

**ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES**  
UNIVERSIDAD POLITÉCNICA DE MADRID

José Gutiérrez Abascal, 2. 28006 Madrid  
Tel.: 91 336 3060  
[info.industriales@upm.es](mailto:info.industriales@upm.es)

[www.industriales.upm.es](http://www.industriales.upm.es)



**POLITÉCNICA**

**INDUSTRIALES**

**05 TRABAJO FIN DE GRADO**

**Sonsoles Jiménez Martín**

**TRABAJO FIN DE GRADO**

# **DISEÑO Y DESARROLLO DEL MÓDULO DE ALUMNOS PARA AULAWEB2.0**

**FEBRERO 2017**

**Sonsoles Jiménez Martín**

**DIRECTOR DEL TRABAJO FIN DE GRADO:  
Raquel Martínez Fernández  
Ángel García Beltrán**

## Agradecimientos

La realización de este trabajo no habría sido posible sin la colaboración y ayuda del equipo del laboratorio de Informática de la ETSII-UPM (Julio y David), así como de los profesores de la unidad docente de Informática Industrial. Especialmente a mi tutora, que siempre me ha dado soluciones a los problemas que he planteado y material de apoyo.

También quiero agradecer a mi familia el apoyo que me ha proporcionado en la realización de este proyecto.

## Resumen ejecutivo

Una plataforma de *Tele-enseñanza* o de *e-learning* es un espacio virtual de aprendizaje orientado a facilitar la experiencia de capacitación a distancia. Este tipo de plataformas ofrecen numerosos beneficios, tanto para los alumnos, que pueden acceder a los contenidos de sus cursos independientemente de donde se encuentren, eliminando la barrera de la distancia y del tiempo, como para los docentes, que tienen un mayor control sobre los contenidos y documentación que proponen a sus alumnos. Además, incrementan y facilitan la comunicación entre alumnos y profesores. El uso de plataformas de *e-learning* no está limitado a instituciones docentes sino que se extiende a empresas o a organizaciones informales de aprendizaje.

Este trabajo aborda la actualización de la plataforma de *Tele-enseñanza* que se utiliza en la Escuela Técnica Superior de Ingenieros Industriales (ETSII) de la UPM. La ETSII viene utilizando, desde hace algunos años, una herramienta de *Tele-enseñanza* y de comunicación con los alumnos denominada *AulaWeb*. *AulaWeb* ha cumplido perfectamente con su cometido pero, durante estos años, ha sido necesario incorporar una serie de módulos y funcionalidades que no estaban previstos inicialmente. Por esto, las bases de datos han tenido que ser modificadas, lo que ha llevado a cambios del diseño que las han hecho algo incomodas y con una estructura innecesariamente compleja. Por otra parte, el continuo incremento en su utilización hace conveniente el rediseño de esta plataforma, de forma que sea mucho más modular y escalable. Además, la plataforma actual *AulaWeb 1.0* estaba desarrollada con tecnología ASP.NET, cuya utilización supone un coste a la Universidad.

La existencia de soluciones informáticas abiertas y gratuitas, como son PHP, JavaScript y mySQL hace muy aconsejable acometer el rediseño de *AulaWeb* empleando este tipo de soluciones. Este es el objetivo de *AulaWeb2.0* que es el proyecto que enmarca este trabajo de fin de grado.

Como punto de partida de este trabajo, se ha realizado una recopilación de las ventajas que suponen las plataformas de *Tele-enseñanza*, tanto para el mundo académico como para las empresas. Se han incluido, además, algunos datos estadísticos que permiten augurar una expansión considerable de este tipo de plataformas en el futuro. De esta forma se justifica económicamente el esfuerzo realizado en este proyecto como forma de preparar a la ETSII para posibles acciones futuras. Esta ampliación se está realizando empleando soluciones de Software abierto y gratuito, concretamente utilizando PHP y JavaScript sobre servidores Apache. Este tipo de arquitecturas son cada vez más populares y la tendencia hacia su utilización es cada vez más alta.

En el trabajo se analiza el concepto de Software abierto o libre y se consideran distintas opciones: de forma específica, se analizan las ventajas de las soluciones basadas en PHP con respecto a las basadas en ASP.NET. La bibliografía y las distintas fuentes consultadas muestran una clara tendencia en favor de las primeras, tanto por la mayor cantidad de soluciones de desarrollo disponibles como por el carácter gratuito de muchas de ellas.

El desarrollo de *AulaWeb 2.0* es un proyecto de cierta envergadura, por lo que ha sido conveniente separarlo en varios módulos, dependiendo del tipo de usuarios que van a interactuar con la plataforma: Administrador, Profesor y Alumno. En el presente trabajo se desarrollará el **módulo de Alumno** que permite la interacción de los alumnos con los distintos contenidos, su descarga, así como el acceso a los foros de comunicación de que disponen las asignaturas. Para su realización se ha mantenido un contacto muy cercano con los desarrollos de otros módulos, especialmente el referente al gestor de contenidos, que establece el diseño y la interfaz del alumno y algunas tablas de la Base de Datos. Además, como ocurre con los restantes módulos del proyecto *AulaWeb 2.0* el desarrollo del módulo de alumno ha tenido en cuenta las sugerencias y propuestas de los profesores y de los restantes usuarios potenciales del sistema. Estas sugerencias fueron previamente clasificadas y analizadas.

El módulo de alumno incluye la creación de las correspondientes interfaces para el acceso a las distintas funcionalidades por parte de los alumnos usuarios de la herramienta. Asimismo, se ha diseñado y desarrollado la interacción con la base de datos correspondiente. Como en el caso de otras secciones de esta plataforma, se ha buscado un desarrollo modular, de forma que sea posible modificar y actualizar los distintos elementos de la manera más simple.

De forma específica, este proyecto se centra en la creación de un sistema que permite la interacción entre profesores y alumnos, la realización de evaluaciones, el intercambio de archivos y la participación en foros. Se han diseñado e implementado los módulos y la Interfaz del Alumno. Concretamente, se ha desarrollado la sección de “Información” que comprende las subsecciones de datos del alumno, de la asignatura, de los grupos, observaciones y calificaciones. También se ha desarrollado en la sección de “Comunicaciones” el cuestionario y el foro. Además, se ha modificado e integrado el apartado “Contenidos” realizado en un Trabajo Fin de Grado anterior para el usuario Profesor.

Para lograr el objetivo del trabajo se han utilizado tecnologías *Open Source* como son: PHP, HTML, CSS, JavaScript y JQuery UI, como alternativa a ASP.NET, que se utilizó en la versión anterior de la plataforma. Estos lenguajes de programación se caracterizan por ser, además de abiertos, totalmente gratuitos y por estar ampliamente extendidos y utilizados, con lo que se facilita su mantenimiento.

En lo que se refiere al desarrollo propiamente dicho, éste ha tenido lugar en uno de los ordenadores del Laboratorio de Informática de la ETSII. Previamente, ha habido un periodo formativo de cuatro meses. En dicho periodo se han realizado tres cursos (HTML, CSS, PHP, JavaScript/JQuery) de una duración de 200 horas cada uno, en la escuela online “Seiscocos”. De forma complementaria, se ha asistido a la asignatura “Desarrollo de Webs Dinámicas” de la ETSII. Además, se ha instalado una versión de XAMPP en un ordenador convencional con Windows para realizar pruebas y depuraciones cuando no era posible o cómodo acceder a los ordenadores de la ETSII.

El funcionamiento del programa sigue el modelo Cliente-Servidor: parte del código interactúa con la base de datos. Otras partes están dedicadas a la funcionalidad y, finalmente, otros elementos se dedican a la interfaz.

La estructura del programa realizado está basada en una arquitectura modular. De este modo la propagación de los errores queda limitada y su depuración es más rápida. Por otra parte, en caso de ser necesario, la actualización que afectaría únicamente a los módulos que sean necesarios, lo que implica un mantenimiento más cómodo.

Como actividad previa a la realización de estos módulos ha sido necesario reorganizar las tablas y bases de datos existentes y se ha creado un formato común a todas ellas, eliminando aquellas que se repetían o, simplemente, no eran válidas.

Para facilitar la identificación de los módulos y mejorar la experiencia de usuario se ha diseñado un código de colores para diferenciar los tres interfaces. Así, la interfaz “Alumno” quedará definida por un color azul que se distingue del verde y amarillo - naranja de las interfaces “Administrador” y “Profesor”, respectivamente.

Finalmente, como elemento central de todo proyecto Software, se ha realizado una serie de pruebas para ver el correcto funcionamiento del programa. Se ha procurado que estas pruebas fueran lo más exhaustivas posibles para reducir la presencia de fallos y la necesidad de modificaciones ulteriores.

Para concluir este trabajo se realiza una valoración económica del proyecto. Como punto de partida, el trabajo recoge un resumen de las técnicas de valoración de proyectos software más frecuentes. En este proyecto, se ha optado por emplear el método COCOMO. En el trabajo se pasa revista a sus orígenes y a sus posibilidades, para concluir con la aplicación práctica considerando el desarrollo de la interfaz Alumno para *AulaWeb 2.0*.

Palabras clave:

*AulaWeb*, Desarrollos modulares, Herramientas *e-learning*, Teleenseñanza, Software abierto, Desarrollo JavaScript, Desarrollo PHP

Código UNESCO: 3310.99

<b>Agradecimientos .....</b>	<b>1</b>
<b>Resumen ejecutivo .....</b>	<b>2</b>
<b>1. Introducción .....</b>	<b>7</b>
1.1 Planteamiento .....	7
1.2 Objetivos.....	16
<b>2. Metodología de desarrollo.....</b>	<b>20</b>
2.1 Selección de la arquitectura.....	20
2.2 Requisitos funcionales.....	27
2.3 Requisitos técnicos .....	28
2.4 Diseño de la solución .....	28
2.5 Control de versiones.....	68
<b>3. Planificación y Programación .....</b>	<b>70</b>
3.1 Plan de trabajo.....	70
3.2 Diagrama temporal .....	71
<b>4. Valoración del proyecto.....</b>	<b>72</b>
4.1 Alternativas consideradas.....	72
4.2 COCOMO y técnicas de estimación paramétricas .....	74
4.3 Valoración económica.....	79
<b>5. Conclusiones y Líneas de Futuro Desarrollo .....</b>	<b>82</b>
5.1 Conclusiones Generales .....	82
5.2 Conclusiones por objetivos .....	82
5.3 Impacto y responsabilidad legal, ética y profesional .....	84
5.4. Líneas futuras .....	85
<b>6. Referencias.....</b>	<b>87</b>
<b>Glosario de términos .....</b>	<b>90</b>
<b>Índice de Tablas .....</b>	<b>92</b>

<b>Índice de figuras.....</b>	<b>95</b>
-------------------------------	-----------

## 1. Introducción

### 1.1 Planteamiento

Las herramientas informáticas de Tele-enseñanza (*e-learning*) y de interacción entre profesores y alumnos tienen cada vez más importancia en la pedagogía moderna. Estas herramientas disponen de una creciente flexibilidad y cada vez son más adaptables a los contenidos, necesariamente cambiantes, que los docentes ofrecen a sus alumnos. Además, de acuerdo con las modernas técnicas pedagógicas, sirven para incrementar la interacción entre alumnos y profesores.

La Unidad Docente de Informática Industrial del Departamento de Automática, Ingeniería Eléctrica y Electrónica e Informática Industrial de la Escuela Técnica Superior de Ingenieros Industriales de la Universidad Politécnica de Madrid (ETSII) es consciente, desde hace tiempo de la necesidad de actualizar y mejorar las herramientas informáticas de que disponen alumnos y profesores de la ETSII para interactuar entre sí y mejorar la experiencia educativa.

El departamento realizó una primera versión del sistema de *e-learning* para la ETSII en el año 1999. Este primer sistema, denominado *AulaWeb*, proporciona soporte organizativo a los departamentos y ayuda a los profesores en la organización de las asignaturas [aula1-6]. Además facilita el seguimiento de las actividades de las asignaturas por parte de los alumnos.

Sin embargo, las constantes nuevas necesidades pedagógicas y los desarrollos de nuevas herramientas informáticas, hacen aconsejable la realización de una nueva herramienta que aproveche estas facilidades y se adapte mejor a las necesidades de profesores y alumnos.

Este es el objeto de *AulaWeb 2.0*, cuyo objetivo de partida es la actualización de la solución existente, ejecutada sobre una plataforma ASP.NET para convertirla en una solución que se ejecuta en PHP /JavaScript sobre Linux, utilizando el modelo cliente servidor. Esta solución es mucho más versátil y económica.

Este trabajo se centra en el desarrollo de uno de los módulos en los que se ha dividido el desarrollo de *AulaWeb 2.0* el módulo de alumno.

El trabajo presenta, en primer lugar, una breve introducción a las plataformas de Tele enseñanza, que tienen una alta potencialidad tanto pedagógica como de mercado. A continuación se analiza la solución *AulaWeb 2.0*, considerando su arquitectura y alternativas de programación. Dentro del proyecto, se consideran con más detalle los objetivos, requisitos funcionales y técnicos y se describe la solución adoptada. Finalmente, se realiza una valoración económica empleando la técnica COCOMO. Previa a esta valoración, el trabajo incluye una discusión sobre las posibles alternativas de valoración que podrían utilizarse en un proyecto comercial.

### 1.1.1 Las plataformas de Tele-enseñanza

La aplicación de las tecnologías TIC (Tecnologías Informáticas y de Telecomunicaciones) a la enseñanza está suponiendo unos cambios muy importantes en la forma de realizar el contacto entre los profesores y los alumnos y los alumnos entre sí. Estos cambios no solo afectan a la forma de la comunicación, que es mucho más rápida y dinámica, sino que también influyen a la información propiamente dicha que se modifica y se adapta a las características de los alumnos y a sus propias idiosincrasias y formas de aprendizaje.

En los últimos años están apareciendo multitud de plataformas de tele-enseñanza, hasta el punto que muchos expertos consideran que esta forma comunicación entre profesores y alumnos va a suponer una revolución en las formas pedagógicas y puede llegar a cambiar el concepto de la universidad como elemento central del aprendizaje [eco1][rit].

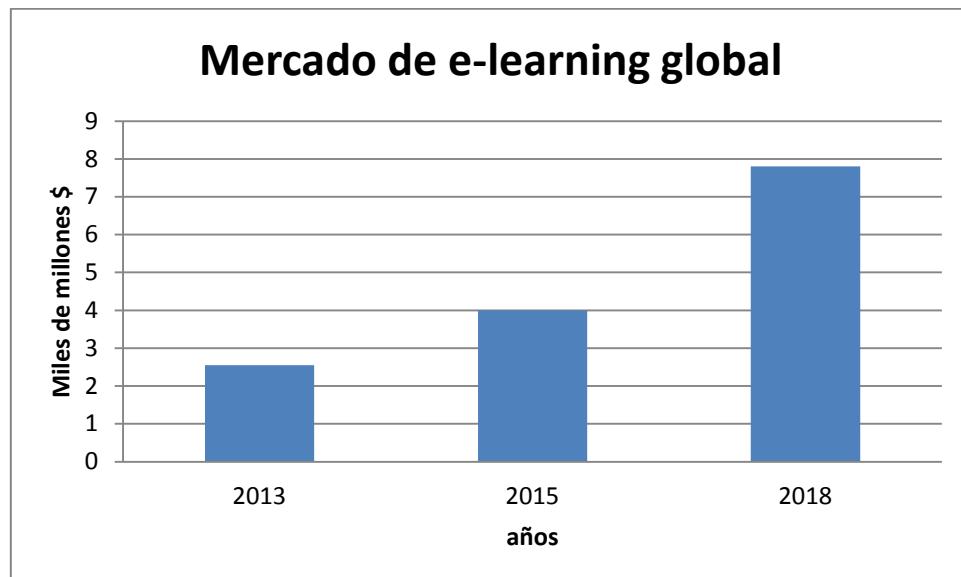
En muchas universidades, las herramientas de tele-enseñanza constituyen una parte central del proceso educativo y están plenamente integradas en el mismo. Como ejemplo [eco1] en algunos cursos de la universidad de Nueva York, 30 minutos después de haberse concluido la lección, el curso, que ha sido grabado con 3 cámaras, está disponible en la red para que los alumnos matriculados puedan comprobar las dudas y repetir los tramos dudosos. Por otra parte, muchos profesores reconocen que, sobre todo en lo que se refiere a la educación superior, la tele-enseñanza puede suponer una modificación del rol del profesor que pasa a tener un papel de mentor, más allá de la mera impartición de unas clases, ya que la existencia de numerosos cursos on-line pueden hacer innecesario explicar las asignaturas a fondo en clase. En todo caso, aunque las técnicas de tele-enseñanza signifiquen un cambio importante, parece claro que el contacto directo entre alumnos y profesores y alumnos entre sí es esencial para el proceso educativo. Además, muchos profesores señalan que los cursos exclusivamente virtuales corren el riesgo de que los alumnos no aprendan bien los temas básicos y, en el caso de los exámenes a distancia, se menciona el alto riesgo de que los alumnos puedan copiar el material.

Por otra parte, desde el punto de vista económico, el desarrollo de aplicaciones y programas informáticos relacionados con la tele-enseñanza está creciendo de una manera extremadamente rápida. Así, según datos de [elear][global] el mercado de las aplicaciones y plataformas de tele-enseñanza alcanza en 2015 el valor de 107 miles de millones de dólares, con una tasa de crecimiento anual que, desde 2010 se encuentra alrededor del 9,2%.

Los analistas de mercado indican que el crecimiento de este tipo de aplicaciones se centra, sobre todo, en los países en desarrollo (con tasas de crecimiento anual de 55% en la India, China 52%, pero también es muy importante en Latinoamérica, por ejemplo Brasil 26% o Colombia con un 20%). Este mercado resulta de especial interés para las empresas y universidades españolas y podría ser un campo de expansión de este sistema o de otros similares al que constituye este proyecto.

En lo que se refiere a los sistemas LMS<sup>1</sup> (*Learning Management Systems*), mercado al que corresponde la aplicación *AulaWeb*, el mercado alcanzó la cifra de 2 550 millones de dólares en 2013 con una tasa de crecimiento de 25,2%. Otros mercados que pueden ser muy significativos son el mercado de las aplicaciones de aprendizaje para dispositivos móviles, que se espera que alcance 12 200 millones de dólares en 2017, especialmente en EE.UU, Japón y Corea, China y en la India. También se espera un crecimiento muy importante en los denominados MOOC (*Massive Open Online Courses*). Según los datos de la *E-learning Industry Association* [elear], 8% de las compañías usan ya MOOC para la formación de su personal. Estos cursos MOOC también se utilizan para la selección de aspirantes (350 compañías americanas colaboran con *Coursera* y *Udacity* para la selección de personal y Google ha inscrito a 80000 trabajadores en cursos de *Udacity*, especialmente en cursos relacionados con programación de aplicaciones en web). Es interesante notar que el mercado de aplicaciones y servicios de *e-learning* se dirige, en gran medida, a las grandes corporaciones que lo van a utilizar para cursos internos.

En la figura 1 se incluyen algunos de los datos más significativos, según la referencia [elear] referentes al mercado de las aplicaciones de Tele-enseñanza.



**Figura 1. Datos del mercado Tele-enseñanza (elaboración propia a partir de datos de *E-learning Industry Association*)**

En definitiva, estos datos muestran que el futuro de las aplicaciones de Tele-enseñanza es muy prometedor y que pueden llegar a ser productos especialmente atractivos para potenciales clientes fuera de la ETSII. Todos estos aspectos avalan la conveniencia de adaptar

---

<sup>1</sup>

Un LMS o Learning Management System es un sistema de gestión de aprendizaje, es un software instalado en un servidor web que se emplea para administrar, distribuir y controlar las actividades de formación no presencial de una institución u organización

y mejorar las citadas herramientas y hacerlas más flexibles. Para ello, es muy conveniente que se utilicen herramientas software avanzadas, modernas y flexibles, tal y como se desarrolla en el siguiente apartado.

### 1.1.2 Código abierto

Cuando se aborda el desarrollo de nuevas soluciones informáticas, como es el caso de la nueva herramienta *AulaWeb* 2.0, es preciso tener en cuenta las nuevas tendencias de tecnología Software. En este caso, merece la pena comentar brevemente alguno de los aspectos que pueden ser relevantes para estos nuevos desarrollos y señalar las ventajas que suponen así como sus posibles dificultades, de forma que pueda prepararse un plan de actuación flexible y con la adecuada proyección de futuro. De todas ellas, la más relevante para este trabajo es la utilización de tecnologías de código abierto.

La utilización de código abierto se está convirtiendo en una tendencia cada vez más frecuente en el desarrollo de aplicaciones software puesto que supone una serie de ventajas de fiabilidad y ayuda a asegurar la continuidad de los desarrollos.

En todo caso, y como punto de partida, es conveniente clarificar algunas confusiones que pueden existir sobre el significado de los términos relacionados con el código abierto. De todas ellas, la más frecuente es la confusión entre los términos “código abierto” y “código gratuito”.

La definición de código abierto (“open software” según la organización *Open Source* [open]) es la siguiente: *"open source" refers to something that can be modified and shared because its design is publicly accessible.*

Esta definición no aplica exclusivamente al software. Incluye también a módulos hardware (plataformas de desarrollo hardware) e incluso a diseños gráficos o, por extensión, a obras de arte como pinturas, murales, dibujos o creaciones literarias como novelas o cuentos. El tema es muy amplio y controvertido. Por ejemplo, las novelas abiertas pueden ser modificadas por cualquiera, adaptándolas y modificándolas. Lo mismo ocurre con la música.

En lo que se refiere a las plataformas de hardware abierto ésta es una tendencia cada vez más importante. Como ejemplo, la lista de plataformas hardware de código abierto [ohw] incluye más de 40 plataformas distintas, en áreas que van desde el sistema de radioaficionados hasta la robótica. Entre las más conocidas están las plataformas *Arduino* [ard] y *Raspberry Pi* [rasp] que constituyen, en este momento, uno de los elementos básicos de muchas aplicaciones industriales y, desde luego, son frecuentemente utilizadas para la construcción de prototipos y para la enseñanza.

Concretando el término empleado con aplicaciones software, lo que en inglés se conoce con las siglas OSS [oss] (*Open-Source Software*) se refiere a un *código software que está*

*disponible para su estudio, modificación y distribución para cualquier persona o entidad interesada.*

Como se observa, la definición no entra en si esta disponibilidad es o no gratuita, ni las condiciones en que se pueda realizar esta modificación ni los derechos que tiene el desarrollador sobre la misma.

