevo sitio**.

Este es un proceso que ha requerido varias etapas de refinamiento durante el desarrollo hasta obtener los resultados deseados: un encaminamiento dinámico, automático y a prueba de errores comunes. Además, se ha conseguido hacerlo funcionar en las versiones 2.2 y 2.4 de Apache mediante una configuración única compatible con ambas versiones.

Veamos de qué nos sirve cada una de las funcionalidades utilizadas de Apache:

Funcionalidad o módulo	Utilidad
Virtualhosts	El uso de VirtualHosts permite definir diferentes aplicaciones web independientes en un mismo servidor
RewriteEngine	El RewriteEngine o Motor de Rescritura permite alterar la forma en la que se accede a contenidos de una aplicación web mediante la modificación transparente de las URLs, para servir páginas de forma inteligente y dinámica Se basa en la utilización de expresiones regulares y otras condiciones especiales que determinan la activación de la rescritura de una URL
RewriteMap	Es una característica que forma parte del Motor de Reescritura y que permite la utilización de recursos externos, por ejemplo otro programa o un fichero de texto, para establecer relaciones especiales que no se pueden definir mediante otro tipo de reglas básicas

Tabla 6 - Descripción de las funcionalidades necesarias para el encaminamiento de sitios Moodle

Una explicación simplificada del funcionamiento, sin entrar en detalles de la implementación, es la siguiente:

1. Durante el **proceso de sincronización**, MMoodle actualiza un archivo de texto plano que es leído y cacheado por Apache. Este archivo contiene **relaciones de mapeo directas entre dominios y directorios** de código fuente Moodle.
2. En dicha configuración de Moodle se establecen una serie de reglas que intentan tratar toda petición web que el servidor recibe como una petición a uno de los sitios Moodle.
    a. Si hay éxito en la detección del sitio asociado a partir del archivo anterior, se dirige la petición a tal sitio.
    b. Si no hay éxito y la petición está dirigida al dominio de la aplicación MMoodle, simplemente se carga la aplicación MMoodle (uno de los errores comunes mencionados al comienzo de esta página).
    c. En el caso de que la petición al dominio de MMoodle contenga un subdominio, de nuevo se intenta relacionar este subdominio con un sitio Moodle a partir del archivo de mapeo.

Si no hay éxito en la relación, se redirige la petición al dominio base de MMoodle, considerando que se trata de un error (similar al anterior error común).

d. En cualquier otro caso, la petición podría ir dirigida a otras posibles aplicaciones web que deban convivir con el sistema MMoodle en el mismo servidor Apache.

El **aspecto del archivo de mapeado** de sitios a directorios de código es similar al siguiente:

```
duplicado.com      moodle-public-sites/duplicado.com
prueba.net         moodle-public-templates/moodle263
localtest          moodle-public-templates/moodle263
local27            moodle-public-templates/moodle27
sitio27.net        moodle-public-templates/moodle27
otroduplicado.net moodle-public-sites/otroduplicado.net
```

Como podemos observar, se trata de un archivo tipo CSV sencillo en el que la columna izquierda contiene el dominio, y la columna derecha contiene el directorio de código Moodle asociado a ese sitio.

Los directorios de código contenidos en moodle-public-templates se corresponden con sitios que reutilizan el código de su plantilla original, mientras que los contenidos en moodle-public-sites utilizan una copia exclusiva del código de la plantilla original en la que se basan.

Con este sencillo archivo de relaciones clave-valor, Apache es capaz de servir las peticiones utilizando el directorio adecuado dependiendo del dominio de la petición. Por ejemplo, una petición al dominio prueba.net (o al subdominio prueba.net.mmoodle.com) sería servida con el código de la plantilla moodle263, y por supuesto, utilizando la base de datos y archivos privados del sitio prueba.net.

Para determinar qué base de datos y directorio privado utilizar, como hemos visto anteriormente, lo conseguimos gracias a la sincronización automática del archivo de configuración de cada plantilla y sitio. Por ejemplo, el archivo de configuración de la plantilla moodle263 contiene lo siguiente:

```
$serverName = $_SERVER['SERVER_NAME'];//Dominio de la petición a servir
if  ($serverName == "localtest") {
        $CFG->wwwroot = "http://localtest";
        $CFG->dbname = 'moodle_local';
        $CFG->dataroot = 'C:\dev\mmoodle\files\sites\localtest\moodledata';
} elseif  ($serverName == "localtest.mmoodle.com") {
        $CFG->wwwroot = "http://localtest.mmoodle.com";
        $CFG->dbname = 'moodle_local';
        $CFG->dataroot = 'C:\dev\mmoodle\files\sites\localtest\moodledata';
} elseif  ($serverName == "prueba.net") {
        $CFG->wwwroot = "http://prueba.net";
        $CFG->dbname = 'moodle_pruebanet';
        $CFG->dataroot = 'C:\dev\mmoodle\files\sites\prueba.net\moodledata';
} elseif  ($serverName == "prueba.net.mmoodle.com") {
        $CFG->wwwroot = "http://prueba.net.mmoodle.com";
        $CFG->dbname = 'moodle_ pruebanet';
        $CFG->dataroot = 'C:\dev\mmoodle\files\sites\prueba.net\moodledata';
}
```

3.1.4 Problemas y dificultades durante el desarrollo

Reglas de Apache y soporte de varios modos de utilización de plantillas

En la primera versión del gestor multi-sitio, solamente se implementó la opción de duplicar todos los archivos de cada plantilla, sin posible reutilización del mismo directorio de código fuente. A partir de esto se construyeron las reglas de Apache asumiendo que cada sitio siempre tenía su propia carpeta de código y siempre se encontraba en public-moodle-sites/dominiodelsitio. Gracias a estas suposiciones, el conjunto de reglas de Apache era relativamente sencillo y no precisaba de un archivo de mapeo de sitios a directorios.

Después se consideró la opción de no duplicar las carpetas de código fuente de las plantillas para agilizar el proceso de creación de sitios y ahorrar espacio en disco.

Una vez implementada tal opción, se encontró con el problema de que era **imposible determinar la carpeta de código de un sitio sabiendo solamente su dominio**. Entonces fue necesario:

- Investigar qué soluciones existen para solventar este problema con Apache.
- Aprender el funcionamiento de RewriteMap una vez se tomó como solución al problema.
- Realizar pruebas de concepto.
- Implementar el sistema de generación y sincronización de este archivo y adaptar las reglas para que lo utilizasen.

Funcionamiento del sistema de mapeo en diferentes versiones de Apache con un conjunto de reglas idéntico

Durante la fase de pruebas final, se detectó que el conjunto de reglas de Apache no funcionaba correctamente en la versión 2.2 de Apache. Estas reglas habían sido construidas utilizando la versión 2.4.

Como era un **requisito dar soporte a ambas versiones** y los cambios a realizar para cada versión eran de pequeña magnitud, se optó por adaptar las reglas de forma unificada. Es decir, se deseaba conseguir un único conjunto de reglas que funcionasen en ambas versiones.