Una definición alternativa, proporcionada por GNU [gnu], que es una organización que ha desarrollado un sistema operativo de código abierto y gratuito y que es una autoridad de “facto” en la difusión de licencias de utilización y modificación relacionadas con el software abierto es la siguiente [gnu]:

*Software libre es aquel que se suministra con autorización para que cualquiera pueda usarlo, copiarlo y/o distribuirlo, ya sea con o sin modificaciones, gratuitamente o mediante pago. En particular, esto significa que el código fuente debe estar disponible*

El término de software abierto comenzó a popularizarse a partir de la publicación en 1997 del libro “La catedral y el bazar” por E. Raymond. Este libro fue muy comentado y fue un factor significativo en la decisión de Netscape de liberar (convertir en abierto) el software del famoso navegador Netscape (antecesor del Mozilla y otros navegadores como Seamonkey, Thunderbird y KompoZer).

Como consecuencia de ello, Raymond y otros crearon la *Free Software Foundation FSF*. Esta fundación tenía también una importante dimensión de activismo social y no resultaba atractiva comercialmente por lo que el término fue redenominado “Open Software” o “software Abierto”. Este nuevo término fue adoptado por la editorial de libros Tim O'Reilly, y por desarrolladores como Linus Torvald que crearon en 1998 la *Open Source Initiative*. Aunque muchas grandes empresas se opusieron inicialmente a esta iniciativa; por ejemplo Microsoft hizo un comunicado en 2001 que indicaba que el software abierto era un enemigo de la propiedad intelectual y un peligro para el negocio. Sin embargo, otras empresas como IBM, Oracle y Google aceptaron el concepto y concluyeron que no significaba un peligro para los desarrollos en software y los beneficios. Progresivamente, las ideas del software abierto fueron ganando aceptación incluso para sus primitivos detractores.

Como se indica, un aspecto fundamental del software abierto son las condiciones de utilización que se establecen en las licencias. Existe una gran variedad de licencias de Software abierto [tux][wilic][gnu]. Las licencias establecen derechos del propietario del software al usuario final sobre las copias del programa, los límites en la responsabilidad por fallos, el plazo de cesión de los derechos, el ámbito geográfico de validez de la licencia e incluso pueden establecer determinados compromisos del usuario final hacia el propietario, tales como la no cesión del programa a terceros o la no reinstalación del programa en equipos distintos al que se instaló originalmente.

La lista de opciones de licencias puede ser muy extensa, según el grado de permisividad que tiene el propietario original. Las más permisivas establecen muy pocas limitaciones al desarrollador final. Entre estas licencias están las licencias Apache, la licencia BSD o la

licencia *MIT*. Las licencias de utilización del programa PHP, utilizado en este proyecto, se encuentran también en este grupo.

Otras licencias, denominadas *robustas* establecen algunas restricciones. A su vez, estas restricciones pueden ser *fuertes* o con *copyleft* fuerte, contienen una cláusula que obliga a que las obras derivadas o modificaciones que se realicen al software original se deban licenciar bajo los mismos términos y condiciones de la licencia original. Entre estas licencias se encuentran las licencias *GNU o Affero*. Otras licencias denominadas de código abierto *débiles*, permiten que las obras derivadas que se puedan realizar de él puedan ser licenciadas bajo otros términos y condiciones distintas. Entre estas licencias están las licencias de *Mozilla*.

Por el contrario, las licencias de código cerrado no permiten modificar, copiar o distribuir el software de formas no especificadas en la propia licencia, que regula el número de copias que pueden ser instaladas e incluso los fines concretos para los cuales puede ser utilizado. La mayoría de estas licencias limitan fuertemente la responsabilidad derivada de fallos en el programa.

Finalmente las licencias de *dominio público* permiten el uso, copia, modificación o redistribución con o sin fines de lucro.

El término *Copyleft*, que aparece en muchas licencias, se refiere a que cualquiera que distribuya el programa con o sin cambios tiene que dar la libertad al usuario de poder volverlo a distribuir y/o modificarlo más si lo desea.

La figura 2 [gnufi] muestra un resumen de las diferentes posibilidades que plantea el software Abierto.

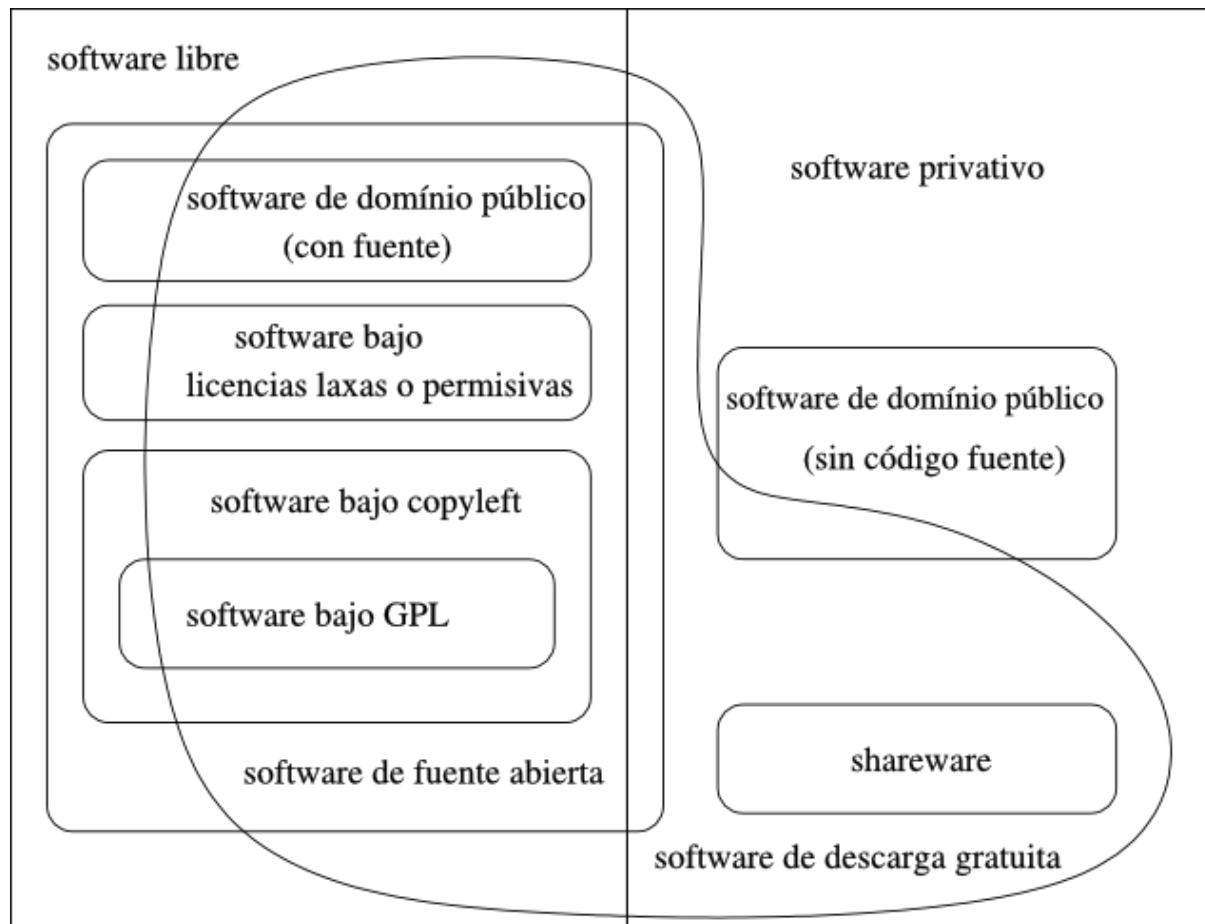


Figura 2. Distintas opciones del Software Abierto (fuente: Asociación GNU)

En lo que se refiere al coste económico, muchas veces se puede descargar gratuitamente un programa aunque no se pueden descargar las fuentes pero sí se puede usar el programa libremente. En este caso es una licencia *Freeware*: el programador no cobra licencia pero el código fuente no está disponible. Distinto es el caso de la licencia *Shareware* donde el usuario tiene que pagar por la licencia a pesar de tener la posibilidad de evaluarlo gratuitamente durante un tiempo.

Puede ser aclaratoria la figura 3 [linux] en la que se comparan, de una manera algo general, las posibilidades de los usuarios entre los tres tipos de software más relevantes para este trabajo.



Figura 3. Comparación entre Software libre, Open Source y privado. (fuente: blog.desdelinus.net)

En todo caso, aunque las condiciones de utilización pueden variar muchísimo, la idea central del código abierto tiene cada vez más partidarios ya que supone muchas ventajas tanto en reducción de costes de desarrollo como en continuidad de los mismos. Según un estudio frecuentemente citado [wikios] los desarrollos en open software suponen un ahorro a los consumidores de 60 miles de millones de dólares anuales. Otro de los informes más conocidos es el realizado por CENATIC [cenatic] que es más optimista y cifra el ahorro para la economía de la UE gracias al SW abierto en, al menos, 114.000 millones de € al año.

Según el citado informe de CENATIC un porcentaje considerable del software desarrollado es actualmente de código abierto, un hecho que se aplica tanto a productos de software de desarrollo interno como a productos desarrollados y contratados externamente.

Por su parte, el informe ONTSI [ontsi] de las TIC en las empresas españolas indica que más de la mitad de las microempresas (el 51,4%) utiliza alguna tipología de software de código abierto.

Por otra parte, el código abierto se utiliza cada vez más por los gobiernos y entidades internacionales. Como ejemplo, la UE busca promover su utilización y que el Software desarrollado en sus proyectos corresponda a estas reglas [EC1], tal y como se recoge en la recomendación que recoge los objetivos de la Comisión en lo que respecta a la utilización de Software. Dicha recomendación se refleja en la figura 4.

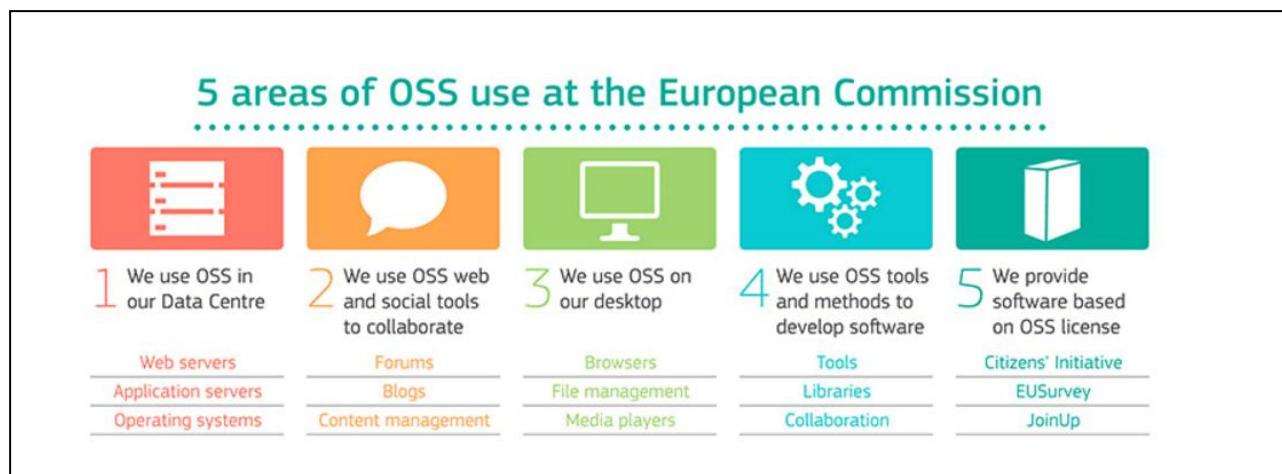


Figura 4. Uso del software abierto por la Comisión Europea (fuente EC)

Como resumen, la tabla 1 muestra las ventajas del código abierto:

Tabla 1. Ventajas y posibles dificultades del Código Abierto (elaboración propia)

Ventajas	Possibles problemas
<i>Rentabilidad para los desarrolladores:</i> Se genera gran cantidad de Código Fuente reutilizable del que nuevos programadores se vuelven a aprovechar, reduciéndose los costes de desarrollo.	No es adecuado comercialmente (al ser gratuito) si no se obtiene dinero por otra fuente, como por ejemplo por el lado del soporte y del mantenimiento.
<i>Rentabilidad para las empresas:</i> Evita los problemas que origina el utilizar productos que viene de una única fuente. Si la empresa que fabrica el código desapareciese, se encontraría con un código obsoleto, mientras que de esta forma se puede contratar a otra empresa.	Problemas de seguridad: permite a cualquier persona con conocimientos de programación contribuir al código de software y tal vez introducir trampas o errores
<i>Rentabilidad para los estados:</i> A la hora de comprar licencias para la administración y demás. No tiene que depender de una única empresa privada que le proporcione el código.	Puede tener los mismos problemas de seguridad indicados para empresas. Estos problemas pueden ser más graves por comprometer la seguridad nacional.

## 1.2 Objetivos

### 1.2.1 Objetivos generales

El objetivo del proyecto es el diseño y desarrollo del **módulo del Alumno** de la nueva plataforma *Aulaweb 2.0*, utilizando herramientas de código libre. Así como, integrar la sección “Contenidos”, de un Trabajo Fin de Grado anterior, en dicho módulo.

### 1.2.2 Objetivos específicos

Este proyecto se centra en el desarrollo del **módulo de alumno**. Las especificaciones de este módulo, así como su funcionamiento general, están definidas por las características generales del proyecto. De esta forma, las características más significativas establecidas por la dirección del proyecto para este módulo se resumen en la tabla 2:

Tabla 2. Objetivos específicos del módulo Alumno (elaboración propia)

<i>Característica</i>	<i>Condiciones del problema establecidas por la dirección del proyecto</i>
<b>1. Diseño del módulo Alumno</b>	Debe incorporar todas las funcionalidades que tenía en la anterior <i>Aulaweb</i> , eliminando aquellas que no se utilizan.
<b>2. Diseño y modificación de la Base de Datos gestionada por el módulo del Alumno.</b>	Puesto que la BB.DD. va a mantenerse se debe proceder a la unificación de todos los identificadores y que esté libre de redundancias
<b>3. Diseño de la interfaz de acceso del alumno al nuevo módulo, sus hojas de estilo y su formato.</b>	El color que definirá dicho módulo será el azul y la apariencia debe ser similar a la de los módulos Profesor y Administrador.
<b>4. Desarrollar utilizando herramientas de código libre cada una de las secciones y subsecciones de que consta el módulo de Alumno</b>	Se propone el uso de PHP y JavaScript.
<b>5. Desarrollar una nueva interfaz más “amable” de utilización por parte del alumno</b>	Incorporación de botones en lugar de pestañas en las distintas secciones y subsecciones.

<b>6. Construcción de nuevas tablas en la Base de Datos</b>	Las necesarias para el funcionamiento del módulo Alumno
<b>7. Integrar la sección “contenidos” del módulo Profesor.</b>	Modificando las funcionalidades que eran privativas del profesor.
<b>8. Integrar el módulo Alumno en el contexto global de la plataforma</b>	El módulo Alumno debe estar plenamente integrado en la estructura <i>AulaWeb2.0</i> que está en desarrollo y cuenta ya con algunos módulos operativos. En todo caso, al tratarse de una estructura modular, debe estar claramente diferenciado para permitir su depuración y eventual reparación de forma independiente.

### 1.2.3 Estructura de la memoria

Este trabajo sigue la estructura habitual en los proyectos de fin de grado. Consta de tres elementos principales.

- a) Metodología de desarrollo
- b) Implementación
- c) Valoración

A continuación, se describen los elementos con un cierto detalle.

**Introducción:** Se ha realizado una introducción, algo más detallada de lo usual, en la que se ha recogido la historia del proyecto, su justificación: esencialmente la conveniencia de efectuar una programación en código abierto para mejorar las posibilidades de evolución de la herramienta. La introducción incluye un breve análisis de las plataformas de Tele enseñanza y su importancia pedagógica y económica. También se realiza un rápido repaso de las características del software de código abierto y el software libre. La introducción sirve para enmarcar los objetivos, tanto generales como específicos de este proyecto y justifican la necesidad del proyecto, la nueva versión de AulaWeb, así como las opciones de diseño, esencialmente la utilización de código abierto para la realización de una herramienta fácilmente extensible y de una gran utilización, con unos costes bajos para el departamento y la universidad.

La introducción incluye también, de forma sintética, los objetivos generales y específicos del proyecto así como un resumen estructurado del documento (esta sección).

**Metodología:** Una vez concluida esta introducción, el capítulo 3 incluye la metodología de desarrollo del proyecto. En primer lugar, se define la arquitectura y los lenguajes de programación que van a utilizarse. La selección de la arquitectura cliente servidor y el uso de JavaScript y PHP son, en gran parte, el resultado de los otros proyectos de desarrollo de AulaWeb 2.0 en los que se enmarca este trabajo. Aun así, se ha considerado conveniente repasar y justificar plenamente las razones de la elección. En las siguientes secciones del capítulo, se describen los requisitos funcionales, resultado de las reuniones con las conversaciones con los usuarios de la herramienta, así como los requisitos técnicos, para terminar, con el diseño de la solución que va a adoptarse en el módulo Alumno, para asegurar la plena integración de este módulo en los restantes elementos del proyecto AulaWeb 2.0. Se define la arquitectura general, así como la arquitectura Software, la estructura básica y requerimientos de la base de datos, incluyendo una descripción de los elementos constitutivos: datos de Alumnos y asignaturas, profesores, cuestionarios y observaciones.

Finalmente se describen los elementos de la interfaz de alumno y otros aspectos relacionados con la estructura del proyecto y sus versiones. El diseño de la solución constituye el elemento central del trabajo e incluye una descripción de las pantallas que aparecen en el programa desarrollado, las distintas acciones previstas así como los resultados y las respuestas a los errores. En la parte correspondiente a las bases de datos se describen, en un cierto detalle, los elementos involucrados en el módulo alumno y los distintos campos que se utilizan. De forma específica se ha desarrollado la sección de “Información” que comprende las subsecciones de datos del alumno, de la asignatura, de los grupos, observaciones y calificaciones. También se ha desarrollado la sección de “Comunicaciones”, el cuestionario y el foro. Además, se ha modificado e integrado el apartado “Contenidos” realizado en un Trabajo Fin de Grado anterior para el actor Profesor. Como se indica en los requerimientos se han utilizado tecnologías *Open Source* como son: PHP, HTML, CSS, JavaScript y JQuery UI. El funcionamiento del programa sigue el modelo Cliente-Servidor: Parte del código interactúa con la base de datos. Otras partes están dedicadas a la funcionalidad y, finalmente, otros elementos se dedican a la interfaz.

La estructura del programa sigue una arquitectura modular para facilitar la depuración y la eventual actualización de los módulos, caso que sea necesario. También se han reorganizado las tablas y bases de datos incluyendo un formato común a todas ellas, eliminando elementos repetitivos o erróneos.

**Implementación:** En el siguiente capítulo, se describen los elementos de implementación que se han descrito en el capítulo anterior, incluyendo la estructura del proyecto, el diagrama temporal GANTT. Esta parte permite, de un vistazo, tener una idea de los elementos principales

**Valoración económica:** El capítulo 5 incluye una valoración económica. Dada la importancia de este último aspecto y su importancia pedagógica, se ha realizado un análisis justificativo del método elegido, así como un análisis algo detallado de las diferentes posibilidades que existen para su realización.

**Conclusiones:** El apartado correspondiente a las conclusiones repasa el cumplimiento de los objetivos

**Líneas futuras:** finalmente, se mencionan las posibles líneas de trabajo futuras del proyecto AulaWeb 2.0. Se espera que este desarrollo, junto con los otros trabajos que han contribuido al sistema, pueda ser la base de un sistema que facilite la interacción entre los profesores y los alumnos de la ETSII y sea un elemento pedagógico y de referencia útil y práctico.

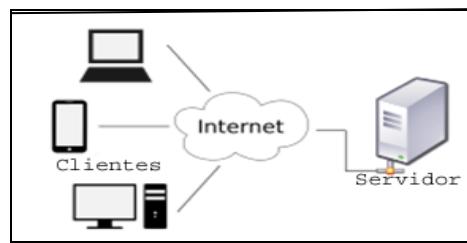
**Referencias:** Se recogen, en este apartado, las referencias utilizadas para elaborar este trabajo

## 2. Metodología de desarrollo

### 2.1 Selección de la arquitectura

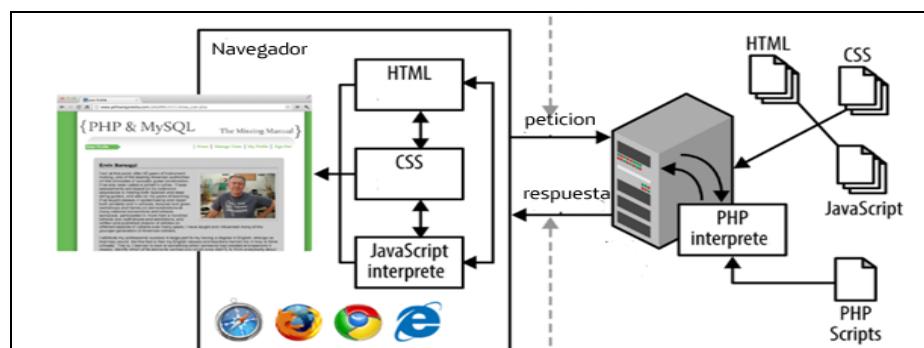
El sistema *AulaWeb 2.0* debe seguir la misma estructura cliente-servidor [clie1] que se utiliza en *AulaWeb* ya que es una condición establecida en los objetivos generales.

Este modelo es aplicable cuando se desea establecer una comunicación entre varios ordenadores, tal y como indica la figura. En este caso, hay un ordenador central, situado normalmente en una empresa o, como en este caso, en un centro educativo. Este ordenador, denominado servidor (*Server*) es el que contiene la información. Por otra parte, a este ordenador central acceden una serie de ordenadores, generalmente más pequeños, denominados clientes (*Clients*). Estos ordenadores pueden estar situados en lugares físicamente alejados, tal y como se muestra en la figura 5.



**Figura 5. Modelo de comunicaciones entre ordenadores (fuente Wikipedia commons)**

El modelo cliente servidor se popularizó especialmente con el desarrollo de Internet, ya que es el que se emplea para la comunicación en esta red. En este caso, la figura, presenta una descripción más detallada. En la figura 6 [clie1] se han indicado los lenguajes de programación que, cada vez más, se utilizan en este tipo de estructuras y que son los que se utilizan en *AulaWeb 2.0*.



**Figura 6. Sistema Cliente Servidor (versión general y desarrollo en Apache, PHP/LAMP) (fuente: [clie1])**

La arquitectura cliente servidor puede implementarse utilizando diversos programas y plataformas. En *AulaWeb* inicialmente se utilizó ASP.NET pero, como se ha comentado, esta solución presenta una serie de limitaciones. En la nueva versión, se mantiene la estructura cliente servidor pero se va a utilizar una arquitectura LAMP PHP/HTML.

La solicitud de información comienza desde el cliente, que solicita información utilizando un navegador (que es un programa especialmente diseñado para ejecutar el modelo cliente servidor, como *FireFox*, *Chrome* o *Explorer*) realiza los siguientes pasos:

1. El usuario escribe la dirección de internet deseada. Por ejemplo: [www.ejemplo.com/AulaWeb/main.php](http://www.ejemplo.com/AulaWeb/main.php) en el buscador del navegador.
2. El navegador contacta con ese ordenador, cuya dirección es [www.ejemplo.com](http://www.ejemplo.com) solicitándole la página *AulaWeb/main.php*.
3. El servidor (un programa como *Apache*, por ejemplo) se ejecuta en el ordenador con el que se está contactando, solicita al intérprete PHP (u otro programa similar), información de la página Web.
4. El intérprete en el servidor lee el archivo de la unidad de disco.
5. El intérprete ejecuta los comandos en *main.php*, posiblemente intercambiando datos con un programa de base de datos como *MySQL*.
6. El intérprete recibe la salida del programa y se lo devuelve al servidor como respuesta.
7. El servidor devuelve el contenido de la página que recibe a través de Internet como respuesta la petición del navegador.
8. El navegador muestra la página en pantalla.

Esta solución de arquitectura, con los programas indicados, una de las más frecuentes para implantar el modelo cliente servidor. De todos modos, el uso de PHP no es la única solución posible. En el siguiente apartado se analizan las alternativas existentes y se justifica la elección de PHP para el desarrollo de varios elementos del módulo Alumno.

## 2.1.1 Selección del lenguaje de programación

Se analizan a continuación las plataformas y lenguajes de programación que pueden utilizarse para implantar el modelo cliente servidor, para justificar la selección de PHP/HTML en la nueva versión *AulaWeb 2.0*

**ASP. Net.** Como se ha indicado, esta es la tecnología anteriormente utilizada en *AulaWeb*. Es un *framework* (conjunto de programas) para aplicaciones web, desarrollado y comercializado por Microsoft. Se utiliza para construir sitios web dinámicos, aplicaciones web y servicios web XML. Apareció en enero de 2002 con la versión 1.0 del .NET Framework, y es la tecnología sucesora de la tecnología Active Server Pages (ASP). ASP.NET permite utilizar código ASP.NET usando cualquier lenguaje admitido por el .NET Framework.

En realidad, ASP.NET es ya un proyecto de software abierto. De hecho, ASP es una plataforma OSS según indica Microsoft en su página web [asp]. Concretamente es una licencia Apache 2.0 que GNU reconoce como compatible con GPL. Sin embargo, la utilización de ASP.NET suele estar ligada a el uso de servidores de Microsoft por lo que requieren el pago de una licencia.

En este proyecto se plantea substituir ASP por PHP, por lo que, a continuación se resumen alguna de las características de esta tecnología:

**PHP (con HTML, CSS):** Este lenguaje fue creado en 1994. El nombre es una referencia recursiva a su objetivo fundamental. *PHP: Hypertext Preprocessor* Es un lenguaje de programación que se utiliza en su mayoría para la creación de sitios Web, por lo que normalmente se ejecuta en un servidor Web al que acceden otros programas usuario utilizando el modelo cliente servidor para la comunicación con el ordenador central al igual que hace el sistema ASP.net. En la tabla 3 se enumeran las distintas ventajas que tiene PHP.

**Tabla 3. Ventajas de PHP (elaboración propia)**

<b>Ventajas de PHP</b>	<b>Comentarios:</b>
Es gratuito	No existen costes de licencia, soporte, mantenimiento, actualización o de cualquier otro tipo.
Es libre	Como proyecto de código abierto, PHP pone su código disponible a cualquiera que desee inspeccionarlo
Es multiplataforma	Se puede utilizar PHP con un servidor Web que ejecute Windows, Mac OS X, Linux, Solaris y muchas otras versiones de Unix. Además, si cambia el sistema operativo del servidor Web, por lo general no tiene que cambiar ninguno de los programas PHP.

Oculta su complejidad	PHP puede utilizarse tanto para grandes servidores como para pequeños sistemas. En ambos casos, el programa es capaz de manejar los recursos sin que el programador (o el usuario) sean conscientes de algunas de las operaciones.
-----------------------	--

Además del uso de PHP, la nueva versión de *AulaWeb* utilizará servidores Apache y bases de datos SQL. Por otra parte, otros módulos del programa utilizan JavaScript y JQuery que es una biblioteca que se utiliza en numerosos módulos de JavaScript.

**Apache:** Apache es un Servidor Web, es decir, un programa que permite que un ordenador conectado a la red sirva páginas Web a otros que las solicitan, por ej., mediante un navegador.

El proyecto Apache está gestionado por *The Apache Software Foundation* (creada en 1999), que es la encargada de sostener las versiones oficiales y coordinar los esfuerzos de la multitud de desarrolladores que la componen.

Apache es especialmente emblemático en aplicaciones de código abierto. Más del 56% de los servidores Web emplean Apache. Esto no es casual ya que forma parte del denominado sistema de publicación LAMP (Linux, Apache, MySQL, PHP). Este conjunto de programas de Código Abierto se está imponiendo en el mercado como una de las formas de gestionar contenidos dinámicos en Internet. En este caso, Apache es el encargado de interpretar instrucciones del lenguaje, por ej. PHP, y acceder a la base de datos (MySQL) en busca de contenidos.

**MySQL:** Es la base de datos relacional de tipo Código Abierto más popular de la comunidad, con más de dos millones de instalaciones censadas. Forma el cuerpo del conjunto LAMP (Linux Apache, MySQL, PHP), dotándolo de la infraestructura necesaria para el almacenamiento y gestión eficaz de la información.

Aunque su uso más popular es bajo el sistema operativo Linux, esta aplicación puede ejecutarse en una amplia variedad de plataformas.

**JavaScript:** Es un lenguaje de programación interpretado. Se define como orientado a objetos, basado en prototipos, imperativo y dinámico. JavaScript se utiliza principalmente en su forma del lado del cliente, implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas, aunque existe una forma de JavaScript del lado del servidor.

Tradicionalmente se venía utilizando en páginas web HTML para realizar operaciones y únicamente en el marco de la aplicación cliente, sin acceso a funciones del servidor. Actualmente es ampliamente utilizado para enviar y recibir información del servidor junto con ayuda de otras tecnologías como AJAX.

En la tabla 4 se enumeran las ventajas que presenta JavaScript.

**Tabla 4. Ventajas de JavaScript (elaboración propia)**

<i>Ventajas de JavaScript</i>	<i>Comentarios:</i>
Fácil de aprender, rápido y potente	Se puede empezar a trabajar con él desde el principio. Es ideal para agregar funciones rápidas a una página. Es un lenguaje muy potente de alto nivel, si bien no se puede hacer nada directamente a nivel de máquina. No necesita una fase de compilación como Java o C, sólo hay que crear el código y cargarlo. Utiliza una arquitectura orientada a objetos parecida a Java o C++.
Usabilidad	Es el lenguaje de programación que más se usa en la Web.
Reducción de la carga del servidor	Se puede hacer cargo de gran parte de las funciones del cliente de las cuales se encargaba el servidor.

**JQuery:** es uno de los complementos más usado en sitios Web, ya que facilita el desarrollo de aplicaciones del lado del cliente, en JavaScript, compatibles con todos los navegadores. JQuery no es un lenguaje, sino una serie de funciones y métodos de JavaScript. Por tanto es una librería que puede usarse opcionalmente. Anteriormente, los desarrolladores estaban obligados a discriminar entre los diversos navegadores, para ejecutar las versiones del código JavaScript que funcionaba en cada caso. Con la llegada de JQuery, ya no se necesita preocupar por el tipo de navegador: (*Explorer, Chrome, Firefox, etc*), sino que la propia librería ejecuta el código que sea compatible con el software del cliente que está accediendo a nuestra web.

#### Comparación PHP ASP.NET

Como se ha comentado, uno de las decisiones de diseño de *AulaWeb 2.0* es la substitución de las plataformas ASP.NET por soluciones LAMP con PHP. Como resumen de lo comentado en el apartado anterior la tabla 5, recoge algunos elementos de la comparación [aspv]. Quizá la más relevante es el menor precio de despliegue y la mayor popularidad.

**Tabla 5. ASP.net vs PHP (elaboración propia)**

<i>Característica</i>	<i>ASP.NET</i>	<i>PHP</i>
Escalabilidad y facilidad de mantenimiento	similar	similar

Velocidad y prestaciones Acceso a BB.DD.		Para acceso a BB.DD. SQL PHP parece ligeramente superior
Sistema operativo		PHP sobre Linux es algo más rápido que Windows OS usando NFTS (sistema de archivos de Windows)
Tamaño del programa		Los sistemas sobre Linux son más ligeros
Velocidad de cálculo	Cuando se precisa mucho cálculo (lo que no es frecuente) las aplicaciones basadas en C# son algo más rápidas. Además el lenguaje está compilado con lo que suele ejecutar más rápido	
Coste		PHP, Apache y MySQL son gratuitos. Esto es especialmente importante si la aplicación requiere más de una máquina
Soporte		La comunidad de desarrolladores de PHP es más numerosa
Tiempo necesario para despliegue	ASP normalmente requiere menos líneas de código	
Editores y herramientas	ASP utiliza <i>Visual Studio</i> que es muy versátil aunque necesita un tiempo de aprendizaje para su uso	Tiene una gran cantidad de sistemas de desarrollo, pero son menos potentes.
Plataformas	Precisa Windows	Puede ejecutarse en cualquier plataforma
Popularidad		Las plataformas LAMP (Linux, Apache, MySQL, PHP) son mucho más populares que las basadas en Windows servers
Facilidad de administración		La mayoría de los expertos opinan que es PHP sobre Linux es más sencillo

Como dato final, complementario, se incluye la tabla 6 con los datos de algunos “sites” de internet populares donde se comprueba la mayor popularidad y posiblemente las mayores posibilidades de evolución de las soluciones PHP.

**Tabla 6. Soluciones adoptadas para los servidores de algunas plataformas (elaboración propia)**

Sitio Web	Desde	Servidor de Plataforma	Lenguaje de Programación
Google.com	Noviembre 1998	Linux	C, Java, C++, PHP & MySQL
Facebook.com	Febrero 2004	Linux	PHP, MySQL & C++
Youtube.com	Febrero 2005	Linux	C, Java & MySQL
Yahoo.com	Agosto 1995	Linux	C++; C, Java, PHP & MySQL
MSN.com (propiedad actual de Microsoft)	Agosto 1995	Windows	ASP.net
Live.com (propiedad actual de Microsoft)	Agosto 2008	Windows	ASP.net
Wikipedia	Enero 2001	Linux	PHP & MySQL
Amazon.com	Octubre 1995	Linux & Solaris	C++, Java, J2EE
WordPress.com	Noviembre 2005	Linux	PHP & MySQL

En definitiva, las razones anteriores justifican la migración hacia la utilización de PHP sobre Apache/Linux en la parte del servidor de la nueva versión de *AulaWeb*. Como BB.DD. se va a utilizar MySQL que, como se ha indicado, está muy bien integrada con las soluciones LAMP.

## 2.2 Requisitos funcionales

La determinación de los requisitos de funcionamiento del módulo Alumno se realizó, como en los restantes elementos de la herramienta *AulaWeb 2.0* a través de una serie de reuniones y conversaciones con los usuarios, profesores y alumnos, de la herramienta actual. Como este módulo es el último en desarrollarse, gran parte de los requisitos están establecidos de ante mano por los otros módulos y, por tanto, dos elementos fundamentales para el módulo alumno son la interoperabilidad y la coherencia con los módulos profesor y administrador.

A modo de esquema, la tabla 7 se resume alguna de las características más significativas.

**Tabla 7. Requerimientos de la interfaz AulaWeb 2.0 (elaboración propia)**

<b>Requerimiento</b>	<b>Comentario</b>
<b>Interoperabilidad y coherencia con otros módulos</b>	<p>El módulo alumno debe funcionar con los otros dos módulos, acceder a la BB.DD. común y mantener un funcionamiento similar al de los módulos anteriores. De esta forma se asegura la facilidad de manejo y mantenimiento.</p> <p>La interfaz debe seguir la misma estructura que los restantes módulos de la herramienta, con una clave de colores diferente para facilitar el manejo</p> <p>El acceso a la BB.DD. no debe modificar los contenidos de la misma, salvo en lo que se refiere a aquellos datos de alumno, como la contraseña, que son específicos de este módulo.</p> <p>La aplicación debe, en todo momento, controlar el tamaño de las BB.DD. para no superar los tamaños previstos en el programa general.</p>
<b>Simplicidad</b>	<p>Crucial para que el usuario se sienta satisfecho y la utilice. De este modo se propone eliminar las múltiples pestañas y solapas que configuran los distintos apartados, sustituyéndolas por dos simples filas de botones colocados en la parte superior</p> <p>Se intenta que todo el contenido quede reflejado en la pantalla sin tener que usar demasiado el scroll.</p>
<b>Coherencia interna</b>	<p>Cada botón de cada apartado queda definido por un título que señala el contenido del mismo. De esta forma, se conoce con certeza dónde y qué guarda cada subapartado, por lo que se puede acceder con facilidad al contenido deseado sin que el usuario tenga que hacer ningún razonamiento complicado.</p>
<b>Eficiencia:</b>	<p>La interfaz debe tener un rendimiento elevado independientemente del hardware o software utilizado. Para ello, deberá disponer de unas velocidades de conexión del cliente y del servidor lo suficientemente elevadas.</p>
<b>Portabilidad</b>	<p>capacidad para ser soportada por cualquier plataforma o sistema operativo y que se pueda acceder a ella desde cualquier navegador</p>

	sin que cambie el aspecto de la interfaz
<b>Manejo de errores</b>	En caso de que se produzca algún error, o la información suministrada presente falta de coherencia, debe notificarse al usuario y no se deberá proceder a ningún cambio en la BB.DD.

Como guía central para el desarrollo de *AulaWeb 2.0*, se debe tener en cuenta que el objetivo será crear una interfaz alumno más económica basada en la de *AulaWeb 1.0* pero de manera crítica, haciendo modificaciones en el aspecto gráfico y en los contenidos de acuerdo a los nuevos requerimientos de los alumnos y profesores.

## 2.3 Requisitos técnicos

Como consecuencia de la discusión realizada en el apartado de selección de arquitectura, gran parte de los objetivos de la nueva versión de *AulaWeb 2.0* se refieren a aspectos técnicos de programación. Estos requisitos son comunes a diversos módulos de *AulaWeb 2.0* y han sido justificados en los apartados anteriores.

- a) Utilizar plataformas “*open source*”.
- b) El lenguaje en el lado del servidor será *PHP*.
- c) El lenguaje en el lado del cliente será *HMTL, CSS* y *JavaScript*.
- d) El servidor en el que se alojará será Apache.
- e) La aplicación debe ser de fácil manejo y apariencia similar a su predecesora.
- f) La base de datos será del tipo *MySQL*.
- g) La administración y mantenimiento de la base de datos *MySQL* se realizará a través de las herramientas de mantenimiento *phpmyadmin*
- h) La confirmación de eventos relacionados con acciones importante sobre la estructura de la base de datos se realizará a través de la herramienta “*open dialog*” de la librería “*JQuery\_UI*” escrita en *JavaScript*.

## 2.4 Diseño de la solución

### 2.4.1 Arquitectura general

Tal y como se ha indicado la nueva estructura general de la plataforma *AulaWeb 2.0* será de tipo modular. Este tipo de estructura implica la separación de todos los elementos en distintos módulos. Con esta estructura se permite que el mantenimiento y mejora de la plataforma en un futuro sea mucho más sencilla.

Los distintos módulos están relacionados pero son independientes unos de otros aunque pueden compartir información. La estructura finalmente utilizada, tal y como se ha adelantado en la introducción responde al esquema de la figura 7:

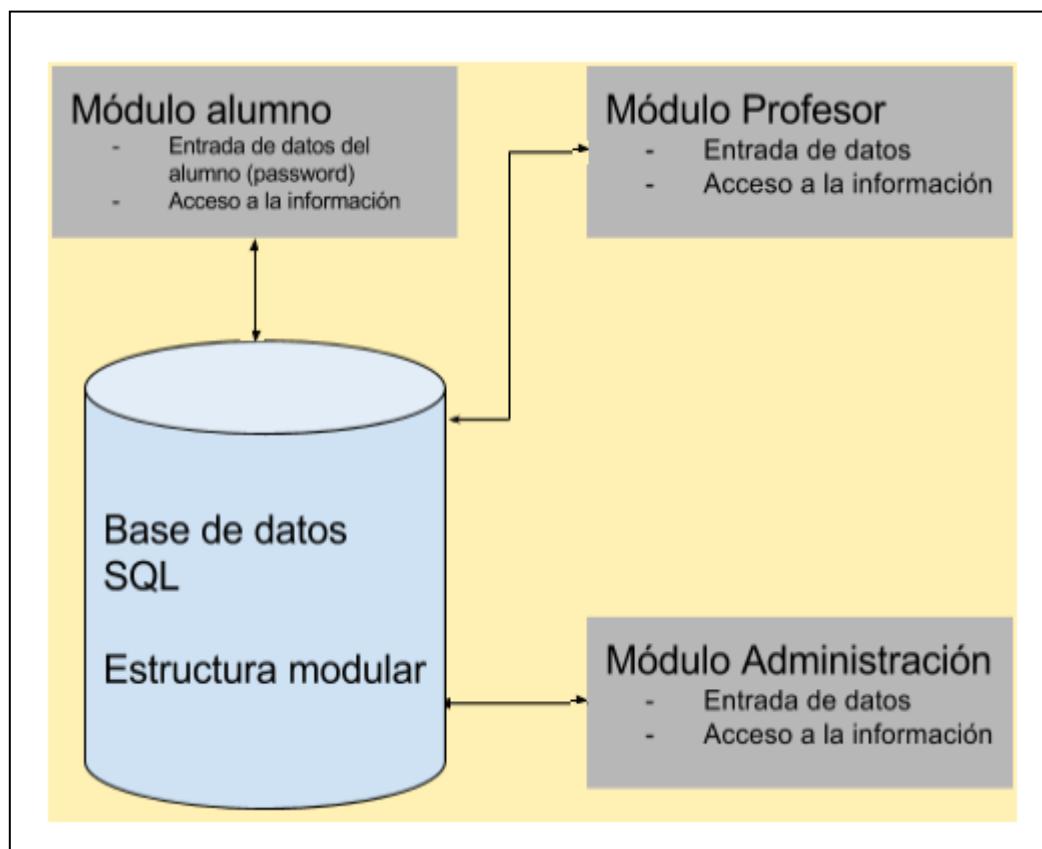


Figura 7. Estructura de programa AulaWeb 2.0. (Elaboración propia)

#### 2.4.2 Arquitectura del Software

De acuerdo con la estructura general y los requerimientos software descritos en los apartados anteriores, se adoptan los lenguajes de programación resumidos en la tabla

**Tabla 8. Arquitectura Software (elaboración propia)**

<i>Característica</i>	<i>Decisiones más significativas</i>
<b>Utilización de código abierto y gratuito</b>	Uso de PHP y JavaScript.
<b>La aplicación debe tener una apariencia similar a la existente</b>	Uso de JQuery en pestañas y desplegables para lograr una interfaz atractiva Se ha optado por colocar botones en lugar de pestañas para lograr una aplicación más orientada al usuario
<b>Arquitectura modular</b>	Cada módulo es independiente de los otros y se relacionan mediante lazos que si se cortan no se vean afectadas las funcionalidades
<b>Diseño de la Base de Datos</b>	Unificación del tipo de letra y coherencia en la denominación de los campos de la base de datos

En la figura 8 y 9 se indica la estructura final que tienen las distintas clases/subprogramas que constituyen el módulo alumno

Se indica en cada caso, con un código de colores

- a) Programa principal y módulos escritos en php (azul)
- b) Los elementos .css que se utilizan para formatos (verde)
- c) Las partes .js escritas en JavaScript (rojo)

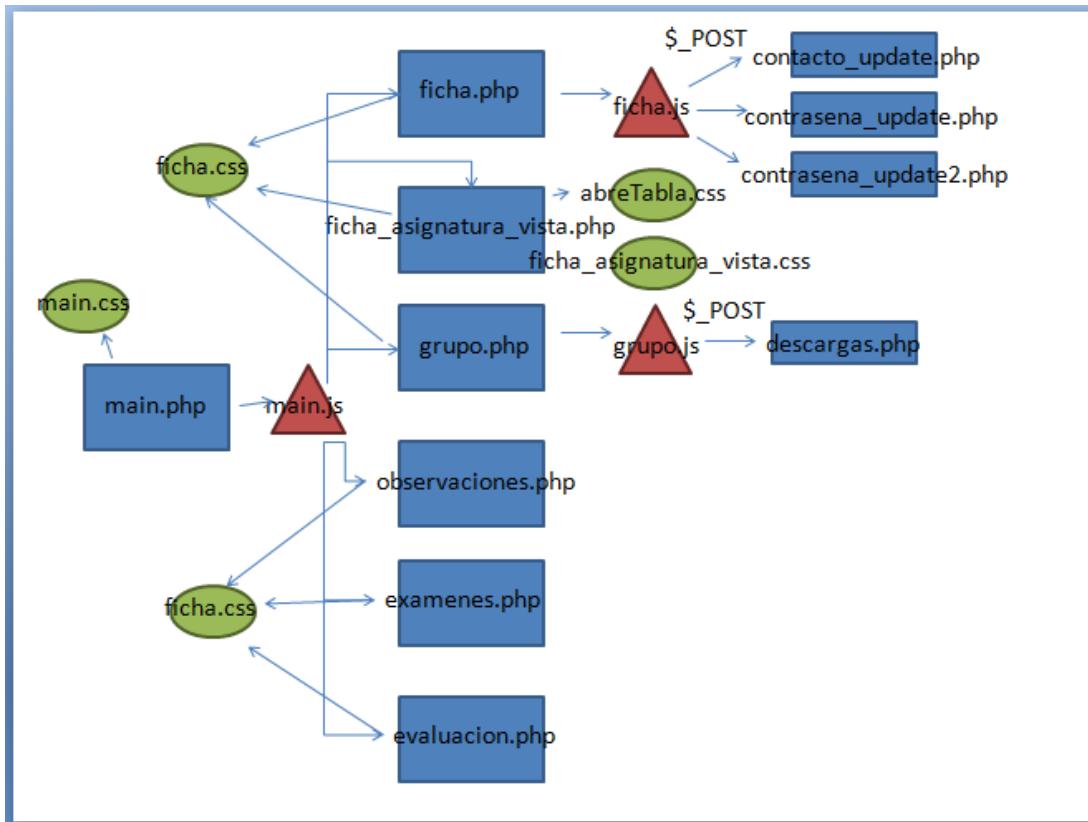


Figura 8. Esquema de la interfaz Alumnos. (elaboración propia)