Para ello se aprovechó la funcionalidad de apache de detectar qué versión está instalada mediante:

```
<IfVersion >= 2.4>
        ...Reglas específicas...
</IfVersion>
```

Pero no bastó con el anterior código, debido a que tal funcionalidad depende de la activación del módulo version_module. Por suerte, este módulo está disponible en ambas versiones de Apache, solo que no está activado por defecto en Apache 2.4 y sí lo está en Apache 2.2.

Para solucionarlo, basta con activarlo en la configuración si se trata de Apache 2.4.

3.2 Personalización de sitios mediante un tema para Moodle

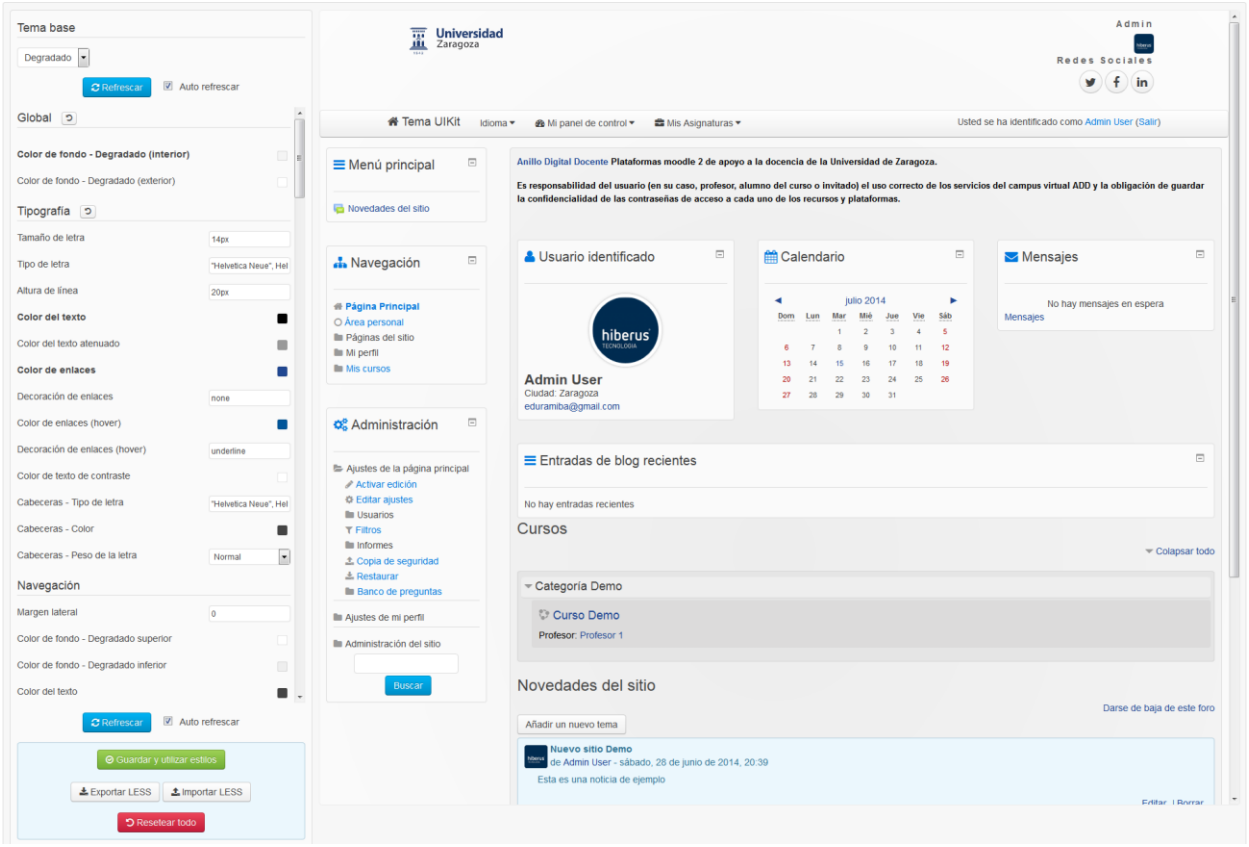


Figura 5 - Gestor visual de estilos del Tema UIkit

Esta se trata de la segunda y mayor parte del proyecto. Las razones por las que se decidió concentrar gran parte del esfuerzo la realización de un tema para Moodle son:

- **En el contexto de un multi-sitio, es importante el disponer de una herramienta que permita la personalización** de los diferentes sitios, e incluso en determinados casos que facilite la aplicación de un diseño base al que se puedan realizar una serie de ajustes menores que resultan en una serie de sitios de aspecto similar pero único. Esta herramienta, al igual que el gestor multi-sitio, debería ser sencilla de utilizar, flexible y además ofrecer resultados inmediatos a un usuario sin conocimientos de programación.
- Es un proyecto en el que **resulta útil e interesante** incorporar una serie de tecnologías recientes que lo hacen innovador y adecuado en el ámbito de un proyecto final de carrera. Además la utilización de una API existente es una parte importante en el día a día del desarrollo de software, por lo que permite al autor del proyecto ampliar su conocimiento y experiencia en el **uso y diseño de APIs**.
- Se trata de un **módulo potencialmente utilizable** por una gran cantidad de usuarios gracias a la existencia previa de una fuerte **comunidad** tanto de usuarios como de desarrolladores alrededor del proyecto **Moodle**.

El nombre escogido para este tema es Tema UIkit o *UIkit theme*. Esto es debido, como veremos más adelante, a la utilización del framework UIkit para la construcción, diseño y personalización del tema.

3.2.1 Esquema técnico simple

En el caso de implementar un tema para Moodle, o cualquier otro tipo de extensión o complemento, la **tecnología y estructuración del código viene impuesta por la naturaleza del proyecto Moodle y sus APIs**: PHP como lenguaje, normas de nombrado de clases y funciones, estructura de carpetas y nombres de ficheros, números de versión…

Gracias a estas estrictas normas y convenciones el núcleo de Moodle es capaz de determinar de forma inequívoca cómo comunicarse con un complemento, extraer información importante de tal complemento, permitir la integración de la extensión con Moodle y facilitar el desarrollo. Por otra parte, estas normas deben cumplirse rigurosamente a la hora de publicar un complemento en la página oficial de Moodle.

Tales imposiciones resultan naturales y fácilmente comprensibles a cualquier persona que haya participado en el diseño de una o varias APIs o SPIs[6] en un programa informático o haya tenido que implementar extensiones para otras aplicaciones o plataformas. Una breve lectura muy útil respecto a este tema se puede encontrar en las diapositivas de la referencia bibliográfica "*How to Design a Good API and Why it Matters*" ***(4)***.

La documentación oficial de Moodle para el correcto desarrollo de temas, el núcleo y otros complementos se puede encontrar, principalmente, en ***(5)*** (construcción de un tema), ***(6)*** (documentación general para desarrolladores) y ***(7)*** (uso de *Core APIs*).