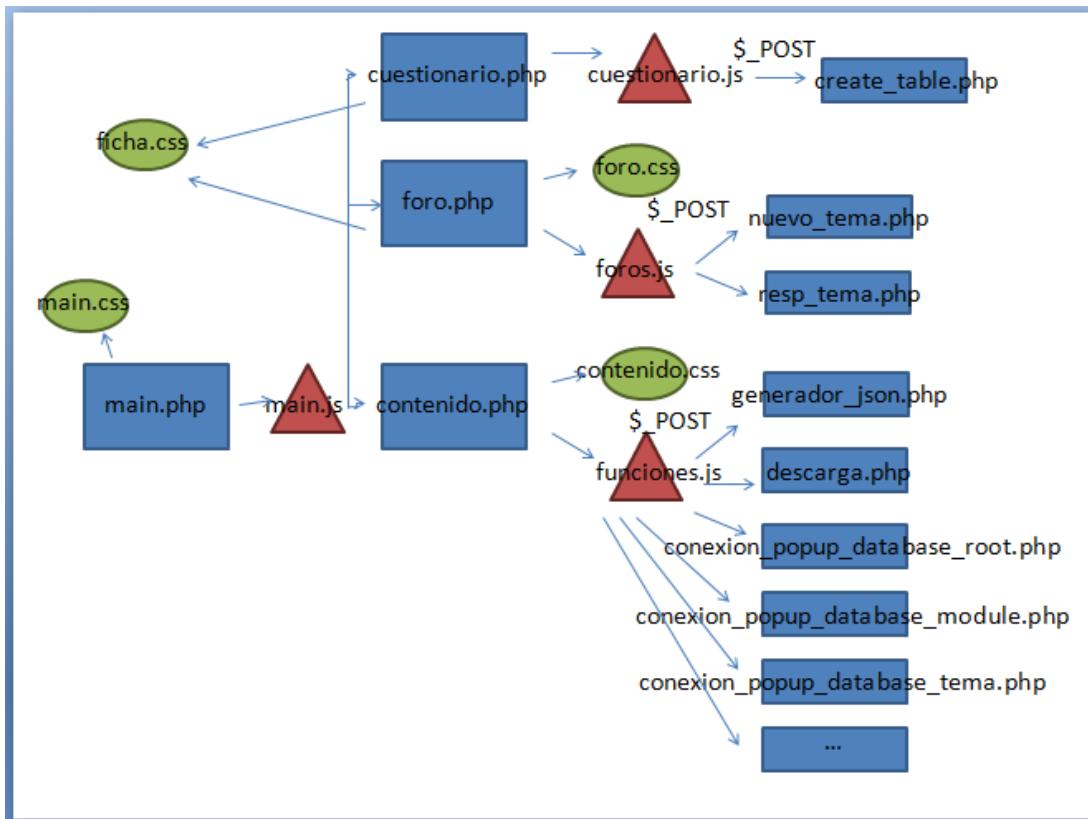


Figura 9. Esquema de la interfaz Alumnos. (elaboración propia)

### 2.4.3 Diseño de la Base de Datos

Como se ha indicado, la nueva versión de *AulaWeb*, va a utilizar una BB.DD. MySQL. La implementación de esta nueva BB.DD. representa una oportunidad de revisar y actualizar el funcionamiento de la BB.DD. existente.

La base de datos de la versión inicial de *AulaWeb* no incluía toda la funcionalidad requerida. Por tal motivo, fue necesario realizar una serie de actualizaciones de una manera poco estructurada, incorporando nuevos campos a medida que aparecían nuevos requerimientos. La estructura resultante estaba poco organizada, careciendo de homogeneidad en la escritura de entidades y atributos e incluso en la forma de presentarnos los datos, pudiendo aparecer, además, la información referente a una misma entidad dispersa en varias tablas. Por otro lado, existen diferentes maneras de nombrar un campo en distintas tablas que contiene la misma información de una entidad. Por ejemplo, el campo que hace referencia a una asignatura se denomina de manera diferente, por ejemplo: id\_asignat, ID\_Asignatura, ID\_asig,...

Se propone, por tanto, normalizar dichos campos e imponer su escritura en minúscula. Así como eliminar aquéllos que estén vacíos o no tengan ninguna utilidad.

Por este motivo, se ha realizado una nueva versión de la base de datos. Esta base de datos es el elemento central del modulo administrador pero incluye, además, las necesidades del modulo de interfaz Alumno.

Las nuevas tablas de la base de datos realizadas se listan en la tabla 9:

**Tabla 9. Modificaciones realizadas en la Base de Datos (elaboración propia)**

Tabla	Módulos
Alumnos	Asignatura
	Evaluaciones
	Horarios
	Grupos
	Grupos-asignatura
	Grupo-profesor
	Grupo-tipo
	Autoevaluación

	Exposiciones
	Prácticas
	Pruebas parciales (Peces)
	Trabajos en grupo (tr-Grupal)
	Trabajos individuales (tr-individual)
Alumnos y asignaturas	Alumnos
	Alumno y asignatura
	Estado
Profesores y asignaturas	Profesores
	Profesores y asignaturas
	Tutorías
Cuestionarios	Cuestionarios
	Respuestas a cuestionarios
Observaciones	Observaciones
	Foro
	Visualizaciones del foro

A continuación, se detallan las modificaciones y actualizaciones realizadas

## Tablas alumnos

### *Tabla Asignatura*

Realizada, inicialmente, para el módulo administrador, ha sido modificada para adecuarla al módulo Alumno, también. Recoge todos los atributos relativos al campo de asignatura que pudieran afectar tanto al alumno, como al administrador y como al profesor.

**Tabla 10. BB.DD. Tabla contenidos (elaboración propia)**

nombre	tipo	Descripción
id_asignat	varchar (11)	identificador de la asignatura
nombre_asignat	varchar (100)	nombre de la asignatura
nombre_asignat_ing	varchar (100)	nombre de la asignatura en inglés
num_pruebas	int (20)	número de pruebas
ponderacion_controles	varchar (50)	ponderación de cada prueba
pag_web_inicial	varchar (100)	página web de la asignatura
idioma	varchar (8)	idioma en el que se imparte
codigo_etsii	varchar (4)	código etsii de la asignatura
creditos_ects	varchar (4)	créditos ects de los que consta
clase_semanal	varchar (4)	número de clases semanales
numero_alumnos_min	varchar (4)	mínimo de alumnos matriculados
numero_alumnos_max	varchar (4)	máximo de alumnos matriculados
informacion_adicional	text	información adicional
recursos	text	información complementaria
id_curso	varchar (4)	identificador del curso al que pertenece
curso_academico	varchar (100)	de qué año es toda esta información
bloque_tematico	text	bloque temático de la asignatura
factor_estudio	varchar (100)	ratio de estudio
id_departamento	varchar (4)	identificador del departamento al que pertenece
teléfono	varchar (100)	teléfono del coordinador
correo	varchar (100)	correo del coordinador
semestre	varchar (100)	semestre en el que se imparte

<code>id_coordinador</code>	<code>varchar (11)</code>	identificador del coordinador
<code>tipo</code>	<code>varchar (255)</code>	obligatoria, competencia, etc
<code>borrado</code>	<code>int (11)</code>	0 por defecto

### Tabla evaluaciones

Incluye la información relacionada con el curso de la asignatura: peso de las diversas actividades e información acerca del tipo de evaluación seguida.

En principio, existen dos tipos de evaluaciones: continua (la nota final se ve condicionada por varias calificaciones parciales) y final (la nota final suele depender mayoritariamente del examen final).

**Tabla 11. BB.DD. Tabla Evaluaciones (elaboración propia)**

nombre	tipo	descripción
<code>id_evaluacion</code>	<code>varchar (11)</code>	identificador de la evaluación
<code>id_asignat</code>	<code>varchar (11)</code>	identificador de la asignatura
<code>anno_academico</code>	<code>int (100)</code>	año en el que se dan estas características
<code>nombre</code>	<code>varchar (100)</code>	nombre del tipo de evaluación
<code>evaluacion_conocimientos</code>	<code>text</code>	descripción de la evaluación de conocimientos
<code>evaluacion_conocimientos_ing</code>	<code>text</code>	descripción de la evaluación de conocimientos en inglés
<code>peso_finales</code>	<code>float</code>	% de los exámenes sobre la nota final
<code>peso_controles</code>	<code>float</code>	% de los controles sobre la nota final
<code>peso_trabajo_ind</code>	<code>float</code>	% de los trabajos individuales sobre la nota final
<code>peso_autoevaluacion</code>	<code>float</code>	% de los ejercicios de autoevaluación sobre la nota final
<code>peso_exposiciones</code>	<code>float</code>	% de las exposiciones sobre la nota final
<code>peso_trabajo_grupo</code>	<code>float</code>	% de los trabajos de grupo sobre

		la nota final
peso_practicas	float	% de las prácticas sobre la nota final
peso_otros	float	% de otras actividades sobre la nota final
nota_minima	float	nota mínima para aprobar
otros_eval	varchar (100)	otras formas de evaluación
otros_eval_ing	varchar (100)	otras formas de evaluación en inglés
evaluacion_capacidades	text	evaluación de capacidades
evaluacion_capacidades_ing	text	evaluación de capacidades en inglés
evaluacion_competencias	text	evaluación de competencias
evaluacion_competencias_ing	text	evaluación de competencias en inglés
borrado	int (11)	0 por defecto

### Tabla horarios

Recoge todos los atributos necesarios para la identificación y descarga de los horarios previamente subidos por el profesor.

**Tabla 12. BB.DD. Tabla horarios (elaboración propia)**

Nombre	tipo	descripción
id_horario	int (100)	identificador del horario
documento	varchar (100)	nombre del documento donde se encuentra
comentarios	varchar (100)	comentarios al respecto
id_asignat	varchar (11)	identificador de la asignatura a la que pertenece
titulo	varchar (100)	título del archivo
curso_academico	int (100)	año al que pertenece

fecha_descargas	date	fecha de la última descarga
permisos	tinyint (4)	permisos
descargas_totales	int (200)	número de descargas del documento
privado	tinyint (4)	documento privado o no
borrado	int (11)	0 por defecto

### **Tabla grupos**

Incluye la información referente a los grupos en los que está matriculado el alumno.

**Tabla 13. BB.DD. Tabla grupos (elaboración propia)**

Nombre	tipo	descripción
id_grupo	int (100)	identificador del grupo
id_tipo	int (20)	identificador del tipo de grupo al que pertenece
nombre	varchar (100)	nombre del grupo
horario	int (100)	identificador del horario con el que está relacionado, de haberlo
aula	varchar (100)	aula en la que se imparte
borrado	int (11)	0 por defecto

*Tabla grupos asignaturas*

Identifica todos los grupos formados dentro de una misma asignatura.

**Tabla 14. BB.DD. Tabla grupo-asignatura (elaboración propia)**

nombre	tipo	descripción
id_grupo	int (100)	identificador del grupo
id_asignat	varchar (11)	identificador de la asignatura a la que pertenece
curso_academico	int (100)	curso al que pertenece
borrado	int (11)	0 por defecto

*Tabla grupo profesor*

Identifica el/los profesor/es que participan en cada grupo.

**Tabla 15. BB.DD. Tabla grupo profesor (elaboración propia)**

nombre	tipo	descripción
id_grupo	int (100)	identificador del grupo
id_profesor	varchar (11)	identificador del profesor
curso_academico	int (100)	curso al que pertenece
tutoria	varchar (200)	documento tutoría con el que está relacionado
borrado	int (11)	0 por defecto

### **Tabla grupo-tipo**

Establece los tipos de grupos que existen:

En la versión inicial se prevén dos grupos, el grupo virtual y el grupo normal.

**Tabla 16. BB.DD. Tabla grupo tipo (elaboración propia)**

nombre	tipo	descripción
id_tipo	int (20)	identificador del tipo de grupo
nombre	varchar (100)	nombre del tipo de grupo
borrado	int (11)	0 por defecto

### **Tabla autoevaluación**

Recoge la información sobre los ejercicios de autoevaluación realizados a través de la plataforma.

**Tabla 17. BB.DD. Tabla Autoevaluación (elaboración propia)**

nombre	tipo	descripción
id_ejercicio	int (11)	identificador del ejercicio
id_alumno	varchar (11)	identificador del alumno
id_asignat	varchar (11)	identificador de la asignatura
curso_academico	int (100)	curso en el que se establece
num_ejercicio	int (20)	número del ejercicio
calificación	double	calificación del ejercicio
fecha_entrega	date	fecha de entrega del ejercicio
grupo_oficial	int (100)	grupo oficial
grupo_virtual	int (100)	grupo virtual
semilla	int (100)	número aleatorio que identifica el ejercicio
tiempo	time	tiempo de realización del ejercicio

preguntas	varchar (100)	preguntas
respuestas	varchar (100)	respuestas
correcciones	varchar (100)	correcciones
hora	time	hora de realización
ip	varchar (100)	dirección IP establecida con el ordenador
navegador	varchar (100)	navegador empleado
borrado	int (11)	0 por defecto

### **Tabla exposiciones**

Incluye todos los datos referentes a las exposiciones realizadas por los alumnos.

**Tabla 18. BB.DD. Tabla exposición (elaboración propia)**

nombre	tipo	descripción
id_alumno	varchar (11)	identificador del alumno
id_asignat	varchar (11)	identificador de la asignatura
curso_academico	int (100)	curso en el que se da
num_ejercicio	int (20)	número del ejercicio
calificación	double	calificación
fecha_entrega	date	fecha de entrega del ejercicio
grupo_oficial	int (100)	grupo oficial
grupo_virtual	int (100)	grupo virtual, de haberlo
borrado	int (11)	0 por defecto

### **Tabla practicas**

Recoge la información sobre cada práctica realizada por cada alumno.

**Tabla 19. BB.DD. Tabla prácticas (elaboración propia)**

nombre	tipo	descripción
id_alumno	varchar (11)	identificador del alumno
id_asignat	varchar (11)	identificador de la asignatura
curso_academico	int (100)	curso en el que se da
num_practica	int (20)	número de práctica
calificación	double	calificación
fecha_entrega	date	fecha de entrega del ejercicio
grupo_oficial	int (100)	grupo oficial
grupo_virtual	int (100)	grupo virtual, de haberlo
id_laboratorio	int (20)	laboratorio al que asistir
nombre_archivo	varchar (100)	nombre del archivo a entregar como práctica
borrado	int (11)	0 por defecto

**Tabla Pruebas parciales “pecs”**

Clasificación de las pruebas parciales realizadas por los alumnos.

**Tabla 20. BB.DD. Tabla pruebas parciales (elaboración propia).**

nombre	tipo	descripción
id_control	int (20)	identificador del control
id_alumno	varchar (11)	identificador del alumno
id_asignat	varchar (11)	identificador de la asignatura
curso_academico	int (100)	curso en el que se da
num_control	int (20)	número de control
calificación	double	calificación
fecha_entrega	date	fecha de entrega del ejercicio
grupo_oficial	int (100)	grupo oficial
grupo_virtual	int (100)	grupo virtual, de haberlo

borrado	int (11)	0 por defecto
---------	----------	---------------

### ***Tabla Trabajos en grupo “tr\_grupal”***

Representa las características de los trabajos en grupo realizados por los alumnos.

**Tabla 21. BB.DD. Tabla trabajo en grupo (elaboración propia)**

nombre	tipo	descripción
id_alumno	varchar (11)	identificador del alumno
id_asignat	varchar (11)	identificador de la asignatura
curso_academico	int (100)	curso en el que se da
num_ejercicio	int (20)	número del ejercicio
calificación	double	calificación
fecha_entrega	date	fecha de entrega del ejercicio
grupo_oficial	int (100)	grupo oficial
grupo_virtual	int (100)	grupo virtual, de haberlo
borrado	int (11)	0 por defecto

### ***Tabla trabajos individuales “tr\_individual”***

Comprende las características de los trabajos individuales realizados por los alumnos.

**Tabla 22. BB.DD. Tabla trabajo individual (elaboración propia)**

nombre	tipo	descripción
id_alumno	varchar (11)	identificador del alumno
id_asignat	varchar (11)	identificador de la asignatura
curso_academico	int (100)	curso en el que se da
num_ejercicio	int (20)	número del ejercicio
calificación	double	calificación
fecha_entrega	date	fecha de entrega del ejercicio

grupo_oficial	int (100)	grupo oficial
grupo_virtual	int (100)	grupo virtual, de haberlo
borrado	int (11)	0 por defecto

## *Alumnos y Asignaturas*

### **Tabla alumnos**

Hace referencia a los datos del alumno que se utilizan en esta parte de la interfaz; incluyendo datos estadísticos de accesos realizados

**Tabla 23. BB.DD. Tabla alumnos (elaboración propia)**

nombre	tipo	descripción
id_alumno	varchar (11)	identificador del alumno
contraseña	varchar (255)	contraseña del alumno
nombre	varchar (100)	nombre
apellidos	varchar (100)	apellidos
num_mat	varchar (100)	número de matrícula
correo	varchar (100)	correo
teléfono	varchar (100)	teléfono
imagen	varchar (200)	imagen del alumno
visitas_total	int (11)	número de visitas
visitas_ultima	date	fecha de la última visita
borrado	int(11)	0 por defecto

### **Tabla Alumnos y asignaturas “alumno-asignat”**

Incluye las medias de las calificaciones de las distintas actividades, así como otras calificaciones y notas del alumno en la asignatura en cuestión.

**Tabla 24. BB.DD. Tabla Alumnos asignaturas (elaboración propia)**

nombre	tipo	descripción
id_alumno	varchar (11)	identificador del alumno
id_asignat	varchar (11)	identificador de la asignatura
anno_matricula	int (100)	año en el que se establece
grupo_oficial	int (100)	grupo oficial
grupo_virtual	int (100)	grupo virtual
observaciones	text	observaciones
media_practicas	double	media de las prácticas
media_autoevaluacion	double	media de los ejercicios de autoevaluación
media_controles_escritos	double	media de los controles escritos
media_trabajos_ind	double	media de los trabajos individuales
media_trabajos_eq	double	media de los trabajos en equipo
media_exposiciones	float	media de las exposiciones
tipo_ev	int (20)	identificador del tipo de evaluación
ordinaria	double	nota de la convocatoria ordinaria
extraordinaria	double	nota de la convocatoria extraordinaria
estado	int (20)	estado de la asignatura con respecto al alumno
curso	varchar (100)	curso
cuestionario	text	cuestionario
provisional	text	provisional
borrado	int (11)	0 por defecto

## *Profesores y Asignatura*

### **Tabla profesores**

Hace referencia a los datos del profesor que se manejan en esta parte de la interfaz

**Tabla 25. BB.DD. Tabla profesores (elaboración propia)**

nombre	tipo	descripción
id_profesor	varchar (11)	identificador del profesor
contraseña	varchar (255)	contraseña del profesor
nombre	varchar (100)	nombre
apellido1	varchar (100)	primer apellido
apellido2	varchar (100)	segundo apellido
correo	varchar (100)	correo
teléfono	varchar (100)	teléfono
dni	varchar (100)	DNI
visitas_total	int (11)	número de visitas
visitas_ultima	date	fecha de la última visita
alta	date	fecha de alta
baja	date	fecha de baja
borrado	int (11)	0 por defecto

### **Tabla Profesores y asignaturas “prof-asignat”**

Relaciona la tabla “profesores” con la de “asignaturas”.

**Tabla 26. BB.DD. Tabla profesores-asignaturas (elaboración propia)**

nombre	tipo	descripción
id_profesor	varchar (11)	identificador del profesor
id_asignat	varchar (11)	identificador de la asignatura
administrador	int (11)	si el profesor es administrador o no
tipo	varchar (100)	tipo de profesor
curso_academico	varchar (100)	curso en el que se da
borrado	int (11)	0 por defecto

### **Tabla tutorías**

**Tabla 27. BB.DD. Tabla tutorías (elaboración propia)**

nombre	tipo	descripción
id_tutoria	int (20)	identificador del archivo de tutoría
documento	varchar (100)	documento en el que se aloja
id_profesor	varchar (11)	identificador del profesor
comentarios	varchar (200)	comentarios
titulo	varchar (100)	titulo del archivo
id_asignat	int (20)	identificador de la asignatura
curso_academico	int (100)	curso en el que se da
fecha_descargas	date	fecha de la última descarga
permiso	tinyint (4)	permiso de descarga
descarga_totales	int (200)	número de descargas

privado	tinyint (4)	privado o no
borrado	int (11)	0 por defecto

Recoge todos los atributos necesarios para la identificación y descarga de los documentos de tutorías previamente subidos por el profesor.

### Cuestionarios

**Tabla cuestionarios**

**Tabla 28. BB.DD. Tabla cuestionarios (elaboración propia)**

nombre	tipo	descripción
id_pregunta	int (100)	identificador de la pregunta
id_asignat	int (11)	identificador de la asignatura
curso_academico	int (100)	curso en el que se da
tema	tinyint (1)	si es tema o no
texto_pregunta	varchar (500)	texto de la pregunta
orden	int (20)	orden al que se ve sujeto
borrado	int (11)	0 por defecto

Recoge las preguntas de los cuestionarios.

**Tabla respuestas cuestionarios "resp-cuestionario"**

**Tabla 29. BB.DD. Tabla respuestas a cuestionarios (elaboración propia)**

nombre	tipo	Descripción
id_respuesta	int (20)	identificador de la respuesta
id_alumno	varchar (11)	identificador del alumno
id_asignat	varchar (11)	identificador de la asignatura
curso_academico	int (100)	curso en el que se da
respuestas	varchar (500)	respuestas separadas por ","

mejor	varchar (500)	respuesta a "lo mejor"
peor	varchar (500)	respuesta a "lo peor"
sugerencias	varchar (500)	respuesta a "sugerencias"
borrado	int (11)	0 por defecto

Incluye las respuestas de los cuestionarios.

### *Observaciones*

#### *Tabla Observaciones*

**Tabla 30. BB.DD. Tabla respuesta a cuestionarios (elaboración propia)**

nombre	tipo	descripción
id_alumno	varchar (11)	identificador del alumno
id_asignat	varchar (11)	identificador de la asignatura
curso_academico	int (100)	curso en el que se da
fecha	date	fecha en la que se escribe
observación	varchar (500)	texto de la observación
orden	int (20)	orden en el que queda
borrado	int (11)	0 por defecto

Recoge las observaciones realizadas por el profesor y mostradas a los alumnos.

## Foro

### *Tabla foro*

**Tabla 31. BB.DD. Tabla foro (elaboración propia)**

nombre	tipo	descripción
id_clave	int (20)	identificador de la clave
id_resp	int (20)	identificador del texto al que responde. En caso de no responder a ninguno es "0"
persona	varchar (500)	persona que escribe
imagen	varchar (200)	imagen del escritor
titulo	varchar (100)	titulo del texto
texto	varchar (500)	texto escrito
id_asignat	varchar (11)	identificador de la asignatura
curso_academico	int (100)	curso en el que se da
fecha	date	fecha de escritura
hora	time	hora de escritura
borrado	int (11)	0 por defecto

Recoge la información del foro, así como sus características.

### *Tabla Vistas del foro “foro-visto”*

**Tabla 32. BB.DD. Tabla vista de foro (elaboración propia)**

nombre	tipo	descripción
id_clave	int (20)	identificador del texto visto
persona	varchar (500)	identificador de la persona que lo ha leído
fecha	date	fecha en la que se lee

borrado	int (11)	0 por defecto
---------	----------	---------------

Evita que un envío se marque como “nuevo” una vez visto.

#### 2.4.4 Diseño e interfaz del modulo Alumno

Como se ha indicado, el objetivo del modulo alumno, es el acceso a la BB.DD. para proporcionar a los clientes la información solicitada. Dentro del módulo alumno, se puede acceder a distintos contenidos proporcionados por la herramienta:

Los datos a los que permite acceder la herramienta son de tres tipos.

- Datos informativos
- Acceso a contenidos específicos (descarga de información)
- Interfaz para facilitar las comunicaciones con el profesor, realización de consultas y contactos a través del foro de alumnos.

En la figura 10 se pueden visualizar los distintos apartados en los que queda dividido el módulo alumno.



Figura 10. Actividades desde el modulo alumno (elaboración propia)

Desde el punto de vista de la interfaz, estas informaciones se clasifican de la siguiente manera (tabla 33):

**Tabla 33. Funcionalidad de AulaWeb (elaboración propia)**

Funcionalidades de AulaWeb 2.0	Comentario
<b>Área de Información</b>	Datos de partida del alumno, asignaturas en las que está matriculado, observaciones  Datos de evaluación: Evaluación final, calificaciones emitidas por el profesor, certificaciones de estudios, etc.
<b>Área de contenidos</b>	Incluye los contenidos y los correspondientes programas de los cursos, guías didácticas, materiales de los cursos y recursos externos, etc.
<b>Área de comunicaciones</b>	Se relaciona con el correo electrónico, foros de debate, chat, etc
<b>Área de Actividades</b>	Actividades y trabajos realizados por los alumnos, individualmente o en grupo

La carga y presentación de la información se estructura en una serie de ventanas, a partir de un esquema que sigue la estructura indicada por la tabla 33 y que, para facilidad de referencia se presenta al usuario. De esta forma, el alumno, usuario de *AulaWeb 2.0*, conoce en cada momento a qué tipo de información está accediendo y puede conocer los elementos que están relacionados.

La interfaz que se utiliza tiene una clave de colores propia del modulo alumno (azul) y presenta una o varias líneas con pestañas que indican el tipo y características de la información a la que se accede. Por ejemplo, éste es la cinta superior para el acceso a la información básica de alumno (figura 11).

**Figura 11. Presentación de la interfaz (elaboración propia)**

Este modelo es el que se utiliza en todas las ventanas informativas de la aplicación.

A continuación, se detallan las interfaces y funcionalidades de los distintos elementos. En cada caso, se explica:

- a) El área al que pertenece la información solicitada
- b) El tipo de información al que se accede
- c) La forma de la interfaz de usuario
- d) Las acciones posibles sobre esta información
- e) Las respuestas en caso de error

#### 2.4.4.1 Información: Información del Alumno

Área	Información
Funcionalidad	información del alumno

Muestra los datos del alumno, clasificándolos en:

- Datos de acceso, (usuario y contraseña)
- Datos Personales (nombre, apellidos, etc.)
- Datos de Contacto, (e-mail y teléfono)
- Incluye la fotografía del alumno.

La figura 12 contiene un ejemplo:

Figura 12. Interfaz de acceso al sistema (elaboración propia)

A continuación, se resume la funcionalidad en la tabla 34 y las acciones de error en la tabla 35.

**Tabla 34. Funcionalidad básica del Interfaz de acceso (elaboración propia)**

Acciones del usuario	Respuesta del Sistema
El usuario pulsa en el vínculo “Cambiar Contraseña”	Aparece una ventana por la que se puede cambiar la contraseña (ver condición cambio de contraseña)
El usuario pulsa en el vínculo “Cambiar Contraseña” y a continuación en “?”	Aparece en un cuadro la “ayuda” para cambiar la contraseña
El usuario modifica los datos de contacto y pulsa en el vínculo “Actualizar Contacto”	Se actualiza la base de datos (“correo” y “teléfono” del alumno) con la información enviada

**Tabla 35. Acciones error. Acceso al sistema (elaboración propia)**

Acciones de Error	Respuesta del Sistema
Alguno de los campos a modificar queda en blanco	Aparece una ventana emergente que te avisa de que no se actualizará la Base de Datos en caso de no rellenarlos. (El programa no realiza ninguna comprobación posterior)
Se produce un error de acceso a la Base de Datos	Aparece el texto, “Error de Acceso”. (Este error puede aparecer debido a fallos en la línea de comunicación)

#### 2.4.4.2 Información: Editar información del Alumno

Área	Información
Funcionalidad	Editar información del alumno. Contraseña

Al pulsar en el vínculo “Cambio de Contraseña” se abre una ventana con los siguientes campos: “Contraseña Actual”, “Nueva Contraseña” y “Confirmación” (figura 13).

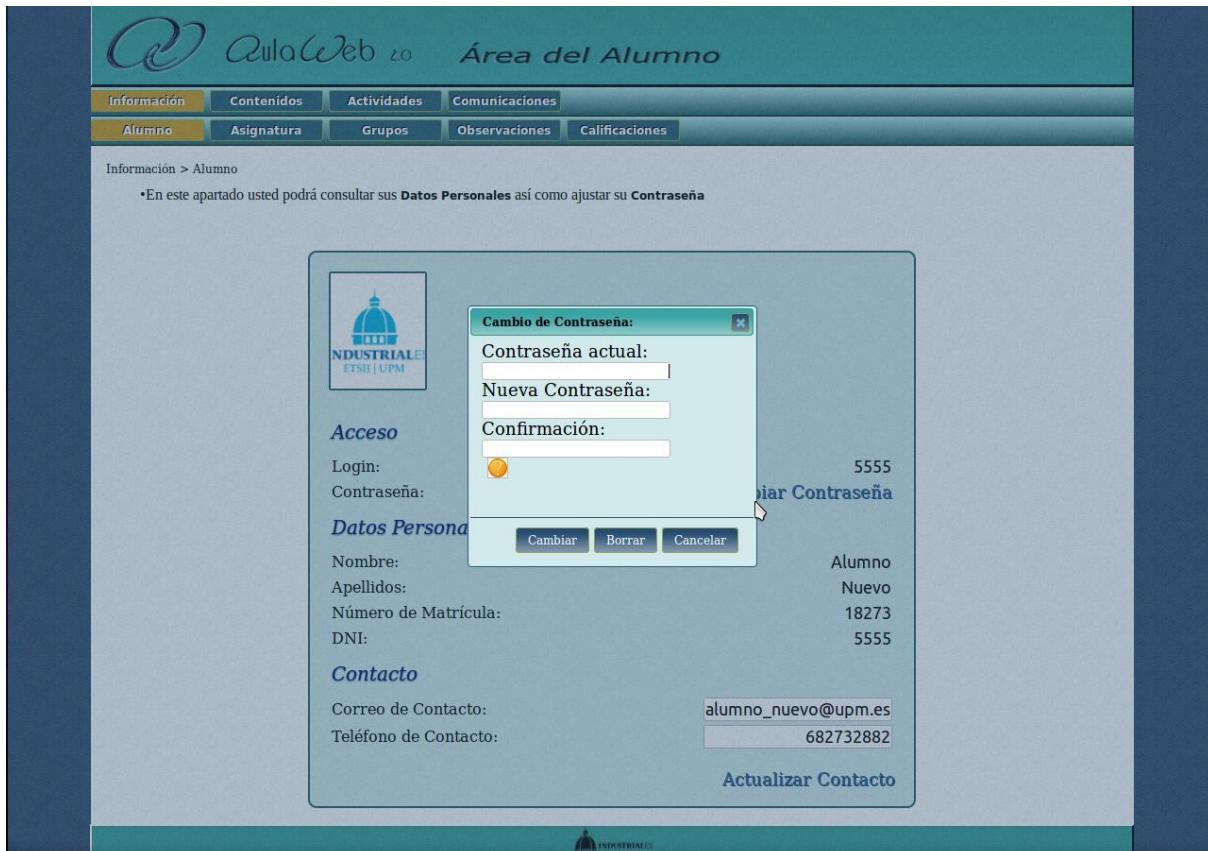


Figura 13. Interfaz Datos personales (elaboración propia)

Para que se cambie la contraseña correctamente es necesario rellenarlos todos con:

- En el primero: la contraseña actual.
- En el segundo: la contraseña que se quiere poner en su lugar.
- En el tercero: la misma contraseña por la que se quiere cambiar la actual.

Y pulsar, a continuación, en el botón “cambiar”.

**Tabla 36. Acciones de error. Datos personales (elaboración propia)**

Acciones de error	Respuesta del sistema
En el caso de dejar alguno de los campos vacíos,	Aparece una ventana emergente de aviso
“nueva contraseña” y la “confirmación” no coinciden	Aparece un mensaje advirtiendo y/o se modifica la base de datos

#### 2.4.4.3 Información: Asignatura

Área	Información
Funcionalidad	Asignatura

Por medio de esta acción se visualiza la asignatura con todos sus contenidos. No se puede modificar nada, es meramente informativo. Los datos correspondientes se introducen a través del modulo de profesor.

5007-Fundamentos de Programación				
Código ETSII	5007	Nombre	Fundamentos de Programación	
Tipo de Asignatura	Obligatoria	Plan de Estudios	Grado en Ingeniería en Tecnologías Industriales	
Departamento	Automática, Ingeniería Electrónica e Informática Industrial		Teléfono	91-336-3200
Unidad Docente	Informática Industrial		Web	<a href="http://www.dii.etsii.upm.es">www.dii.etsii.upm.es</a>
Bloque Temático	Informática		E-mail	informatica@etsii.upm.es
Idioma	Semestre	Especialidad	Coordinador/a de la Asignatura	
Español	2	0		
Nº Alumnos		Curso	Clases/Sem	Factor estudio
Min	Max			ECTS
0	1033	Cuarto Curso	4	1,5
CONOCIMIENTOS QUE NECESITA				
Asignatura	Fundamentos de Programación			
Módulo	Fundamentos			
Tema	Estructura de un programa (2 h)			
Sin Asignar				
CAPACIDADES Y HABILIDADES QUE NECESITA				
<ul style="list-style-type: none"> <li>Conocimiento de matemáticas a nivel pre-universitario</li> </ul>				
CONTENIDO BREVE		CONOCIMIENTOS QUE APORTA		
MODULO 0 : Información general de la asignatura		• Tema 0: Información general de la asignatura		

**Figura 14. Interfaz informativo de la asignatura (elaboración propia)**

**Tabla 37. Acciones de error. Interfaz informativo de la asignatura (elaboración propia)**

Acciones de error	Respuesta del sistema
No se puede acceder a la base de datos	aparece un texto con “Error de Acceso”

#### 2.4.4.4 Información: Grupos Asignados

Área	Información
Funcionalidad	Grupos asignados

Recoge la información acerca de los grupos, clasificada en: “Grupos Asignados”, “Profesores” e “Información Adicional”.

- Grupos Asignados: Grupo por matrícula y Grupo profesor.
- Profesores: Proporciona los profesores de cada grupo así como sus datos personales y tutorías.
- Información Adicional: Aula y horarios

Como en el caso anterior, esta información no puede modificarse

**Figura 15. Interfaz de información de asignaturas (elaboración propia)**

**Tabla 38. Acciones usuario (elaboración propia)**

Acciones del usuario	Respuesta del Sistema
El usuario pulsa en el vínculo “Consultar Tutorías”	Se descarga el documento relativo a las tutorías
El usuario pulsa en el vínculo “Consultar Horarios”	Se descarga el documento relativo a los horarios

**Tabla 39. Acciones de error (elaboración propia)**

Acciones de error	Respuesta del sistema
No se puede acceder a la base de datos	aparece un texto con “Error de Acceso”
No existe documento	No se produce ninguna descarga

#### 2.4.4.5 Información: Observaciones

<b>Área</b>	Información
Funcionalidad	observaciones

Recoge las observaciones realizadas por los profesores a lo largo del curso y la fecha en la que se publicaron.



Figura 16. Interfaz observaciones (elaboración propia)

Tabla 40. Acciones. Observaciones (elaboración propia)

Acciones del Usuario	Respuesta del Sistema
El usuario pulsa en la primera flecha de la derecha	Se muestra la siguiente observación más reciente
El usuario pulsa en la segunda flecha de la derecha	Se muestra la última observación realizada
El usuario pulsa en la primera flecha de la izquierda	Se muestra la anterior observación realizada
El usuario pulsa en la segunda flecha de la izquierda	Se muestra la primera observación realizada

**Tabla 41. Acciones de error. Observaciones (elaboración propia)**

<b>Acciones de error</b>	<b>Respuesta del sistema</b>
No se puede acceder a la base de datos	aparece un texto con “Error de Acceso”

#### 2.4.4.6 Información: Calificaciones

<b>Área</b>	Información
Funcionalidad	Calificaciones

Muestra las calificaciones obtenidas en Enero/Junio y Julio, así como el estado en el que se encuentra la asignatura seguida hasta el momento.

**Figura 17. Interfaz calificaciones (elaboración propia)****Tabla 42. Acciones de error. Calificaciones (elaboración propia)**

<b>Acciones de error</b>	<b>Respuesta del sistema</b>
No se puede acceder a la base de datos	aparece un texto con “Error de Acceso”

#### 2.4.4.7 Información: Evaluación

<b>Área</b>	Información
Funcionalidad	Evaluación

Muestra las valoraciones de las pruebas seguidas así como sus calificaciones, según la evaluación que se esté cursando en ese momento. También incluye la media y la ponderación de cada apartado.

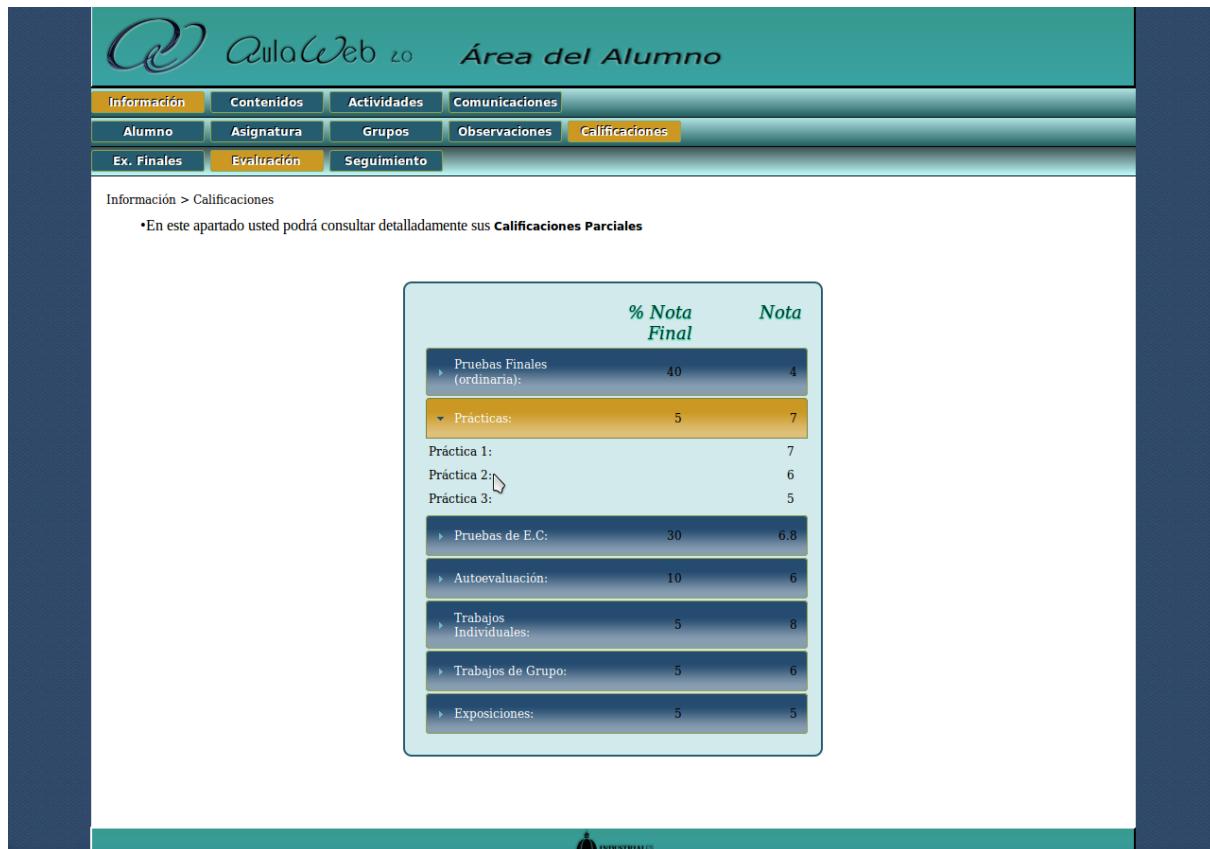


Figura 18. Interfaz Calificaciones parciales (elaboración propia)

Aparecen una serie de secciones con dos números: el primero representa el porcentaje de nota que establece esa sección y el segundo la nota.

Tabla 43. Acciones usuario calificaciones parciales (elaboración propia)

Acciones del Usuario	Respuesta del Sistema
El usuario pulsa en cada división de cada apartado	Se muestran las pruebas que se han llevado a cabo de ese tipo de evaluación, así como las calificaciones y ponderaciones que suponen.

**Tabla 44. Acciones de error. Calificaciones parciales (elaboración propia)**

Acciones de error	Respuesta del sistema
No se puede acceder a la base de datos	aparece un texto con “Error de Acceso”

#### 2.4.4.8 Información: Seguimiento

Área	Información
Funcionalidad	Seguimiento

Recoge el tipo de evaluación seguida, así como información acerca de ésta en caso de ser evaluación continua: número de pruebas a realizar y porcentaje de cada prueba.

**Figura 19. Interfaz evaluaciones (elaboración propia)****Tabla 45. Acciones de error: evaluaciones (elaboración propia)**

Acciones de error	Respuesta del sistema
No se puede acceder a la base de datos	aparece un texto con “Error de Acceso”

#### 2.4.4.9 Comunicaciones: Cuestionario

<b>Área</b>	Comunicaciones
Funcionalidad	Cuestionario

Se trata de un cuestionario escrito por el profesor por el cual el alumno hace una evaluación de la asignatura. La calificación es del 0 al 5, siendo 5 la nota más alta

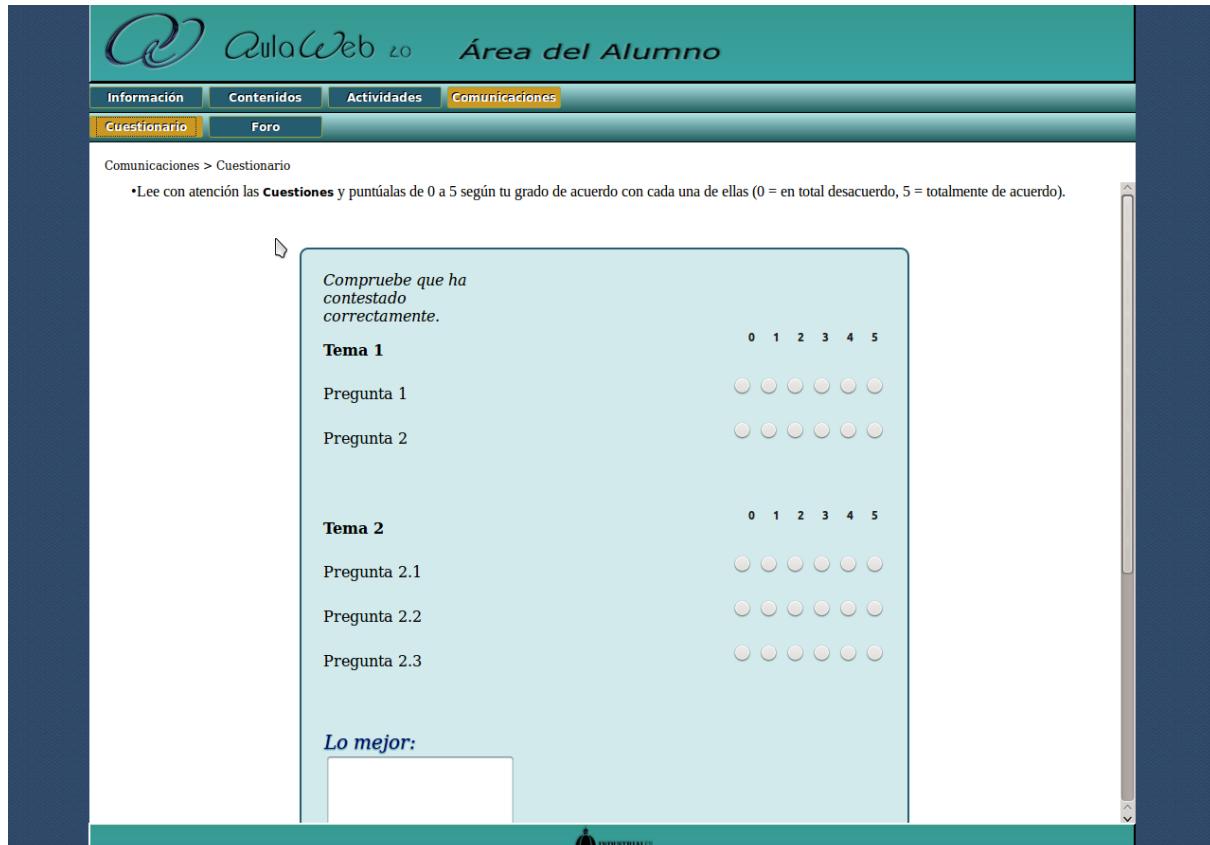


Figura 20. Interfaz de usuario. Cuestionario (elaboración propia)

Tabla 46. Acciones de usuario: Cuestionario (elaboración propia)

Acciones del Usuario	Respuesta del Sistema
El usuario rellena el cuestionario y pulsa enviar	Las respuestas señaladas por el alumno quedan guardadas en la Base de Datos.