[6] Una SPI (Service Provider Interface) tiene una función similar a la de una API. Pero en lugar de ofrecer una interface de uso de la librería o plataforma, proporciona diferentes interfaces (Service), que el programador debe implementar mediante clases (ServiceProvider) para construir estos servicios y que puedan ser integrados en la plataforma o aplicación

3.2.2 Características del tema

A continuación se enumeran las diferentes características principales del tema desarrollado:

Funcionalidad o característica	Observaciones
Diseño del tema adaptativo y moderno	Permite utilizar cómodamente el sitio web Moodle desde multitud de dispositivos con tamaños de pantalla diferentes El diseño intenta evitar parecerse al diseño clásico de Moodle, anticuado hoy en día
Potente herramienta de personalización de estilos del sitio	La **principal ventaja** de este tema es que ha sido construido desde la base para ser personalizado mediante una interfaz especialmente pensada para ello La interfaz se integra completamente en el sitio Moodle, y no requiere de configuración adicional ni de tratamiento de ficheros por parte del usuario ya que es automatizada y transparente El sistema de personalización **no requiere conocimientos de diseño web**, ni de los lenguajes en los que se apoya, pero a la vez permite a un usuario aprovechar tales conocimientos para conseguir mejores resultados
Menú de navegación complementario	El tema incluye un menú de navegación complementario a los bloques de Moodle Este menú contiene accesos rápidos a las partes más útiles de un sitio Moodle para el usuario. Por ejemplo el listado de sus cursos o acceso a su perfil Las partes a mostrar de este menú así como la nomenclatura de ellas también pueden ser personalizadas
Ajustes generales	El tema permite proporcionar una imagen de logo y pie de página, además de imágenes de fondo para la cabecera, cuerpo y pie de página si se desea Otros ajustes incluidos son el favicon, nota en el pie de página o un icono para el sitio si no se dispone de un logo
Complementos para la página principal	Es posible incluir un carrusel de diapositivas en la página principal (hasta 4) También se pueden configurar una serie de spots publicitarios Ambos elementos pueden ser mostrados solamente si el usuario no está autenticado o viceversa
Personalización de la página de autenticación	Permite ocultar ciertas secciones de la página de autenticación, así como cambiar el título de la caja de acceso por una imagen propia
Otras funcionalidades	Otras funcionalidades especiales del tema son la inclusión de enlaces a redes sociales, iconos para aplicaciones móviles, integración con Google Fonts, tracking mediante Google Analytics o poder modificar el comportamiento del listado de categorías y cursos de Moodle para solo mostrar cursos matriculados
Soporte multi-idioma	El tema está construido para soportar múltiples idiomas, y está inicialmente traducido al inglés y español

Tabla 7 - Características del tema Moodle

3.2.3 Planteamiento de soluciones adoptadas y alternativas

La decisión de realizar la implementación de la herramienta para personalizar estilos de forma avanzada mediante un tema para Moodle ya ha sido suficientemente explicada y justificada a lo largo de esta memoria, pero es necesario recordar que esta decisión también permite la implementación de otras características especiales y poco comunes que un cliente puede pedir.

Por ejemplo, algunas peticiones reales del cliente son la modificación del comportamiento de los listados de categorías y cursos para no mostrar aquellos cursos en los que el usuario conectado no está matriculado, o modificar la página de acceso al sitio para ser similar a la del campus virtual del cliente.

Para conseguir la alta personalización mediante el gestor visual de estilos, el tema se apoya principalmente en el uso de las siguientes tecnologías:

a. **UIkit** – Framework *front-end* (parte visible por el usuario de una página web) que ofrece un conjunto de estilos y utilidades para la creación de aplicaciones web modernas y que facilita el diseño adaptativo e interactividad para todos los navegadores web comunes. Su principal ventaja es que **ha sido creado para posibilitar la creación de diferentes temas** a partir de él, por lo que **encaja a la perfección en este proyecto**. Nótese que se ha decidido utilizar un framework web como UIkit en lugar de desarrollar uno propio porque supondría tanta o más complejidad y trabajo como el resto del proyecto. Ver http://getuikit.com para más información.
b. **LESS** – Se trata de una extensión del lenguaje CSS, a modo de preprocesador, que añade múltiples características de lenguajes de programación clásicos a CSS como variables, funciones, jerarquía, extensibilidad, etc. que permiten construir CSS que sea más fácil de mantener, extender y diseñar temas con él. UIkit se basa en LESS para conseguir su extensibilidad. Ver http://lesscss.org para más información.

Respecto a UIkit, la gran alternativa es el framework *Twitter Bootstrap*, que goza de un alto índice de uso en gran cantidad de páginas web de todo tipo, también basado en LESS y permite crear temas a partir de él. Las razones para no utilizar este framework, mucho más conocido que UIkit, son las siguientes:

- UIkit proporciona de serie tres diseños en los que basar otros temas (ver capítulo 2 del Anexo A para más detalle y ejemplos gráficos).
- UIkit es utilizado profesionalmente por la empresa que lo ha publicado libremente para la creación de varios temas personalizables en CMS[7] como *Wordpress* o *Joomla*.
- Al ser menos común, da una apariencia única y original a nuestro tema. Ya existen varios temas para Moodle basados en *Twitter Bootstrap*, aunque no implementen un gestor de estilos interactivo como el de nuestro tema.
- Por preferencia personal del autor del proyecto respecto a *Twitter Bootstrap*, tanto por el aspecto base del framework como por la organización del código LESS y CSS de UIkit, más amigable y menos intrusivo en el proceso de desarrollo con él.

[7] Sistema de gestión de contenidos, herramienta para la creación y administración de contenidos, principalmente en páginas web, por parte de los administradores, editores, participantes y demás usuarios

Podemos encontrar una extensa y detallada documentación con numerosos ejemplos de UIkit en ***(8)***.

En cuanto a **LESS, su principal competidor** es otro preprocesador CSS llamado **SASS**. Aunque al decidir utilizar UIkit, *Twitter Bootstrap* o casi cualquier otro framework front-end, LESS viene impuesto, con toda seguridad existen otros framework basados en SASS. De todas formas, LESS es una solución preferible frente a SASS en este proyecto por las siguientes razones:

- LESS, al contrario que SASS, extiende CSS con una sintaxis muy natural y similar a CSS, facilitando el aprendizaje del lenguaje.
- El compilador de LESS oficial está implementado en lenguaje JavaScript. Esto permite utilizarlo desde cualquier navegador web, una ventaja especialmente útil para nuestro gestor visual de estilos.
  También existen otros compiladores no oficiales para PHP y otros lenguajes, pero suelen estar un paso por detrás de las especificaciones oficiales. El compilador de SASS está implementado en lenguaje Ruby, mucho menos conocido y portable que JavaScript.
- LESS es más flexible y potente que SASS en ciertas características, por ejemplo la extensión de clases. Es cierto que SASS es más intuitivo a la hora de realizar condicionales o bucles pero esto rara vez es necesario, y LESS al ser más reciente que SASS todavía está en fase de crecimiento.