Tabla 47. Acciones de error: Cuestionario (elaboración propia)

Acciones de Error	Respuesta del sistema
No se puede acceder a la base de datos	Aparece un texto con "Error de Acceso"

#### 2.4.4.10 Comunicaciones: Foro

<b>Área</b>	Comunicaciones
Funcionalidad	Foro

Muestra el foro de la asignatura.

En el foro se mantiene un registro actualizado de las comunicaciones realizadas por los distintos alumnos así como las respuestas de otros alumnos y el profesor. Este módulo permite establecer una comunicación fluida entre alumnos y profesores.

Figura 21. Interfaz Foro de la asignatura (elaboración propia)

Figura 22. Interfaz foro de la asignatura. Comentarios. (elaboración propia)

**Tabla 48. Acciones de usuario. Comentarios. (elaboración propia)**

<b>Acciones del Usuario</b>	<b>Respuesta del Sistema</b>
El usuario pulsa en el ícono de la izquierda de los mensajes principales	Se despliega el historial del mensaje con sus respuestas ordenadas jerárquicamente en forma de árbol
El usuario pulsa en el mensaje	Se despliega el mensaje con un posible nuevo mensaje (respuesta) a enviar por el usuario.
El usuario rellena un “Nuevo Tema” y pulsa enviar	Almacena el nuevo mensaje en la Base de Datos

**Tabla 49. Acciones de error. Comentarios (elaboración propia)**

<b>Acciones de Error</b>	<b>Respuesta del Sistema</b>
Se pulsa a Enviar dejando el título en blanco	Aparece un mensaje anunciándonos que no se puede continuar con el proceso de dejar ese campo vacío
No se puede acceder a la base de datos	aparece un texto con “Error de Acceso”

#### 2.4.4.11 Contenidos

<b>Área</b>	Contenidos
Funcionalidad	General

Muestra los documentos: exámenes, guiones de prácticas, etc. visibles al alumno y disponibles para descargar.

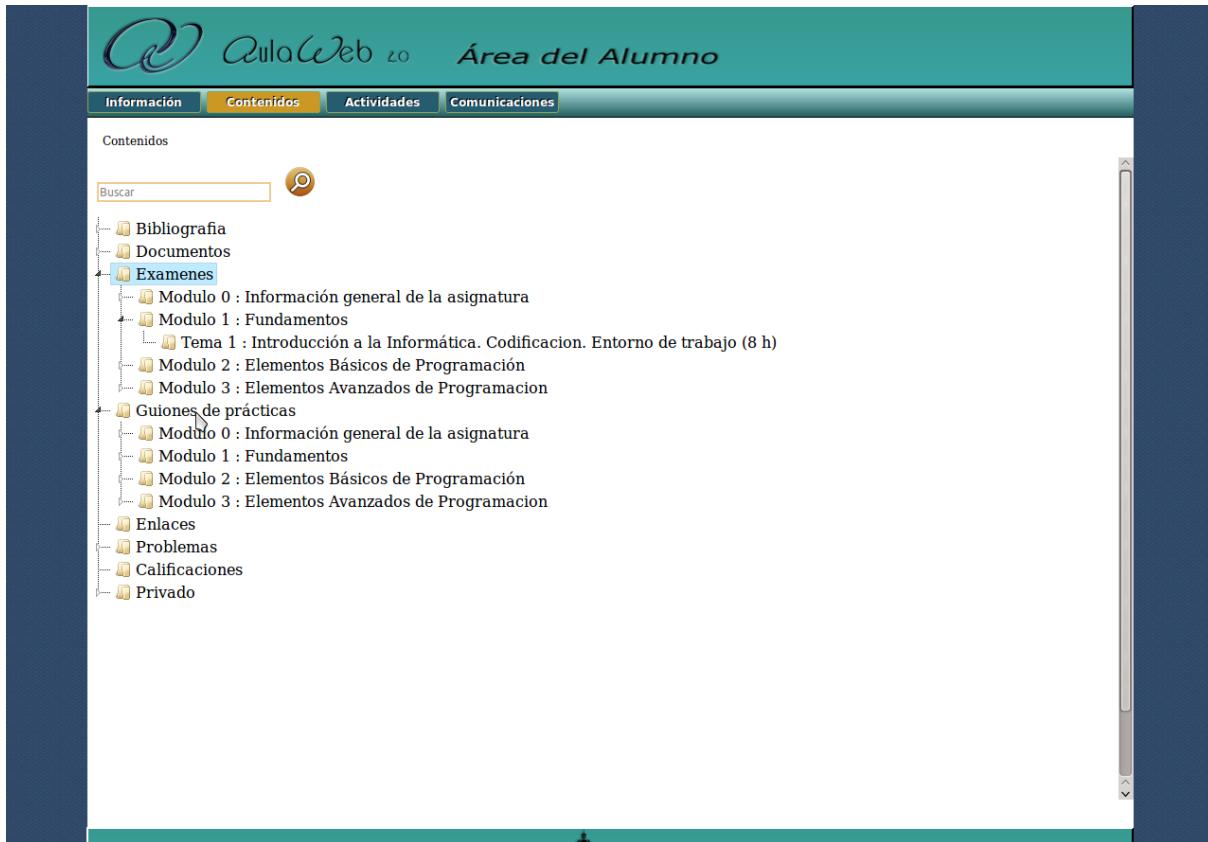


Figura 23. Interfaz de acceso a contenidos (elaboración propia)

Nº Mod.	Nº Tem.	ID Ref.	Título	Profesor	Fecha	Nº descargas desde(dd/mm/aa)	Nº descargas totales	Tipo Acceso
<input checked="" type="checkbox"/>	0	E001	Enunciados Control 1 Modelo A,B,C,D curso 2010-2011	raquelm	04/05/2011	2972 -	3313	Nulo
<input checked="" type="checkbox"/>	0	E002	Enunciados Control 2 Modelo A,B,C,D curso 2010-2011	raquelm	04/05/2011	2448 -	3003	Nulo
<input checked="" type="checkbox"/>	0	E003	Examen de Junio de 2011	agarcia	03/06/2011	3437 -	4178	Nulo
<input checked="" type="checkbox"/>	0	E004	Examen de Julio de 2011	agarcia	30/01/2012	2664 -	2664	Nulo
<input checked="" type="checkbox"/>	0	E005	Enunciados Control 1 Modelo A,B,C,D curso 2011-2012	raquelm	26/03/2012	2706 -	2706	Nulo
<input checked="" type="checkbox"/>	0	E006	Enunciados Control 2 Modelo A,B,C,D curso 2011-2012	raquelm	03/05/2012	2077 -	2077	Nulo
<input checked="" type="checkbox"/>	0	E007	Examen de Mayo-Junio de 2012	agarcia	31/05/2012	2496 -	2496	Nulo
<input checked="" type="checkbox"/>	0	E009	Enunciados Primer Control - Curso 2012-2013 (Mod. A, B, C y D)	agarcia	11/03/2013	1340 -	1340	Nulo
<input type="checkbox"/>	0	E011	Examen de Junio de 2013	raquelm	07/06/2013	1748 -	1748	Nulo

Figura 24. Interfaz de acceso a contenidos (2) (elaboración propia)

**Tabla 50. Acciones de usuario. Acceso a contenidos (elaboración propia)**

<b>Acciones del Usuario</b>	<b>Respuesta del Sistema</b>
El usuario pulsa sobre la flechita de la izquierda de los temas	Se despliega su contenido en forma de árbol
El usuario pulsa sobre los temas	Se despliega en un diálogo todos los documentos que el tema recoge para poder ser descargados
El usuario selecciona alguno de los documentos y pulsa “Iniciar Descarga”	Se procede a la descarga de los elementos solicitados

**Tabla 51. Acciones de error. Acceso a contenidos (elaboración propia)**

<b>Acciones de Error</b>	<b>Respuesta del Sistema</b>
No se puede acceder a la base de datos	Aparece un texto con “Error de Acceso”
No existe documento	No se produce ninguna descarga

## 2.5 Control de versiones

Como en todos los proyectos en los que intervienen varios desarrolladores o que son suficientemente grandes, el proyecto *AulaWeb 2.0* debe realizar un control de versiones para asegurar que los diferentes participantes y programadores del proyecto disponen de la versión actualizada del código y se asegura la interoperabilidad.

El sistema de control de versiones se encarga de manejar todos los cambios que se realizan en un proyecto software. Cualquier modificación que se realice en el código de la aplicación quedará registrada. Normalmente, el control de versiones incluye también los comentarios correspondientes para indicar los cambios y modificaciones realizadas. El controlador de versiones consta de dos partes:

- a) El tronco, donde se almacena la versión principal
- b) Ramas, donde se pueden almacenar variaciones del proyecto, que pueden tener una evolución diferente

Evidentemente, el sistema de control de versiones precisa de la existencia de un lugar donde se almacenan las últimas versiones operativas. Este lugar se denomina “Repositorio”. Normalmente suele estar en un servidor central.

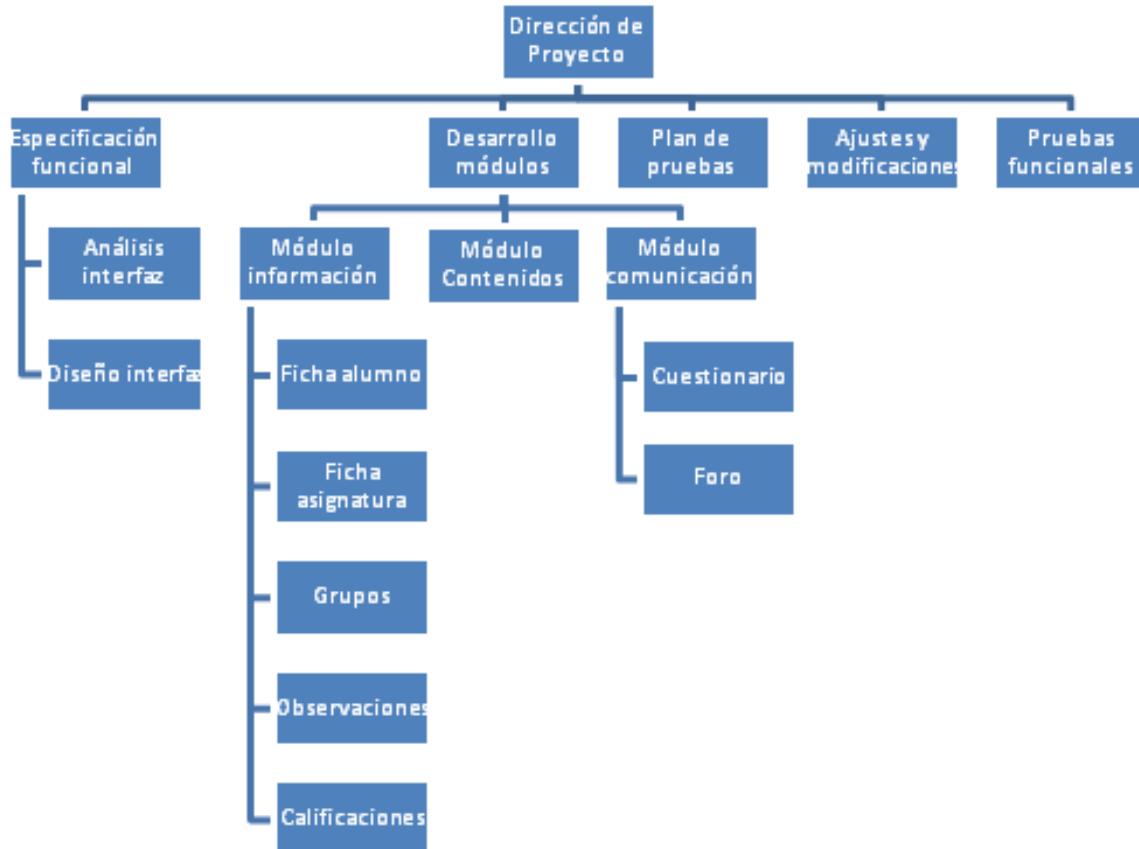
El controlador de versiones es una herramienta esencial para asegurar la funcionalidad del sistema. Si no se lleva un control de los distintos módulos incorporados al programa será muy difícil identificar el origen de los posibles problemas y, por lo tanto, la depuración puede ser mucho más complicada. El poder disponer de las versiones anteriores permite aislar los posibles fallos, facilita la depuración.

En *AulaWeb 2.0* se ha utilizado *Subversion* como sistema de control de versiones. *Subversion* es, en línea con lo previsto en el proyecto, un sistema *Open Source* y ha sido desarrollado en *Apache*.

### 3. Planificación y Programación

#### 3.1 Plan de trabajo

El proyecto se estructura de acuerdo con la siguiente estructura (figura 25).



**Figura 25. Estructura de Descomposición del Producto (elaboración propia)**

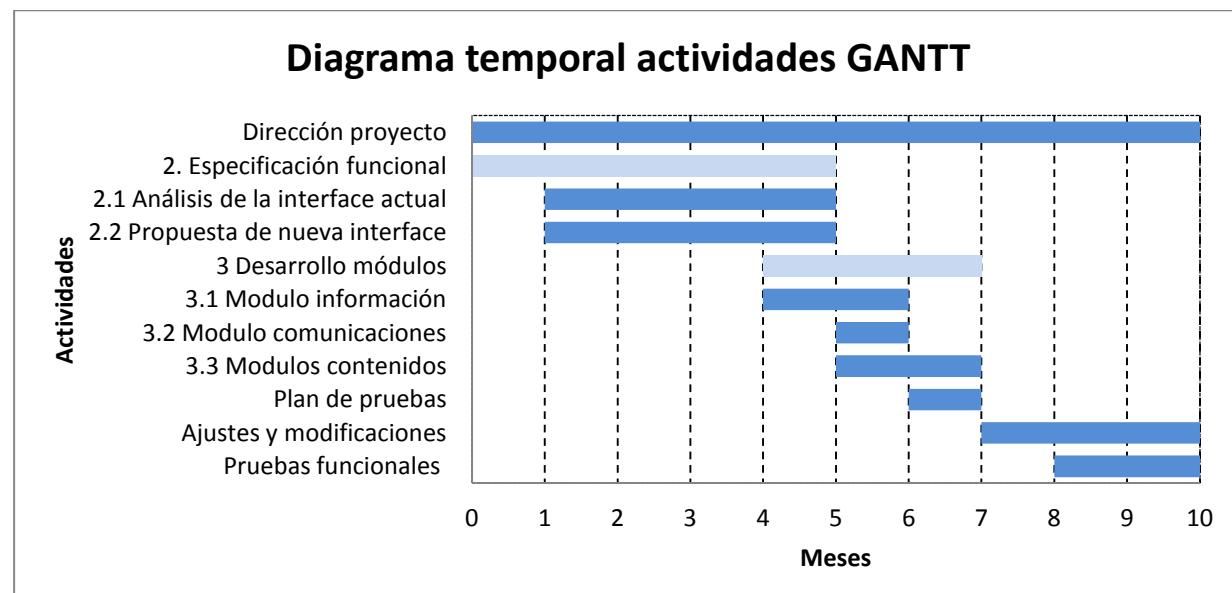
Esta estructura incluye, en la parte correspondiente al desarrollo de los módulos, una indicación de la estructura del programa y los distintos elementos que contiene. Estos módulos están relacionados, de forma directa con la base de datos, según se indica en el apartado anterior.

### 3.2 Diagrama temporal

En la tabla 52 se establece el siguiente plan de trabajo temporal. (La unidad de tiempo es el mes) Y en la figura 26 se muestra el Diagrama de GANTT,

**Tabla 52. Descripción temporal de actividades (elaboración propia)**

Tareas				
Comienzo (mes)	Fin (mes)	Tarea		duración (meses)
0	10	Dirección proyecto		10
0	5	2. Especificación funcional		5
1	5	2.1 Análisis de la interfaz actual		4
1	5	2.2 Propuesta de nueva interfaz		4
4	7	3 Desarrollo módulos		3
4	6	3.1 Modulo información		2
5	6	3.2 Modulo comunicaciones		1
6	7	3.3 Módulos contenidos		2
6	7	Plan de pruebas		1
7	10	Ajustes y modificaciones		3
8	10	Pruebas funcionales		2



**Figura 26. Diagrama temporal GANTT (elaboración propia)**

## 4. Valoración del proyecto

### 4.1 Alternativas consideradas

La valoración del coste de un desarrollo Software es una tarea extraordinariamente complicada puesto que el coste de un desarrollo depende de múltiples factores, que varían desde la complejidad de la tarea a realizar, el grado de fiabilidad que debe tener el software desarrollado, la existencia o no de versiones anteriores, la experiencia de los programadores y el grado de conexión entre ellos, etc.

En la práctica, el presupuesto de mayoría de los proyectos de software se establece por experiencia: es decir, uno o varios expertos que conocen la materia estiman el coste esperado basándose en proyectos similares realizados en el pasado. Evidentemente, estas estimaciones pueden variar enormemente. Para tratar de reducir estas varianzas, es frecuente contar con varias estimaciones de distintos expertos. Si se realiza la media, las estimaciones suelen converger hacia valores comunes, especialmente si los desarrollos previstos se basan en lenguajes y plataformas de desarrollo conocidas.

Sin embargo, aun realizando medias entre diversos expertos, la mayoría de los estudios [wse] indican que las estimaciones tienden a ser muy optimistas y es frecuente que, al menos, los costes reales puedan ser un 30% superior. Por ejemplo, es frecuente constatar que si el profesional dice estar convencido en un 90% de que el proyecto se va a desarrollar en fechas, la estimación frecuentemente es errónea y el esfuerzo real suele ser un 30 o un 40% mayor que el estimado.

Por tal motivo, se han desarrollado algunos métodos, basados en la construcción de modelos estadísticos, en los que se han tratado de tener en cuenta las características del desarrollo y la experiencia anterior en desarrollos similares. Por supuesto, estos métodos son métodos heurísticos, es decir basados en la experiencia, sin que exista una teoría fiable que pueda considerarse como una referencia.

En este momento, la mayor parte de los modelos que pueden utilizarse, además de las estimaciones de expertos mencionadas, suelen ser modelos paramétricos. En estos métodos se toman una serie de datos de partida, como el número de líneas de código, la complejidad del proyecto, la experiencia de los desarrolladores, etc. Estos métodos se denominan de *estimación formal* por contraposición a los métodos puramente de tanteo utilizados normalmente. En muchos casos, los métodos se combinan o comparan con las estimaciones de los expertos utilizando técnicas mixtas. Sin embargo, como se ha indicado, no parece existir, en este momento, una solución convincente para estimar el coste de un desarrollo software, en particular cuando se aplican métodos paramétricos obtenidos en casos que no son totalmente similares a los que se desean analizar.

Como dato curioso y que ilustra la poca fiabilidad de las predicciones, mostrando la visión pesimista que tienen los propios desarrolladores sobre la exactitud de sus predicciones se pueden citar algunos de los comentarios que circulan. Estos comentarios son esencialmente frases humorísticas que no pueden, evidentemente, ser tomadas en serio y son solo ilustraciones de la dificultad de la tarea:

*La regla del 90 por ciento: “El primer 90% del código consume, generalmente, el 90% del tiempo y dinero que se ha presupuestado. El restante 10% consume, también, el 90% del presupuesto” (Tom Cargill)*

*La regla de Hofstadter: “Un desarrollo SW siempre necesita más tiempo del previsto, incluso cuando se tiene en cuenta la regla de Hofstadter”*

En cualquier caso, en la práctica, es preciso realizar una estimación de los costes, sea o no totalmente fiable. En ese sentido la tabla 53 muestra un ejemplo de las soluciones más frecuentes para la estimación de coste de desarrollo SW:

Tabla 53. Técnicas de estimación de coste de proyecto (fuente: [coc2])

Técnica de estimación	Tipo de estimación	Ejemplo
Estimación basada en analogías	Modelo de estimación formal	Método <i>Weighted Micro Function Points</i>
Método basado en <i>Work breakdown</i>	Estimación por expertos	Es la más usada por diversas compañías y software de <i>project management</i>
Modelos paramétricos	Modelo de estimación formal	COCOMO, SLIM,SEER-SEM
Modelos basados en el tamaño	Modelo de estimación formal	<i>Function point analysis</i> y técnicas basadas en desarrollo Agile partiendo de users stories
Estimación por grupos	Estimación por expertos	Técnicas <i>Delphi</i> o “ <i>planning poker</i> ”
Técnicas combinadas		Métodos basados en analogías y en <i>work breakdown</i> combinados

Como consecuencia de esta gran dificultad para estimar el coste de un desarrollo software, una gran parte de los nuevos desarrollos tienden a utilizar metodologías ágiles (*agile methodologies*) para el desarrollo [agil]. Estas metodologías tienen como objetivo esencial contener los costes de los desarrollos. El elemento esencial de las tecnologías ágiles es el

desarrollo de elementos en plazos temporales prefijados de antemano y asegurando que en cada una de las fases se desarrolla un producto que es, al menos parcialmente, operativo.

De esta forma se resuelve el problema del establecimiento del coste “a priori”. El problema se convierte en un caso de “*best effort*”: los desarrolladores intentan hacer el mejor producto posible dentro del tiempo que está previsto inicialmente. El producto final puede no cumplir, exactamente, las especificaciones ideales, pero se asegura que se cumplen los plazos.

Esta forma de desarrollo tiene cada vez más adeptos porque, en la práctica, las especificaciones no están perfectamente determinadas, salvo en los casos de sistemas muy conocidos y sobre los que existe mucha experiencia. En realidad, las especificaciones o requisitos del programa no son fijos sino que estos pueden modificarse. Los clientes que han solicitado un desarrollo no saben, muchas veces, qué es lo que realmente necesitan ni si lo que desean puede realizarse con la tecnología disponible.

Pero, evidentemente, no es posible utilizar estas tecnologías de estimación para el proyecto que nos ocupa porque ha sido desarrollado independientemente por varios alumnos de acuerdo con unas especificaciones más o menos determinadas. De acuerdo con las consideraciones anteriores, la estimación del coste del proyecto realizado se podría realizar simplemente indicando el tiempo que se ha empleado realmente en su desarrollo. Este tiempo se multiplica por un factor que tuviera en cuenta la experiencia (es decir el coste hipotético) de los desarrolladores, en este caso el alumno que ha desarrollado el trabajo.

En este proyecto se va a utilizar COCOMO. Esta técnica paramétrica se adapta bastante bien ya que utiliza variables, como el número de líneas y la experiencia de los programadores, que son fáciles de obtener. Además es una técnica probada y sobre la que se tiene bastante experiencia. Otros proyectos relacionados con *AulaWeb 2.0* utilizan una técnica conocida como *Use Case Points*. Esta es otra técnica paramétrica que está especialmente adaptada para los programas en los que existe una estructura Cliente Servidor. Se basa en estimar la dificultad de diseñar las distintas transacciones del modelo. En realidad es una técnica muy similar a COCOMO, pero el número de factores a estimar es más elevado por lo que la fiabilidad de los resultados es, probablemente, más baja. Este comentario es especialmente aplicable al módulo Alumno en el que las transacciones son, normalmente, sencillas y el coste estimado depende de parámetros cuya estimación tiene un alto grado de subjetividad.

## 4.2 COCOMO y técnicas de estimación paramétricas

El desarrollo de las técnicas paramétricas comenzó en los años 60 del pasado siglo, a medida que comenzaron a desarrollarse los primeros programas de ordenador. Una de las más conocidas y mejor documentadas son las técnicas denominadas COCOMO - *Constructive Cost Model* - que desarrolló la universidad de California del Sur (USC) por el profesor Barry Boehm [coc3]. Esta técnica paramétrica se ha ido adaptando con el paso del tiempo y ha derivado en varias implementaciones alternativas [coc2].

COCOMO tiene dos grandes ventajas: su simplicidad y la gran cantidad de información disponible para ajustar mejor los resultados. Además, gran parte de las soluciones más complejas que han elaborado muchos consultores (como las técnicas SEER [seer]) que utilizan algoritmos propietarios, suelen basarse en soluciones similares. Adicionalmente, COCOMO es la técnica usada por los otros proyectos relativos a *AulaWeb 2.0* por lo que su utilización en este caso permite cierto grado de uniformidad.

El principio básico que utiliza la técnica COCOMO es muy simple [coc2-3]) **El coste del desarrollo es proporcional, al número de líneas de código.** Sin embargo, el número de líneas de código es un indicador insuficiente. No todos los lenguajes de programación son igual de potentes ni tienen la misma simplicidad. Además, los requerimientos de los productos informáticos (fiabilidad, capacidad de relacionarse con otra aplicaciones,...) son diferentes. Por todo ello, el modelo de partida de COCOMO (y de otros métodos paramétricos) es proponer una función genérica que, posteriormente, se adapta y parametriza usando experiencias reales.

En el caso de COCOMO, la función de partida es relativamente simple [coc3]

$$PM = A \cdot (\sum lin)^{\Sigma^B} x \prod(EM) \quad (1)$$

Donde

- PM el número de personas hombre que supone el desarrollo en hombres mes PM o personas-mes o *person month*)
- A es un factor de calibración que depende del lenguaje y otros factores
- lin son las líneas de los distintos módulos del programa (en miles de líneas)
- B son distintos factores que tienen un efecto exponencial en el coste que se explican más adelante
- EM son factores que tienen un efecto multiplicativo en el coste. De nuevo estos factores se explican más adelante.
- $\Sigma$  y  $\prod$  son, respectivamente, los indicadores de sumatorio y de producto

Evidentemente, la anterior relación es enormemente versátil y su valor dependerá mucho de los distintos factores que se utilicen tanto en la parte exponencial, como en la parte multiplicativa.

La USC (Universidad del Sur de California) ha elaborado múltiples variantes de este método, con distintos parámetros, y basándose en su experiencia proporciona tablas para los distintos valores que pueden utilizarse. Esto da lugar a distintas versiones de COCOMO, y variantes del mismo.

En la mayoría de los casos, los parámetros de las distintas variantes de COCOMO se han validado por distintos expertos utilizando tecnología DELPHI. DELPHI es una metodología que permite incorporar las opiniones de los expertos de una forma que suele tender, muy

rápidamente, hacia un valor fijo. Esencialmente DELPHI es un proceso que funciona en dos fases:

- 1.- En una primera fase, los expertos, basándose en su experiencia, proporcionan una serie de indicaciones de los parámetros ajustables del modelo. Esta indicación la hacen exclusivamente utilizando su experiencia y sin saber lo que opinan otros expertos.
- 2.- A continuación, se informa a todos los expertos de los valores que han dado otros expertos, y se les pregunta si desean modificar sus estimaciones.

Normalmente, los expertos suelen modificar ligeramente sus estimaciones para aproximarse a una serie de valores de consenso.

Aun así, el proceso depende mucho de la experiencia de los expertos, del tipo de programas, etc. Por tal motivo, pueden existir muchas variantes de COCOMO, dependiendo de la metodología empleada. (tipo de parámetros, número de datos, uso de soluciones Delphi, cantidad de expertos consultados,...). Como se ha indicado, estos métodos varían en la forma en que se han obtenido los parámetros y, esencialmente en el número de ellos. La tabla 54 es un ejemplo de cómo distintos submodelos tienen diferentes parámetros.

**Tabla 54. Diferentes tipos de uso en COCOMO (fuente: [coc2])**

Nombre del modelo	tipo de estimación	No. factores aditivos	No. factores exponenciales	No. factores multiplicativos
COCOMO	Estimación de esfuerzos y duraciones	1	1	15
COCOMO II	Estimación de esfuerzos y duraciones	1	5	17
COSYSMO	Esfuerzo en ingeniería de sistemas	4	1	14
COCOTS	Sistemas off the self. integración	3	1	13
COSOSIMO	SoS (sistema de sistemas). Integración	4	6	--

En el caso del presente proyecto, se pueden aplicar dos tipos de soluciones basadas en COCOMO: La denominada básica y la denominada intermedia.

En la solución básica, la ecuación de partida es la más simple. No se utilizan elementos multiplicativos y solo se utiliza un elemento exponencial. En este caso, la expresión anterior (1) se simplifica y  $\sum B=b$  y  $A \cdot \prod(EM) = a$ , resultando en

$$PM = a(\sum lin)^b \quad (2)$$

Para el tiempo de desarrollo y del tamaño del equipo, se utilizan las siguientes expresiones

Tiempo de desarrollo TD = c (PM)<sup>d</sup>

Tamaño del equipo TE =  $\frac{PM}{TD}$

Donde los valores de a, b c y d son números empíricos dados por la tabla 55.

**Tabla 55. Valor de los parámetros según tipo de proyecto (fuente: [coc2])**

Tipo de proyecto	a	b	c	d
orgánico	2,5	1,05	2,5	0,38
semi aislado	3	1,12	2,5	0,35
embebido	3,6	1,2	2,5	0,32

Según sea un proyecto orgánico, semi aislado o embebido. La diferencia entre estos tres tipos de proyectos es la siguiente:

- a) Orgánicos - Se trata de proyectos pequeños, compuestos por equipos de desarrollos pequeños y con experiencia y con requerimientos poco rígidos
- b) Semi aislados - Con valores intermedios respecto a lo que se explica en los proyectos embebidos
- c) Proyectos embebidos - En este caso los proyectos serían más grandes, con requerimientos más estrictos y que tienen que combinar elementos hardware y software y otros tipos de requerimientos de operatividad

Es difícil determinar los valores adecuados para el proyecto que nos ocupa, puesto que las definiciones son muy amplias. Por un lado, el equipo de desarrollo no es muy grande, pero no tiene un alto grado de cohesión ni tiene experiencia. Además, los requerimientos son relativamente estrictos, ya que Aula Web 2.0 debe parecerse lo más posible a AulaWeb 1. Por tales motivos, se sugiere utilizar la versión semi-aislado. Otros proyectos dentro de AulaWeb suelen utilizar la versión orgánica.

Una versión algo más elaborada del modelo COCOMO, denominada versión intermedia, propone utilizar como punto de partida una fórmula para el cálculo de los esfuerzos una expresión en la que si se emplean elementos multiplicativos (EM) que corresponde a la expresión (1)

$$PM = Ax(\sum Lin)^{\Sigma^B}x \prod(EM) \quad (3)$$

Los valores de EM se toman de la tabla 56 (que, como en el caso anterior) es totalmente heurística y basada en la experiencia de múltiples proyectos

**Tabla 56. Valor de los parámetros (fuente: [coc2])**

Atributos	Valor					
	Muy bajo	Bajo	Nominal	Alto	Muy alto	Extra alto
<b>Atributos de software</b>						
Fiabilidad	0,75	0,88	1,00	1,15	1,40	
Tamaño de Base de datos		0,94	1,00	1,08	1,16	
Complejidad	0,70	0,85	1,00	1,15	1,30	1,65
<b>Atributos de hardware</b>						
Restricciones de tiempo de ejecución			1,00	1,11	1,30	1,66
Restricciones de memoria virtual			1,00	1,06	1,21	1,56
Volatilidad de la máquina virtual	0,87	1,00	1,15	1,30		
Tiempo de respuesta	0,87	1,00	1,07	1,15		
<b>Atributos de personal</b>						
Capacidad de análisis	1,46	1,19	1,00	0,86	0,71	
Experiencia en la aplicación	1,29	1,13	1,00	0,91	0,82	
Calidad de los programadores	1,42	1,17	1,00	0,86	0,70	
Experiencia en la máquina virtual	1,21	1,10	1,00	0,90		
Experiencia en el lenguaje	1,14	1,07	1,00	0,95		
<b>Atributos del proyecto</b>						
Técnicas actualizadas de programación	1,24	1,10	1,00	0,91	0,82	
Utilización de herramientas de software	1,24	1,10	1,00	0,91	0,83	
Restricciones de tiempo de desarrollo	1,23	1,08	1,00	1,04	1,10	

Dicha tabla se va a utilizar para realizar un análisis de sensibilidad.

Para la estimación de los costes medios por hora de los desarrolladores, se van a utilizar los datos de la referencia [des] (tabla 57).

**Tabla 57. Costes de desarrollador (fuente: [coc2])**

Costes medios desarrollador
Hora- 25-30 €
Consultas 70-90€ /hora
Jornada 100-150 € jornada

## 4.3 Valoración económica

La tabla 58 muestra el número de líneas de cada uno de los módulos de este trabajo

Para contar el número de líneas de código se ha empleado la aplicación CLOC que es una aplicación abierta que puede encontrarse en [cloc]

**Tabla 58. Costes finales (elaboración propia)**

`http://cloc.sourceforge.net v 1.64 T=0.13 s (341.7 files/s, 46878.8 lines/s)`

File	blank	comment	code
Proy\asignaturas.php	569	11	653
Proy\ficha_asignatura_vista.php	50	0	340
Proy\evaluaciones.php	111	21	311
Proy\eval_accordion.php	134	1	262
Proy\foros.php	46	15	228
Proy\grupos.php	28	11	159
Proy\evaluacion.php	86	1	153
Proy\ajaxfileupload.js	26	19	133
Proy\main.js	33	22	113
Proy\grupo.php	37	1	110
Proy\exámenes.php	37	1	94
Proy\ficha.js	29	3	93
Proy\foro.php	61	8	92
Proy\ficha.php	31	2	81
Proy\alumnos.php	22	7	76
Proy\cuestionario.php	30	1	73
Proy\foros.js	27	5	72
Proy\prof_asignat.php	14	5	68
Proy\cuestionarios.php	15	6	66
Proy\observacion.php	14	6	65
Proy\exámenes_(2).php	22	1	63
Proy\alumno_asignat.php	10	5	60
Proy\seguimiento.php	18	1	56
Proy\main.php	21	0	55
Proy\datos.php	23	2	55
Proy\observaciones.php	31	1	55
Proy\ficha_asignatura_vista.js	31	0	54
Proy\grupo_alumno.php	8	5	54
Proy\mensaje.php	20	0	53
Proy\acceso.php	21	2	50
Proy\profesores.php	8	5	48
Proy\contraseña_update.php	10	0	34
Proy\conexion.php	8	1	26
Proy\contraseña_update2.php	6	0	25
Proy\create_table.php	10	0	23
Proy\cuestionario.js	4	0	16
Proy\descargas2.php	5	0	16
Proy\descargas.php	11	1	13
Proy\grupo.js	7	10	11
Proy\contacto_update.php	3	0	6
Proy\nuevo_tema.php	4	0	6
Proy\resp_tema.php	5	0	6
Proy\usuario.php	0	1	5
SUM:	1686	181	4032

Language	files	blank	comment	code
PHP	36	1529	122	3540
Javascript	7	157	59	492
SUM:	43	1686	181	4032

Los resultados finales, utilizando la expresión simplificada, serán

$$PM = a(\sum Lin)^b \quad (4)$$

Tiempo de desarrollo =  $c (PM)^d$

Tamaño del equipo =  $\frac{PM}{TD}$

Dando valores numéricos, según los datos del apartado anterior y utilizando el valor de 4034 líneas se pueden obtener los siguientes resultados, mostrados en la tabla 59.

**Tabla 59. Costes finales (elaboración propia)**

	a	b	c	d	Esfuerzo (PM)	Tiempo Desarrollo (M)	Equipo (personas)	Coste (100€/jornada)	Coste (150€/jornada)
orgánico	2,5	1,05	2,5	0,38	10,81	6,18	1,75	21.626,82 €	<b>32.440,23 €</b>
semi-aislado	3	1,12	2,5	0,35	14,31	6,34	2,26	28.613,79 €	<b>42.920,69 €</b>

Como complemento, para mostrar la sensibilidad del modelo, se va a utilizar el modelo completo para un cálculo más detallado:

$$PM = Ax(\sum Lin)^{\Sigma B}x \prod(EM) \quad (5)$$

Implicaría multiplicar el valor anterior por un número que tendría en cuenta los factores de experiencia, etc.

Si se utilizan los valores de la tabla de parámetros del apartado anterior, los valores pueden modificarse ligeramente. Por ejemplo, parece razonable suponer que la aplicación debe tener una alta fiabilidad y que la BB.DD. es relativamente grande. Además, se supondrá que los programadores tienen muy baja experiencia ya que el desarrollo se realiza por personas que realizan un primer proyecto. La tabla 60 muestra un resumen de lo expuesto.

**Tabla 60. Valor de los parámetros (elaboración propia)**

Atributo Sw	valor	Factor
Fiabilidad	Alta	1,15
Tamaño base datos	Alta	1,08
<b>Atributos personal</b>		
Capacidad análisis	M. Bajo	1,46
Experiencia	M. Bajo	1,29
Calidad programadores	M. Bajo	1,42
Experiencia lenguaje	M. Bajo	1,14
Total (producto factores)		<b>3,79</b>

Obteniéndose ahora:

**Tabla 61. Costes con parámetros (elaboración propia)**

	Esfuerzo (PM)	Tiempo Desarrollo (M)	Equipo (personas)	Coste (100€/jornada)	Coste (150€/jornada)
orgánico	40,95	10,25	4,00	<b>81.893,62 €</b>	122.840,42 €
Semi-ais.	54,18	10,11	5,36	<b>108.350,97 €</b>	162.526,46 €

Esto significa que, en este caso, el esfuerzo es casi 4 veces superior y el tiempo de desarrollo se vería incrementado 1,6 veces, pasando a ser de 10,1 meses, que es la duración de un curso académico aproximadamente. En este caso, sin embargo, los costes de desarrollo deberían modularse para tener en cuenta las características de los programadores y sería preferible utilizar un valor algo inferior a 100€ diarios (que corresponden a 2000 €/mes) por un valor inferior, con lo que el coste total no sería tan diferente al caso anterior.

Como puede verse, el modelo COCOMO es bastante adaptable y, modificando algunos parámetros, los costes o la duración del proyecto puede variar sensiblemente.

En este caso, se prefiere mantener los valores nominales, aunque es interesante mostrar la gran sensibilidad y la dificultad de estimar exactamente la duración en el coste de un desarrollo de este tipo.

## 5. Conclusiones y Líneas de Futuro Desarrollo

### 5.1 Conclusiones Generales

*AulaWeb* resulta una herramienta de la ETSII que facilita el contacto entre profesores y alumnos. Proporciona información de asignaturas y calificaciones y permite acceder a foros de debate. Desde sus inicios, se ha utilizado en esta escuela más que cualquier otra plataforma de Tele-enseñanza. Se trata de una plataforma sencilla, práctica y fácil de utilizar. Además, está enfocada específicamente a los alumnos de la ETSII.

*AulaWeb2.0* pretende seguir la línea de *AulaWeb* pero siendo ésta desarrollada en lenguaje PHP y alojada en servidores Apache. Por ello, se ha diseñado la estructura desde el principio, eliminando incoherencias e incluyendo nuevas funcionalidades.

La nueva versión de *AulaWeb2.0* está dividida en módulos capaces de funcionar sin depender del funcionamiento de otros. De esta forma se facilita mantener una mejora continua. Por otro lado, se ha actualizado la Base de Datos: eliminando, editando y añadiendo tablas de forma que la información redundante sea mínima.

En resumen, se ha conseguido aprovechar la información de la plataforma *AulaWeb*, mejorando el diseño y facilitando su evolución, pero con unas características funcionales muy semejantes.

### 5.2 Conclusiones por objetivos

Del mismo modo, se incluye una tabla (tabla 62) con los objetivos específicos:

**Tabla 62. Cumplimiento objetivos específicos (elaboración propia)**

<i>Característica</i>	<i>Condiciones del problema establecidas por la dirección del proyecto</i>	<i>Cumplimiento</i>
<b>1. Diseño del módulo Alumno</b>	Debe incorporar todas las funcionalidades que tenía en la anterior <i>AulaWeb</i> , eliminando aquellas que no se utilizan.	Incluye información general del alumno, de la asignatura, contenidos, información de los grupos, foro, observaciones,... Y

		se ha eliminado el chat por su falta de uso.
<b>2. Diseño y modificación de la Base de Datos gestionada por el módulo del Alumno.</b>	Puesto que la BB.DD. va a mantenerse se debe proceder a la unificación de todos los identificadores y que esté libre de redundancias	Se ha diseñado una nueva BB.DD. tal y como se describe en apartados anteriores
<b>3. Diseño de la interfaz de acceso del alumno al nuevo módulo, sus hojas de estilo y su formato.</b>	El color que definirá dicho módulo será el azul. Y la apariencia debe ser similar a la de los módulos Profesor y Administrador.	Se ha modificado la estructura de la interfaz, haciéndola más atractiva, incluyendo una clave de colores
<b>4. Desarrollar utilizando herramientas de código libre cada una de las secciones y subsecciones de que consta el módulo de Alumno</b>	Se propone el uso de PHP y JavaScript.	Diseño de una estructura modular, tal y como estaba previsto. Utilización de los lenguajes PHP y JavaScript
<b>5. Desarrollar una nueva interfaz más “amable” de utilización por parte del alumno</b>	Incorporación de botones en lugar de pestañas en las distintas secciones y subsecciones.	Los botones se encienden y se apagan según en la sección y subsección que corresponde. Además se incluye una indicación de la sección donde se encuentra
<b>6. Construcción de nuevas tablas en la Base de Datos</b>	Las necesarias para el funcionamiento del módulo Alumno	Se han incluido y modificado alrededor de 25 tablas. De dichas tablas, la mitad son exclusivas del módulo alumno.
<b>7. Integrar la sección “contenidos” del módulo Profesor.</b>	Modificando las funcionalidades que eran privativas del profesor.	Además, se ha incorporado la opción de descarga que no incluía en el módulo Profesor.

<b>8. Integrar el módulo Alumno en el contexto global de la plataforma</b>	<p>El módulo Alumno debe estar plenamente integrado en la estructura <i>AulaWeb 2.0</i> que está en desarrollo y cuenta ya con algunos módulos operativos. En todo caso, al tratarse de una estructura modular, debe estar claramente diferenciado para permitir su depuración y eventual reparación de forma independiente.</p>	<p>El diseño del modulo está listo para su integración en la plataforma.</p>
--	--	--

### 5.3 Impacto y responsabilidad legal, ética y profesional

Aunque el desarrollo de la nueva versión *AulaWeb 2.0* tiene una funcionalidad muy similar al de la versión anterior y su desarrollo se basa en los resultados y experiencias derivadas de la misma, hay algunos aspectos profesionales y éticos que son significativos y que han guiado el desarrollo del proyecto.

El primero se refiere a las entrevistas realizadas con los profesores y alumnos para el diseño de la nueva interfaz. El desarrollo de estas entrevistas y las opiniones recibidas han sido determinantes para el diseño final. En gran medida el producto es el resultado de las mismas, así como de las opiniones de los profesores encargados de la dirección general del proyecto. Por ese motivo, su colaboración se ha destacado en los apartados correspondientes al diseño. De forma específica, el producto está basado en la versión anterior y, por tal motivo, es importante reflejar en las referencias este hecho y resaltar los trabajos de los profesores y alumnos que contribuyeron a desarrollar *AulaWeb* y los trabajos de diseminación y ajuste que llevaron a cabo.

En segundo lugar, la confidencialidad de los datos y su seguridad. El sistema *AulaWeb* contiene información que puede ser confidencial (calificaciones) o que está sujeta a propiedad intelectual (desarrollos de lecciones, trabajos,...etc.). Todos estos aspectos deben ser tenidos en cuenta en el desarrollo del sistema para asegurar la seguridad, la inviolabilidad de las bases de datos y la propiedad intelectual de los resultados. Como se indica en las conclusiones, la seguridad del sistema y su resistencia a fallos e intrusiones es uno de los aspectos que, posiblemente, deben estudiarse con un cierto detalle para asegurar que la información es suficientemente segura. En todo caso, y en lo que se refiere a la confidencialidad y propiedad intelectual, será preciso detallar, cuando la versión de *AulaWeb* esté plenamente operativa las condiciones legales a las que está sometida la información que se almacene y se utilice, así como las responsabilidades legales que pueden existir para la Universidad y para los usuarios derivadas de un mal uso de las mismas. Antes de proceder a su utilización, los usuarios deberían aceptar las condiciones de utilización de acuerdo con las condiciones de la ley de protección de datos [ldat].