La documentación del lenguaje LESS, sus características, funciones y utilización del compilador JavaScript oficial puede encontrarse en ***(9)***.

3.2.4 Funcionamiento técnico del tema

Gestor visual de estilos

El módulo del gestor visual de estilos se basa en el uso de JavaScript y jQuery[8] para realizar de forma interactiva y sin necesidad de recargar la página multitud de tareas como:

- Leer las variables, sus tipos, nombres y valores por defecto para mostrarlas al usuario, detectar cambios en ellas y restablecer sus valores.
- Compilar el código LESS con los valores modificados mediante less.js (el compilador JavaScript oficial).
- Reemplazar los estilos del sitio en tiempo real para pre-visualizar las personalizaciones sin necesidad de guardarlas en el sitio real todavía.
- Comunicarse con el servidor para obtener el código LESS, realizar post-procesado de este código requerido por Moodle o guardar los estilos finales.
- Importar y exportar archivos sin participación del servidor.

La incorporación de jQuery en el gestor de estilos resulta especialmente útil debido a la gran cantidad de manipulaciones realizadas al árbol DOM[9] de la página para alterar los estilos del sitio real por el nuevo CSS compilado y para mostrar las variables gestionables de forma dinámica (en función del tema base seleccionado, de los valores originales y modificados o de las acciones realizadas mediante múltiples botones).

El autor del proyecto ya tiene suficiente experiencia con jQuery, pero la documentación de referencia de esta librería JavaScript ha sido consultada desde ***(10)***.

En la siguiente página podemos ver un diagrama de secuencia de las acciones y operaciones principales que ocurren al utilizar el gestor visual de estilos (figura 6).

[8] jQuery es una biblioteca de JavaScript, creada inicialmente por John Resig, que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX a páginas web - http://jquery.com

[9] Document Object Model - Es una API que proporciona un conjunto estándar de objetos para representar documentos HTML y XML, un modelo estándar sobre cómo pueden combinarse dichos objetos, y una interfaz estándar para acceder a ellos y manipularlos

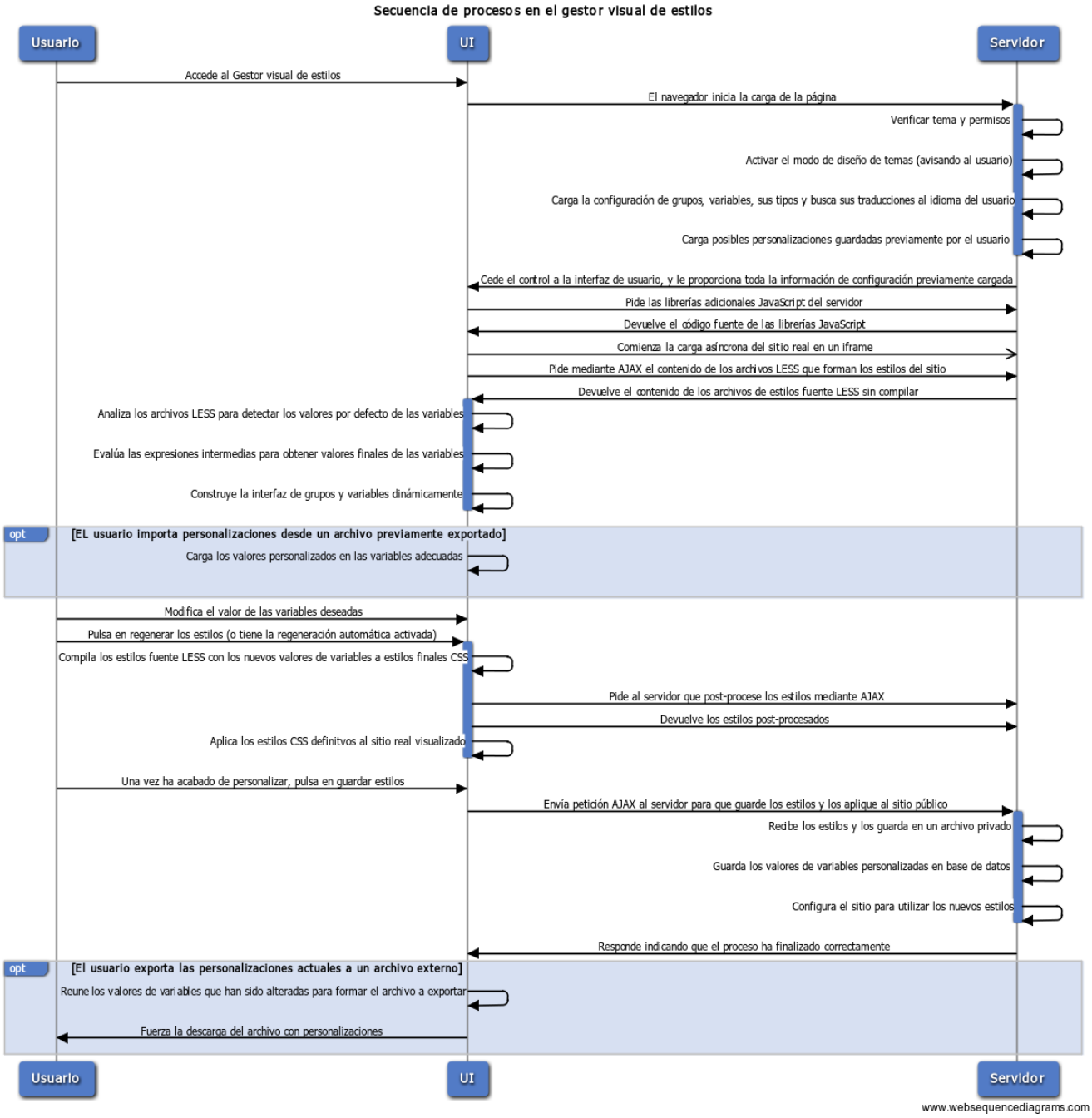


Figura 6 - Diagrama de secuencia del gestor visual de estilos

La explicación de los pasos más importantes que el sistema sigue cuando se accede e interactúa con el gestor visual es la siguiente:

1. Verificar que el tema UIkit está seleccionado, el usuario está autenticado y tiene los permisos de administración necesarios. En caso contrario, se muestra un mensaje de error informativo.
2. Activar el modo de diseño de temas de Moodle si no está activo – Este modo evita que Moodle cachee las hojas de estilos del sitio en un solo archivo (necesario para poder alterar los estilos mediante JavaScript).
3. Cargar la configuración de grupos y variables, y traducir los textos asociados – Este proceso se realiza en el servidor y entrega la información lista para utilizar al código JavaScript del gestor de estilos. La configuración de variables disponibles, sus tipos y sus valores seleccionables está almacenada en un archivo simple estático que facilita la modificación del configurador de estilos sin necesidad de alterar el código fuente del servidor o interfaz de usuario.
4. Comprobar si el usuario ha guardado previamente estilos personalizados para precargar el tema base, los valores de las variables y el posible código adicional personalizado desde base de datos
5. Inicializar todas las dependencias de otras librerías JavaScript utilizadas – less.js para la compilación de estilos, Spectrum Colorpicker ***(11)*** para la selección y gestión de colores y CodeMirror ***(12)*** para la caja de texto en la que un usuario avanzado puede introducir código LESS/CSS propio.
6. Cargar los archivos LESS que forman el tema y averiguar los valores por defecto de cada variable.
7. Construir y mostrar la interfaz de usuario dinámicamente a partir de la configuración – Este es el proceso más intenso en código jQuery.
8. Mostrar el sitio Moodle pre-visualizado – Incluyendo un iframe[10] del propio sitio y alterando sus estilos con JavaScript podemos pre-visualizar nuestros estilos personalizados y navegar por el sitio libremente.
9. Compilar, post-procesar los estilos y aplicarlos al sitio Moodle cada vez que sea necesario – Ocurre cuando el usuario refresca los estilos manualmente o modifica una variable y el auto-refresco está activado.
10. Guardar los estilos y variables modificadas cuando el usuario pulse en guardar y aplicar estos estilos al sitio real.
11. Importar/Exportar variables personalizadas – Implementado mediante un sistema capaz de escribir/leer en un archivo los valores de aquellas variables que el usuario ha modificado, y el código adicional en el caso de que lo haya.

Nota: El post-procesado de estilos CSS consiste en una serie de modificaciones que Moodle hace a los estilos para incluir correctamente URLs de imágenes, iconos, tipos de letra y otros detalles ligados a Moodle.

[10] Un iframe es un elemento HTML que permite insertar o incrustar un documento HTML dentro de otro documento HTML principal

Implementación del resto de características del tema

La implementación del tema en Moodle sin incluir el gestor visual de estilos también se trata de un proceso laborioso en el que hay que realizar tareas como las siguientes:

- Adaptar y revisar los estilos estándar de Moodle en CSS para integrarse correctamente con las reglas LESS que forman el tema. Para realizar esta tarea, que resulta bastante compleja, ha servido de ayuda el basarse en una adaptación previa de los estilos al framework *Twitter Bootstrap* (similar a UIkit) que ya había sido realizada por el equipo de Moodle.
- Extender los estilos de Moodle mediante reglas LESS propias principalmente para conseguir una mejor integración con UIkit (estilos de formularios, botones, cajas de contenido, etc.) a lo largo de todo el sitio.
- Crear uno o varios *layout* (disposición general de elementos en una página) de propósito general que se adapten a diferentes tipos páginas de Moodle. Por otra parte, tales layout deben implementarse teniendo en cuenta las técnicas de diseño adaptativo.
- Extender y modificar las funciones que se encargan de mostrar el HTML de un sitio Moodle para que se adapten a las necesidades tanto de nuestro tema como de UIkit. Para esta tarea, Moodle ofrece una API novedosa y en continuo crecimiento que permite alterar la representación del sitio (*renderers* API).
- Definir, gestionar y codificar la gran cantidad de ajustes que el tema permite a la hora de generar el layout y contenido en general.
- Utilización de las diferentes APIs de Moodle para tareas como guardar datos en tablas propias de la base de datos, ficheros o acceso a ajustes y configuraciones.

La siguiente imagen (figura 7) muestra la estructura de archivos que se debe seguir al crear un tema para Moodle y explica la finalidad de cada archivo. Incluye otros archivos de nuestro tema que no son esenciales para cualquier complemento de este tipo, ya que son específicos de este proyecto.

Para facilitar la lectura de la imagen, se ha seguido un código de colores simple:

- Los directorios o archivos **impuestos o requeridos** por la API de temas o complementos de Moodle han sido marcados de color **rojo**.
- Los directorios o archivos **incluidos libremente por nuestro tema** han sido marcados de color **azul**.

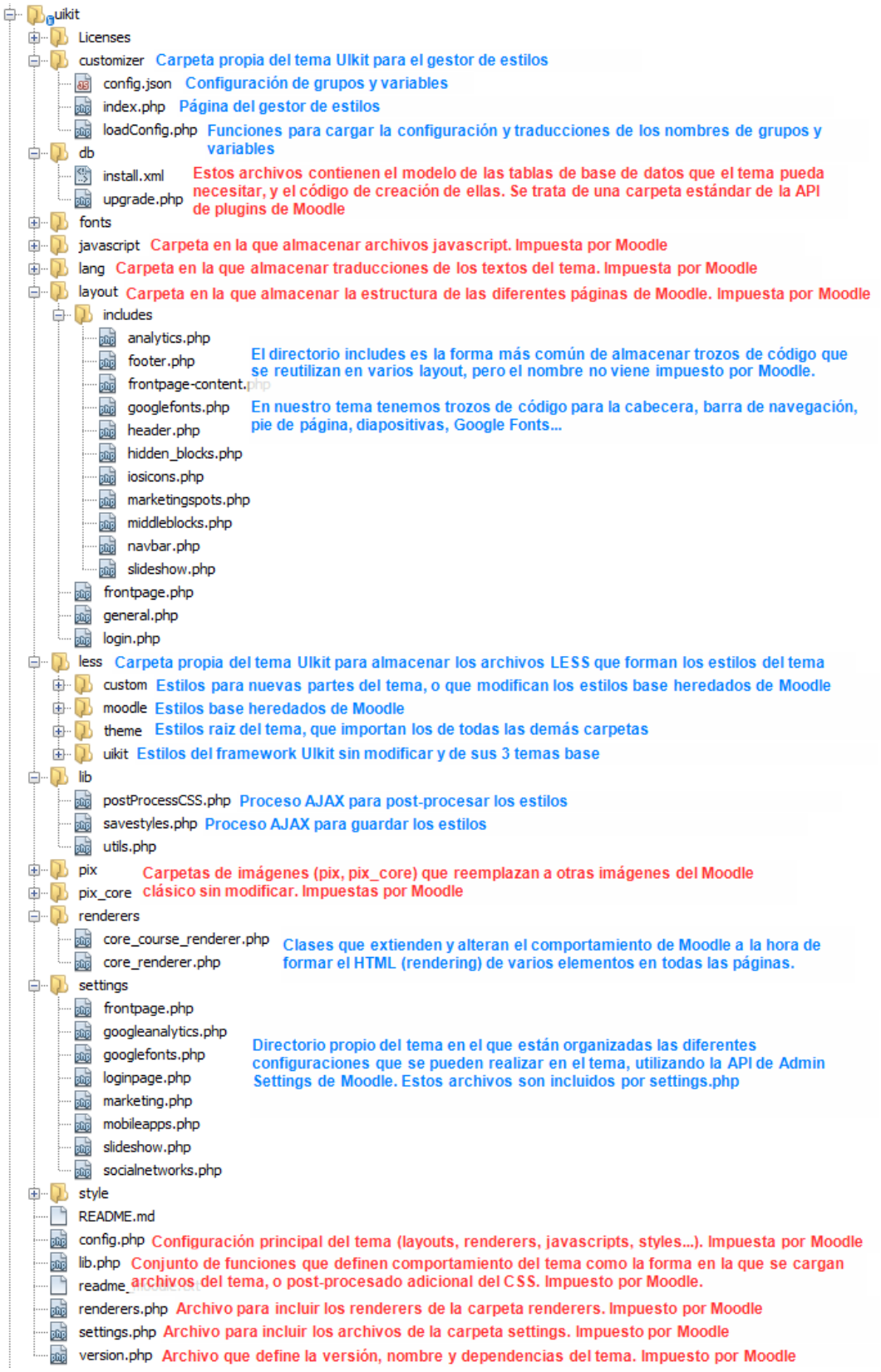


Figura 7 - Estructura de archivos del tema

3.2.5 Problemas y dificultades durante el desarrollo

Aunque ya ha quedado claro que la implementación del gestor visual de estilos es bastante compleja debido a su factor investigativo y la combinación de varias tecnologías avanzadas y relativamente recientes, no se han expuesto los problemas y retos de su desarrollo. Veamos estos problemas.

Integración del compilador LESS adecuado

En la primera implementación del gestor visual de estilos, la compilación de estilos LESS se realizaba en el lado del servidor mediante una librería PHP llamada **Lessphp** (http://leafo.net/lessphp). Esta decisión fue condicionada por la necesidad de realizar el post-procesado de Moodle al CSS compilado final. Al realizar tanto la compilación como el post-procesado en el servidor el lado del cliente era más sencillo y previsible.

Por otra parte, dicho compilador cuenta con un rendimiento elevado y, aunque comenzaba a estar **obsoleto** a principios de 2014, servía para realizar la compilación de los primeros estilos LESS de nuestro tema.

Además, el compilador JavaScript oficial (less.js), por entonces en su versión 1.6, todavía era bastante lento y pesado como para utilizarlo en un navegador web, y el volumen de estilos del tema Moodle es bastante elevado, resultando en tiempos de compilación demasiado largos que perjudicaban la experiencia de usuario en el gestor visual de estilos.

Conforme el desarrollo del tema fue avanzando, el autor del proyecto quiso utilizar funcionalidades más avanzadas y recientes de LESS que, desafortunadamente, lessphp no soporta. Esta librería sigue estancada hoy en día en la versión 1.4.2 de LESS (la versión actual a agosto de 2014 es 1.7.4).

Una funcionalidad especialmente útil no disponible es la directiva ":extends", que permite hacer que una regla CSS extienda todos los estilos de cualquier otra regla deseada. Por ejemplo, se deseaba que todos los botones del sitio se comportasen como si tuviesen la clase "uk-button" (botón de UIkit), algo muy sencillo de conseguir con ":extends", pero muy engorroso de realizar de otras forma.

Fue en este momento, a principios de marzo de 2013, en el que se **empezó a buscar alternativas**. Para el lenguaje PHP, existe otra librería que sí cumple las últimas especificaciones de LESS. Esta librería es **Less.php** (https://github.com/oyejorge/less.php), que pretende ser la sucesora de Lessphp. Su ventaja es que sí implementa las especificaciones más recientes, ya que pretende ser un reflejo de la implementación oficial, pero sus desventajas la hacen una opción inviable:

- Es difícil de utilizar y contiene varios bugs si se utiliza en el sistema operativo Windows – El autor del proyecto invirtió una gran cantidad de tiempo hasta hacerla funcionar.
- Es increíblemente lenta.

Así pues, después de muchas horas de pruebas y la frustración de no poder mejorar los resultados ni poder utilizar las funcionalidades más recientes de LESS, no se llegó a ninguna solución y el autor del proyecto se resignó a seguir utilizando Lessphp y se continuó con el desarrollo del resto de funcionalidades.

Al cabo de unas dos semanas se volvió a evaluar el estado de las librerías disponibles, y muy convenientemente, se observó que el equipo de **less.js** había lanzado la versión 1.7 de esta librería el día 27 de febrero de 2014. Dicha versión **había mejorado el rendimiento considerablemente**, resultando en tiempos de compilación muy similares a los de Lessphp, y con todas las ventajas que supone utilizar la implementación oficial. Por lo tanto se decidió sin ningún tipo de duda adaptar el gestor de estilos para utilizar less.js y el problema quedó finalmente resuelto.

A continuación, en la tabla 8, queda resumida la experimentación con las diferentes librerías.

Librería	Tiempo de compilación del tema	Observaciones
Lessphp	2 segundos	Inviable al estar **obsoleta** y no mantenida
Less.php	30 segundos en la primera versión probada, 15 segundos en la más reciente	Inviable al ser **demasiado lenta** Problemática en Windows
Less.js	7 segundos en la primera versión probada, de 1 a 2 segundos en la más reciente	Implementación oficial y **solución óptima tras la publicación de la versión 1.7**

Tabla 8 - Comparación de compiladores LESS probados

3.3 Extensiones adicionales para Moodle

Como desarrollo adicional necesario para este proyecto, en el ámbito de las necesidades de SEAS, se han desarrollado una serie de extensiones para Moodle de menor importancia para la comunidad y complejidad que el tema.

Todo desarrollo de extensiones adicionales se ha realizado **a partir de la fase de pruebas ajustes y despliegue, y se continúa realizando hasta la fecha actual**. Es decir, según las necesidades específicas de SEAS, se siguen desarrollando componentes adicionales para este proyecto. Pero no por ello quedan pendientes de finalización los dos grandes componentes que se han presentado como trabajo principal. Al contrario, estos componentes se consideran maduros, estables y en un estado completo y de mantenimiento.

Hay que destacar que ninguna de estas extensiones se publicará bajo una licencia de software libre. Al tratarse de funcionalidades especialmente diseñadas para la integración del flujo de trabajo de SEAS con los sitios Moodle no podrán ser publicadas, ya que implementan requisitos solicitados directamente por SEAS, y al estar dichos requisitos estrechamente ligados a su intranet y campus virtual.

A continuación veremos una breve explicación de cada extensión.

3.3.1 Servicios web adicionales para la integración de Moodle con otras plataformas

Esta extensión es la que ha requerido mayor tiempo de desarrollo ya que se encarga de exponer varios servicios web desde un sitio Moodle para ser utilizados por procesos externos.

Moodle ofrece una API para desarrollar extensiones que definan el funcionamiento de servicios web adicionales a los que la plataforma Moodle básica ya ofrece.

Algunos de los servicios externos más importantes que se han implementado son:

1. Obtención de la estructura de calificaciones completa de un curso – Utilizado para la visualización de expedientes de alumno(s) así como para la realización de varios informes relativos a la estructura de los cursos.
2. Obtención de las notas de uno o varios alumnos asociadas a la estructura de calificaciones completa de un curso – Utilizado para la visualización de expedientes de alumno(s) y para la obtención de varios informes relativos a las calificaciones de uno o varios alumnos pertenecientes a un curso.
3. Obtención de algunas estadísticas de utilización de la plataforma por los alumnos.
4. Configuración de elementos de menú de navegación especiales para algunos alumnos.

3.3.2 Sistema de autenticación mediante un servicio web externo

Esta extensión es utilizada para permitir tanto a alumnos como profesores de SEAS el conectarse a un sitio Moodle con el mismo usuario y contraseña que en el campus o intranet de SEAS.

El funcionamiento es el siguiente:

1. En Moodle se configura la URL y parámetros del servicio web externo.
2. Durante el proceso de acceso al sitio Moodle, la extensión se conecta al servicio web para consultar si debe permitir el acceso a un usuario.
3. Si hay éxito, se fuerza el acceso del usuario. En caso contrario se sigue el proceso de autenticación normal de Moodle.

3.3.3 Sistema de autenticación automatizada mediante *tokens* de uso único (códigos auto-consumibles y no predecibles)

Esta extensión es utilizada para redirigir y auto-autenticar al usuario a un sitio Moodle que ya está conectado en el campus o intranet de SEAS.

La extensión funciona de forma similar a la anterior, sólo que se incorpora un nuevo parámetro que indica que se trata de este tipo de autenticación, y el sistema emplea códigos de acceso que son inutilizados después de su consulta por seguridad.

4. Conclusiones y trabajo futuro

Hoy en día cada vez es más común la formación académica mediante plataformas de aprendizaje online, y Moodle se trata de una de las plataformas de este tipo más extendidas y versátiles, adaptándose a un gran abanico de necesidades.

Mediante este proyecto hemos conseguido **mejorar la gestión automatizada y ágil de plataformas basadas en Moodle** así como mejorar la experiencia de usuario gracias al desarrollo de un tema moderno y que tiene en mente la usabilidad de Moodle en dispositivos de tamaño reducido.

Por lo tanto, se han presentado una serie de avances que potencialmente beneficiarían a la comunidad Moodle y el progreso de ésta hacia una plataforma de aprendizaje online más madura y con mayores posibilidades.

En cuanto a la **puesta en marcha** real de este proyecto, puede considerarse **exitosa** ya que además de haberse cumplido los objetivos de la propuesta inicial del proyecto final de carrera, el proyecto **ha evolucionado durante el desarrollo** (las características de personalización han aumentado respecto a las esperadas en un primer momento).

La evolución del proyecto ha permitido construir un conjunto de herramientas, módulos y utilidades que solucionan al mismo tiempo los problemas del cliente principal y los problemas de otros clientes (actuales y futuros) o incluso de otros usuarios de Moodle, gracias a la publicación del código fuente bajo una licencia de software libre.

Por último, hay que destacar que a 22 de agosto de 2014 **se ha realizado la publicación del tema Moodle** bajo la licencia de software libre GPLv3 y éste ha sido **aceptado en el directorio de complementos** de la comunidad de usuarios y desarrolladores de Moodle. El tema publicado puede consultarse en https://moodle.org/plugins/view.php?plugin=theme_uikit.

Por otra parte**, en el Anexo A se demuestra de forma extensiva** la utilización de las herramientas de este proyecto para el despliegue de varios sitios Moodle desde la base, ejemplificando las características más importantes.

4.1 Nivel de cumplimiento de objetivos propuestos

Analicemos la propuesta de este proyecto (octubre de 2013) en un ejercicio de valoración y crítica de resultados:

> *El sistema propuesto pretende crear una plataforma multi-sitio tipo campus virtual que engloba una serie de plataformas ágilmente replicables, todas ellas basadas en Moodle.*
>
> *El desarrollo de este sistema se debe a la necesidad de una alternativa por parte de SEAS (empresa cliente) de gestionar una serie de cursos, generalmente de pequeña magnitud y corta duración, que no tienen cabida en su intranet y campus virtual específicamente construidos para su oferta formativa más estable y ampliamente establecida.*

Los objetivos principales son:

- *Replicación y gestión automatizada de los sitios Moodle mediante una aplicación web externa independiente.*
- *Extensión y adecuación de Moodle a las necesidades específicas de la empresa.*

Cada plataforma del multi-sitio tendrá una lista idéntica de características disponibles, pero deberá ser fácilmente desplegada y personalizada mediante una interfaz de gestión única de tales sitios.

El gestor de plataformas permitirá:

- *Automatizar la configuración de aspectos técnicos como bases de datos, sub dominios o distribución de ficheros para hacer el sistema lo más transparente y de fácil uso que sea posible.*
- *Personalización del estilo de cada sitio. Combinará una interfaz gráfica de ayuda con la edición avanzada del código de los estilos.*
- *Crear y mantener cada sitio. Diferentes usuarios tendrán diferentes permisos y roles en el sistema. Esto permitirá establecer restricciones sobre los sitios que alguien puede gestionar y qué acciones puede realizar sobre ellos.*

[…]

A partir de la anterior cita, podemos afirmar que **todos los requisitos principales en ella han sido cumplidos** e incluso superados, ya que el texto define en esencia los dos módulos principales del proyecto (el gestor multi-sitio y la personalización de Moodle mediante el tema).

En cuanto al último punto que define la gestión de usuarios, permisos y roles del gestor multi-sitio, no ha sido implementada ya que se consideró de insignificante interés en el ámbito del proyecto final de carrera, y tampoco se ha necesitado por el momento.

Como podemos observar, los objetivos fueron propuestos de forma muy centrada en el ámbito de SEAS, pero posteriormente se implementaron e utilizaron las soluciones para ser adecuadas en múltiples ámbitos.

Veamos la parte final de la propuesta, que redacta los requisitos adicionales específicos a SEAS:

[…]

Funcionalidades docentes requeridas en cada sitio Moodle (a incluir mediante personalización y/o extensión):

- *Aunque Moodle ofrece muchas de las funcionalidades multimedia que se requieren, éstas son insuficientes (visor de PDF avanzado, integración con Google Docs, galerías de imágenes...).*
- *Comunicación entre alumnos y profesores.*
- *Realización de tests.*

Principales características adicionales específicas a añadir mediante extensiones propias:

- *Pantallas para la carga masiva de alumnos, profesores, materiales docentes y calificaciones.*
- *Alertas/avisos especiales en determinadas fechas o eventos como puntos de control de alumnos.*
- *Pasos de agenda o tareas asignadas al alumno con comportamiento bloqueante hasta ser completados. Integración de material multimedia.*
- *Confirmación de condiciones de matriculación.*
- *Seminarios: Gestión de plazas y presentación.*
- *Informes y listados varios.*

El primer listado de funcionalidades docentes se ha podido cumplir combinando las funcionalidades base de Moodle 2.6 y las funcionalidades de otros complementos existentes en la comunidad (en menor medida de lo esperado).

En cuanto al segundo listado de características adicionales:

- La carga masiva de usuarios (alumnos y profesores) se ha llevado a cabo integrando la plataforma de SEAS con Moodle directamente, a través de servicios web.
- Moodle soporta eventos de calendario a nivel de sitio, curso y usuario.
- La integración de material multimedia es bastante avanzada en Moodle y el bloqueo de actividades todavía no ha sido necesario, pero afortunadamente es posible desde Moodle 2.7 (lanzado el 10 de mayo de 2014).
- La confirmación de condiciones de matriculación finalmente no ha sido necesaria de implementar en Moodle porque se realiza siempre desde la plataforma de SEAS.

- Todavía no ha surgido la necesidad de organizar seminarios, por lo que no se ha implementado nada específicamente para ello. En caso de necesitarlos, podrían ser gestionados mediante el módulo *Booking*[11] (sistema de reservas) sin necesidad de desarrollar un sistema propio.
- Los informes y listados que no son parte del núcleo de Moodle se han solucionado con la integración del expediente del alumno en la plataforma de SEAS mediante servicios web.

A agosto de 2014 todavía se sigue ampliando la integración de la plataforma de SEAS con Moodle, a un nivel creciente de funcionalidades que no se previó en la propuesta inicial, pues han sido solicitadas más tarde, y que queda fuera del ámbito de este proyecto final de carrera.

No obstante es interesante indicar que gracias a los resultados de este proyecto se ha facilitado dicha integración y el cliente ha decidido utilizar la plataforma para la impartición de ciertos cursos.

4.2 Trabajo futuro

En esta sección vamos a describir ciertas mejoras y características adicionales que podrían añadirse al proyecto en el futuro.

4.2.1 Gestor Multi-sitio

Para completar la aplicación, en principio sería especialmente necesario:

1. Gestión de usuarios, permisos y roles.
2. Añadir la posibilidad de crear sitios a partir de otros sitios, además de a partir de plantillas. Esto sería relativamente sencillo de realizar teniendo en cuenta la organización de propósito general del código fuente que lleva a cabo el proceso de sincronización.
3. Completar el aspecto de la aplicación web (menú móvil, cabecera y pie…).
4. Detección de falta de permisos de escritura en directorios de la aplicación junto con texto de ayuda para solucionarlo.

4.2.2 Tema Moodle

Una mejora principal del tema sería la posibilidad de **guardar diferentes configuraciones personalizadas** como configuraciones predefinidas en la base de datos, así como **ofrecer unas cuantas configuraciones de ejemplo**.

Esto reduciría la necesidad de importar/exportar archivos con las configuraciones si se estuviesen comparando varias alternativas al mismo momento.

Por otra parte, al ser Moodle un proyecto de grandes dimensiones, es razonable que la comunidad detecte fallos y detalles a corregir una vez publicado.

[11] Es un complemento adicional para Moodle que permite a alumnos suscribirse a eventos. Estos eventos permiten, entre otras funcionalidades, indicar un número máximo de participantes, listas de espera, periodos de suscripción o mensajes de confirmación automáticos. Ver http://docs.moodle.org/27/en/Booking_module

4.3 Valoración personal

A nivel personal y profesional, la experiencia de este proyecto ha sido indudablemente favorable e interesante. Me ha permitido aplicar muchos de los conocimientos de programación web (back-end y front-end) que he obtenido a lo largo de mis estudios en Ingeniería Informática, y los que he obtenido laboralmente en otros proyectos.

También me ha servido a modo de investigación, aprendizaje y aplicación de nuevas técnicas y tecnologías, especialmente aquellas dedicadas a la elaboración de diseños web fácilmente personalizables, adaptativos y flexibles y aquellas empleadas para la construcción de APIs y extensiones de programas o plataformas.

Por otra parte, este proyecto combina varios tipos de tecnologías diferentes, especialmente todas aquellas necesarias para la creación de páginas web modernas.

El proyecto trata tareas que van desde la configuración automatizada de un servidor web, sistema de ficheros y base de datos hasta la estructuración de un sitio al completo, la elaboración de sus correspondientes estilos y la configuración de ellos mediante un proceso realizado en servidor en conjunto con una interfaz de usuario completamente interactiva.

Además, una de las ventajas más destacables de haber trabajado en este proyecto ha sido conocer a fondo el funcionamiento de Moodle y la estrategia de trabajo de su equipo y comunidad, ya que mantenerse abierto a nuevas metodologías, procedimientos y formas de organización siempre es un aspecto positivo y enriquecedor.

La ejecución de este proyecto me ha servido para experimentar con plataformas de aprendizaje online y las diferentes distribuciones estructurales de cursos que pueden ser implantadas en ellas.

Puedo afirmar que la dificultad del proyecto ha residido especialmente en dicha diversidad de tareas realizadas y en el objetivo de afrontar todo el proyecto como una solución generalizada, modular, reutilizable y de calidad que ha hecho posible su puesta en marcha con éxito y la publicación de algunas de sus partes.

Finalmente, ha servido de ayuda para habituarme a la comunicación con el cliente y a ser capaz de analizar, entender y modelar sus necesidades, además de poder explicar y demostrar las soluciones desarrolladas una vez llegado el momento de ponerlas en funcionamiento.

Bibliografía

1. Zend Technologies Ltd. (Último acceso: Octubre 2013) Guía de referencia del programador. Disponible en: http://framework.zend.com/manual/1.11/en/manual.html
2. Apache Software Foundation (Último acceso: Noviembre 2013) Documentación de referencia de Apache mod_rewrite. Disponible en: http://httpd.apache.org/docs/current/mod/mod_rewrite.html
3. Apache Software Foundation (Último acceso: Noviembre 2013) Using RewriteMap. Disponible en: http://httpd.apache.org/docs/current/rewrite/rewritemap.html
4. Bloch, J. (Último acceso: Enero 2014) How to Design a Good API and Why it Matters. Disponible en: http://lcsd05.cs.tamu.edu/slides/keynote.pdf
5. Moodle Pty Ltd. (Último acceso: Marzo 2014) Creación de un tema para Moodle. Disponible en: http://docs.moodle.org/dev/Creating_a_theme
6. Moodle Pty Ltd. (Último acceso: Julio 2014) Documentación general Moodle. Disponible en: http://moodle.com/
7. Moodle Pty Ltd. (Último acceso: Julio 2014) Wiki para desarrolladores de Moodle (Core APIs). Disponible en: http://docs.moodle.org/dev/Core_APIs
8. YOOTheme (Último acceso: Abril 2014) UIkit - Documentación y ejemplos. Disponible en: http://www.getuikit.com/docs/documentation_get-started.html
9. The Core LESS Team (Último acceso: Mayo 2014) Documentación de LESS y Less.js. Disponible en: http://lesscss.org
10. The jQuery Foundation (Último acceso: Abril 2014) Documentación de referencia de jQuery. Disponible en: http://api.jquery.com/
11. Grinstead, B. (Último acceso: Abril 2014) Documentación de Spectrum Colopicker. Disponible en: http://bgrins.github.io/spectrum/
12. Haverbeke, M. (Último acceso: Abril 2014) Documentación de CodeMirror. Disponible en: http://codemirror.net