Finalmente, el uso del código abierto presenta, como se ha indicado en la introducción una serie de restricciones relativas al uso y comercialización de los resultados. El programa PHP tiene una licencia de código abierto según la Open Source Initiative que debe ser respetada. De forma similar, el uso de JavaScript y de las bases de datos que emulan el SQL también tiene una serie de licencias de uso que deben ser reconocidas por los autores y por los usuarios del sistema. En el caso de *AulaWeb 2.0* los desarrollos se realizan con la denominada licencia Apache (Apache License). Se trata de una licencia de software libre permisiva creada por la Apache Software Foundation (ASF). La licencia Apache permite al usuario del software la libertad de usarlo para cualquier propósito, distribuirlo, modificarlo, y distribuir versiones modificadas de ese software pero requiere la conservación del aviso de derecho de autor y el descargo de responsabilidad. Además, no es una licencia copyleft, ya que no requiere la redistribución del código fuente cuando se distribuyen versiones modificadas.

De todos modos, como se indica, aunque la licencia Apache permite al usuario del software la libertad de usar el software se establece que, si el producto *AulaWeb* se distribuyera, deberían incluirse dos archivos en el directorio principal de los paquetes de software redistribuidos que incluyan [gnul]:

## LICENCIA

Copia de la licencia.

## AVISO (NOTICE)

Un documento de texto, que incluye los "avisos" obligatorios del software presente en la distribución y una copia de estas notificaciones debe ser distribuidas como parte de los trabajos derivados, dentro de la forma de código fuente o documentación, o dentro de una pantalla generada por las obras derivadas (donde aparecen normalmente este tipo de notificaciones a terceros).

Además de estas consideraciones legales, los desarrollos que utilizan software abierto y gratuito tienen, en cierto modo, una deuda social y su utilización debería tener una componente que devolviera a la sociedad parte de los beneficios que del uso de estas licencias se obtiene. En este caso, puesto que el objetivo es educativo, el beneficio social se cumple. Aun así, podría considerarse la publicación, en términos de software abierto, de algunos de los módulos desarrollados en la aplicación.

## 5.4. Líneas futuras

El trabajo “Interfaz de Alumno” de la herramienta *AulaWeb 2.0* constituye una parte del proyecto *AulaWeb 2.0*. Tanto los módulos Administrador como el Gestor de Contenidos han sido objeto de trabajos anteriores y este módulo se ha centrado en el Módulo del Alumno.

Para definir las actividades futuras, será preciso, en primer lugar, asegurar la plena integración entre los tres módulos y realizar una serie de pruebas de sistema y de integración. De esta forma se asegura que los tres módulos están plenamente integrados y todos los elementos funcionan con total complementariedad para asegurar que la herramienta es de fácil uso.

Una vez concluidas estas pruebas, será necesario un periodo de adaptación y convivencia con el sistema actual en el que un conjunto de usuarios específico *Beta Testers*, procedan a probar la nueva funcionalidad y elaboren una serie de sugerencia o informen de algunos errores que podrían aparecer una vez que la herramienta estén en producción.

Concluido este proceso, el sistema podría estar plenamente operativo para su utilización por profesores y alumnos de la ETSII. A partir de este momento, existen varias líneas de actuación que pueden ser interesantes:

1. Incrementar de forma significativa los contenidos, quizá con diferentes niveles de seguridad para que los alumnos pudieran acceder a contenidos internos y externos
2. Realizar una asociación más específica entre contenidos y valoraciones de los mismos de forma que las valoraciones permitan mayores sugerencias y propuestas por parte de los alumnos y profesores.
3. Conectar *AulaWeb 2.0* con otras herramientas LMS disponibles en la UPM para lograr un sistema plenamente integrado, que incluyera videos, grabaciones de las clases, posibilidad de realización de pruebas a distancia, etc.
4. Si la herramienta se utilizara como elemento de evaluación, podría ser conveniente aumentar su seguridad para evitar usos indeseados o prevenir ataques malintencionados o “*Denial of Service*”

El campo de posibilidades es muy grande. Como se ha indicado en los primeros capítulos de este trabajo, los sistemas de *tele-enseñanza* tienen un mercado potencial muy significativo. *AulaWeb 2.0* es una experiencia que permite capacitar a los alumnos de la ETSII para acceder a este mercado tan prometedor, al tiempo que es una herramienta de uso diario de profesores y alumnos de la ETSII.

## 6. Referencias

- [agil] BARNES P. *Software Costs Estimation In Agile Project Management*. Toptotal. [En línea] <https://www.toptal.com/agile/software-costs-estimation-in-agile-project-management> [Consulta: 1/11/2016]
- [amb] Ambient Insight. Report. <http://ambientinsight.com> [Consulta: 1/11/2016]
- [ard] ARDUINO. *What is Arduino?*. M. Banzi [en linea] <https://www.arduino.cc/> [Consulta: 1/11/2016]
- [asp] ASP.NET. *Open Source ASP.NET MVC, Web API, and Web Pages (Razor) are Open Source Projects*. [En línea] <http://www.asp.net/open-source> <http://www.asp.net/mvc/open-source> [Consulta: 1/11/2016]
- [aspv] KOHAN. B. PHP vs ASP.net ComparisoN. Comentum. [En línea] <http://www.comentum.com/php-vs-asp.net-comparison.html> [Consulta: 1/11/2016]
- [aula1] GARCÍA-BELTRÁN A. , R. MARTÍNEZ, J. A. CRIADO, J. A. MARTÍN, M. AZA, F. DE ORY, P. GARCÍA, D. MOLINA, L. M. PABÓN Y C. ZOIDO. *Un sistema de e-learning para la ETSII-UPM*, Indumática 12, 24-26 (2002).
- [aula2] A. GARCÍA-BELTRÁN, R. MARTÍNEZ, *AulaWeb: un sistema para la gestión, evaluación y seguimiento de asignaturas*, Industria, 2, 11-16, (2001).
- [aula3] R. MARTÍNEZ, A. GARCÍA-BELTRÁN, J.A. JAÉN, *Un sistema WWW de ayuda a la formación para alumnos y profesores*, Ingeniería I+D, 44, 327-329 (2000)
- [aula4] A. GARCÍA-BELTRÁN, R. MARTÍNEZ. *AulaWeb: Manual del administrador*, ETSII-UPM, 50pp, noviembre de 2002. ISBN: 84-688-5872-2
- [aula5] R. MARTÍNEZ, A. GARCÍA-BELTRÁN. *AulaWeb: Manual del profesor*, 94 pp, diciembre de 2002. ISBN: 84-7484-154-2
- [aula6] A. GARCÍA-BELTRÁN, R. MARTÍNEZ. *AulaWeb: Manual del alumno*, Secc. Publicaciones de la ETSII-UPM, 100 pp, junio de 2002. ISBN: 84-7484-148-8
- [cenatic] CENATIC. *Impacto de la reutilización del software de fuentes abiertas en la Economía*. [En línea] [http://observatorio.cenatic.es/index.php?option=com\\_content&view=article&id=810:dossier-cenatic-impacto-de-la-reutilizacion-del-software-de-fuentes-abiertas-en-la-economia&catid=94:tecnologia&Itemid=137](http://observatorio.cenatic.es/index.php?option=com_content&view=article&id=810:dossier-cenatic-impacto-de-la-reutilizacion-del-software-de-fuentes-abiertas-en-la-economia&catid=94:tecnologia&Itemid=137) [Consulta: 1/11/2016]
- [clie1] IGLESIAS C. *Entono PHP*. Slideshare. [En linea] <http://www.slideshare.net/CarlosIglesias3/entorno-php> . [Consulta: 1/11/2016] Original McLAUGHLIN.PHP & MySQL. O'Reilly Media. November 2012.
- [cloc] SOURCEFORGE. *Cloc* [https://sourceforge.net/projects/cloc/?source=typ\\_redirect](https://sourceforge.net/projects/cloc/?source=typ_redirect) [Consulta y descarga: 1/11/2016]
- [coc2] UNIVERSITY SOUTHERN CALIFORNIA *Cocomo*. [En línea] [http://csse.usc.edu/csse/research/COCOMOII/cocomo\\_main.html](http://csse.usc.edu/csse/research/COCOMOII/cocomo_main.html) [Consulta 16/1/2017]

[coc3] BOEHM Barry, VALERDI R., LANE J.A., BROWN. A.W "COCOMO® Suite Methodology and Evolution", CrossTalk, April 2005 Disponible en [http://csse.usc.edu/csse/research/COCOMOII/cocomo\\_papers.htm](http://csse.usc.edu/csse/research/COCOMOII/cocomo_papers.htm) [Consulta: 1/11/2016]

[des] FOROS DEL WEB. *Coste del programador PHP en España.* Varias contribuciones [En línea] <http://www.forosdelweb.com/f10/coste-hora-programador-php-espana-676855> [Consulta: 1/11/2016]

[EC1] EUROPEAN COMMISSION. *Open source software strategy.* [En línea] [http://ec.europa.eu/dgs/informatics/oss\\_tech/opensource/ossinec\\_en.htm](http://ec.europa.eu/dgs/informatics/oss_tech/opensource/ossinec_en.htm) [Consulta: 1/11/2016]

[eco1] GLENN, M . "The future of higher education: How Technology will shape learning". *Economist Intelligence Unit.* Report October 2008. Disponible en web: <[https://www.nmc.org/pdf/Future-of-Higher-Ed-\(NMC\).pdf](https://www.nmc.org/pdf/Future-of-Higher-Ed-(NMC).pdf)>

[elear] PAPPAS C. *Top eLearning Statistics and Facts For 2015.* E-learning Industry Association [en linea] <http://elearningindustry.com/elearning-statistics-and-facts-for-2015>. [Consulta: 1/11/2016].

[global] GLOBAL INDUSTRY ANALYST. Global e-learning market [en linea]. En: [http://www.prweb.com/releases/distance\\_learning/e\\_learning/prweb9198652.htm](http://www.prweb.com/releases/distance_learning/e_learning/prweb9198652.htm) [Consulta: 1/11/2016]

[gnu] GNU. ¿Qué es GNU? [en linea] <http://www.gnu.org/> [Consulta: 1/11/2016]

[gnufig] GNU, *Categorías de software libre y software que no es libre.* [En línea] <http://www.gnu.org/philosophy/categories.jpg> [Consulta: 1/11/2016]

[gnul] GNU Operating System. LICENCIAS. [En línea].

<https://www.gnu.org/licenses/licenses.html>. [Consulta: 20/1/2017]. Vease también : THE APACHE SOFTWARE FOUNDATION. Apache License. <http://www.apache.org/licenses/>. [Consulta: 20/1/2017].

[ldat] España. Ley Orgánica 15/1999, de 13 de diciembre, de Protección de Datos de Carácter Personal. *Boletín Oficial del Estado* num. 298, de 14 de diciembre 1999, núm. 298, p. 43088 a 43099

[linux] DESDELINUX. *Licencias de SW abierto.* Toro. L. [En linea ] <http://blog.desdelinux.net/sobre-las-licencias-y-los-perjuicios-al-codigo-aberto/ykpjm-jpg> [Consulta: 1/11/2016]

[ohw] DMOZ. Open directory project. *Open Source hardware projects.* [En linea: <[http://www.dmoz.org/Computers/Hardware/Open\\_Source](http://www.dmoz.org/Computers/Hardware/Open_Source)>] [Consulta: 1/11/2016]

[ontsi] ONTSI. Observatorio nacional de las telecomunicaciones y de la SI. *Las TIC en las empresas y microempresas españolas.* Ed 2012. [En linea] <http://www.ontsi.red.es/ontsi/es/estudios-informes/las-tic-en-las-empresas-y-microempresas-espa%C3%B1olas-edici%C3%B3n-2012> [Consulta: 1/11/2016]

[open] OPEN SOURCE Definicion Open source. Hibbets J. <https://opensource.com/resources/what-open-source> [Consulta: 1/11/2016]

[oss] OPEN SOURCE INITIATIVE. *About the Open Source Initiative.* Vidal N. [en linea] <https://opensource.org/> [Consulta 16/1/2017]

[rasp] RASPBERRY PI. *Get Started With Raspberry Pi.* Bate. A. [en línea] <https://www.raspberrypi.org> [Consulta: 1/11/2016]

[rit] CASARES J. et al., RIT (*Rochester Institute of Technology*). “The future of Teaching and Learning in Higher Education”. v 13, September 2011. Disponible en red: <[https://www.rit.edu/academicaffairs/sites/rit.edu.academicaffairs/files/docs/future\\_of\\_teaching\\_and\\_learning\\_reportv13.pdf](https://www.rit.edu/academicaffairs/sites/rit.edu.academicaffairs/files/docs/future_of_teaching_and_learning_reportv13.pdf)>

[seer] SEER. *Product Description*. [En línea] <http://galorath.com/> [Consulta 16/1/2017]

[tux] TUX FILES. "Free Software vs. Open Software". [en linea]  
<https://tuxfiles.wordpress.com/free-software-vs-open-source/> [Consulta: 1/11/2016]

[wikios] STANDISH Group International. "Free Open Source Software Is Costing Vendors \$60 Billion,". [En linea] <http://www.marketwired.com/press-release/free-open-source-software-is-costing-vendors-60-billion-new-standish-group-international-844462.htm>  
[Consulta 16/1/2017]

[wilic] OPEN SOURCE INITIATIVE. *Licenses & Standards*. [en linea].  
<https://opensource.org/licenses> [Consulta 16/1/2017]

[wse] MOLOKKEN, K. Jorgensen, M. "A review of software surveys on software effort estimation". *Empirical Software Engineering*, 2003. ISESE 2003. Abstract disponible en <http://ieeexplore.ieee.org/document/1237981/?reload=true&arnumber=1237981>

## Glosario de términos

**Campo:** representación de un atributo en una base de datos.

**E-learning:** Término que hace referencia al aprendizaje electrónico o asistido por ordenador.

**Hipertexto:** Sistema de organización y presentación de datos que se basa en la vinculación de fragmentos textuales o gráficos a otros fragmentos, lo cual permite al usuario acceder a la información no necesariamente de forma secuencial sino desde cualquiera de los distintos ítems relacionados.

**Internet:** Es un conjunto descentralizado de redes de comunicación interconectadas que utilizan la familia de protocolos TCP/IP, lo cual garantiza que las redes físicas heterogéneas que la componen funcionen como una red lógica única de alcance mundial. Sus orígenes se remontan a 1969, cuando se estableció la primera conexión de computadoras, conocida como Arpanet.

**JavaScript:** Es un lenguaje de programación interpretado, dialecto el estándar ECMAScript. Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico.

**Modelo-vista-controlador:** 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. Para ello el modelo-vista-controlador propone la construcción de tres componentes distintos que son el modelo, la vista y el controlador, es decir, por un lado define componentes para la representación de la información, y otro lado para la interacción del usuario. Este patrón de arquitectura de software se basa en las ideas de reutilización de código y la separación de conceptos, características que buscan facilitar la tarea de desarrollo de aplicaciones y su posterior mantenimiento.

**MySQL:** Sistema de gestión de bases de datos relacional, multihilo y multiusuario. MySQL se desarrolla como software libre en un esquema de licenciamiento dual por un lado GPL/GNU y por otro lado si una empresa quiere incorporarlo a sus productos deben comprar a la empresa una licencia específica.

**Navegador:** Programa para la visualización de documentos web. Se puede reducir a una forma muy simple como una aplicación que lee hipertexto permitiendo la navegación en internet.

**PHP:** Lenguaje de programación de uso general de código del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico. Puede ser usado en la mayoría de los

servidores web al igual que en casi todos los sistemas operativos y plataformas sin ningún costo.

**Servidor:** Es una aplicación en ejecución capaz de atender las peticiones de un cliente y devolverle una respuesta en concordancia. Los servidores pueden ejecutar en cualquier tipo de computadora, incluso en computadoras dedicadas a las cuales se les conoce individualmente como “servidor”.

**Subversión:** Es una herramienta de control de versiones open source basada en un repositorio cuyo funcionamiento se asemeja enormemente al de un sistema de ficheros. Es software libre bajo la licencia de tipo Apache/BSD.

## Índice de Tablas

Tabla 1. Ventajas y posibles dificultades del Código Abierto (elaboración propia).....	15
Tabla 2. Objetivos específicos del módulo Alumno (elaboración propia).....	16
Tabla 3. Ventajas de PHP (elaboración propia) .....	22
Tabla 4. Ventajas de JavaScript (elaboración propia).....	24
Tabla 5. ASP.net vs PHP (elaboración propia) .....	24
Tabla 6. Soluciones adoptadas para los servidores de algunas plataformas (elaboración propia) .....	26
Tabla 7. Requerimientos de la interfaz AulaWeb 2.0 (elaboración propia).....	27
Tabla 8. Arquitectura Software (elaboración propia).....	30
Tabla 9. Modificaciones realizadas en la Base de Datos (elaboración propia).....	32
Tabla 10. BB.DD. Tabla contenidos (elaboración propia).....	34
Tabla 11. BB.DD. Tabla Evaluaciones (elaboración propia).....	35
Tabla 12. BB.DD. Tabla horarios (elaboración propia) .....	36
Tabla 13. BB.DD. Tabla grupos (elaboración propia) .....	37
Tabla 14. BB.DD. Tabla grupo-asignatura (elaboración propia) .....	38
Tabla 15. BB.DD. Tabla grupo profesor (elaboración propia) .....	38
Tabla 16. BB.DD. Tabla grupo tipo (elaboración propia) .....	39
Tabla 17. BB.DD. Tabla Autoevaluación (elaboración propia).....	39
Tabla 18. BB.DD. Tabla exposición (elaboración propia).....	40
Tabla 19. BB.DD. Tabla prácticas (elaboración propia).....	41
Tabla 20. BB.DD. Tabla pruebas parciales (elaboración propia) .....	41
Tabla 21. BB.DD. Tabla trabajo en grupo (elaboración propia).....	42
Tabla 22. BB.DD. Tabla trabajo individual (elaboración propia).....	42
Tabla 23. BB.DD. Tabla alumnos (elaboración propia) .....	43
Tabla 24. BB.DD. Tabla Alumnos asignaturas (elaboración propia) .....	44

Tabla 25. BB.DD. Tabla profesores (elaboración propia) .....	45
Tabla 26. BB.DD. Tabla profesores-asignaturas (elaboración propia) .....	46
Tabla 27. BB.DD. Tabla tutorías (elaboración propia) .....	46
Tabla 28. BB.DD. Tabla cuestionarios (elaboración propia) .....	47
Tabla 29. BB.DD. Tabla respuestas a cuestionarios (elaboración propia).....	47
Tabla 30. BB.DD. Tabla respuesta a cuestionarios (elaboración propia) .....	48
Tabla 31. BB.DD. Tabla foro (elaboración propia) .....	49
Tabla 32. BB.DD. Tabla vista de foro (elaboración propia) .....	49
Tabla 33. Funcionalidad de AulaWeb (elaboración propia) .....	52
Tabla 34. Funcionalidad básica del Interfaz de acceso (elaboración propia).....	55
Tabla 35. Acciones error. Acceso al sistema (elaboración propia) .....	55
Tabla 36. Acciones de error. Datos personales (elaboración propia).....	57
Tabla 37. Acciones de error. Interfaz informativo de la asignatura (elaboración propia) .....	58
Tabla 38. Acciones usuario (elaboración propia).....	59
Tabla 39. Acciones de error (elaboración propia).....	59
Tabla 40. Acciones. Observaciones (elaboración propia).....	60
Tabla 41. Acciones de error. Observaciones (elaboración propia) .....	61
Tabla 42. Acciones de error. Calificaciones (elaboración propia) .....	61
Tabla 43. Acciones usuario calificaciones parciales (elaboración propia) .....	62
Tabla 44. Acciones de error. Calificaciones parciales (elaboración propia).....	63
Tabla 45. Acciones de error: evaluaciones (elaboración propia) .....	63
Tabla 46. Acciones de usuario: Cuestionario (elaboración propia) .....	64
Tabla 47. Acciones de error: Cuestionario (elaboración propia) .....	64
Tabla 48. Acciones de usuario. Comentarios. (elaboración propia) .....	66
Tabla 49. Acciones de error. Comentarios (elaboración propia) .....	66
Tabla 50. Acciones de usuario. Acceso a contenidos (elaboración propia) .....	68

Tabla 51. Acciones de error. Acceso a contenidos (elaboración propia) .....	68
Tabla 52. Descripción temporal de actividades (elaboración propia) .....	71
Tabla 53. Técnicas de estimación de coste de proyecto (fuente: [coc2]).....	73
Tabla 54. Diferentes tipos de uso en COCOMO (fuente: [coc2]).....	76
Tabla 55. Valor de los parámetros según tipo de proyecto (fuente: [coc2]) .....	77
Tabla 56. Valor de los parámetros (fuente: [coc2]) .....	78
Tabla 57. Costes de desarrollador (fuente: [coc2]) .....	78
Tabla 58. Costes finales (elaboración propia) .....	79
Tabla 59. Costes finales (elaboración propia) .....	80
Tabla 60. Valor de los parámetros (elaboración propia) .....	81
Tabla 61. Costes con parámetros (elaboración propia) .....	81
Tabla 62. Cumplimiento objetivos específicos (elaboración propia).....	82

## Índice de figuras

Figura 1. Datos del mercado Tele-enseñanza (elaboración propia a partir de datos de <i>E-learning Industry Association</i> ) .....	9
Figura 2. Distintas opciones del Software Abierto (fuente: Asociacion GNU) .....	13
Figura 3. Comparación entre Software libre, Open Source y privado. (fuente: blog.desdelinus.net).....	14
Figura 4. Uso del software abierto por la Comisión Europea (fuente EC) .....	15
Figura 6. Sistema Cliente Servidor (versión general y desarrollo en Apache, PHP/LAMP) (fuente: [clie1]).....	20
Figura 5. Modelo de comunicaciones entre ordenadores (fuente Wikipedia commons) .....	20
Figura 7. Estructura de programa AulaWeb 2.0. (Elaboración propia) .....	29
Figura 8. Esquema de la interfaz Alumnos. (elaboración propia).....	31
Figura 9. Esquema de la interfaz Alumnos. (elaboración propia).....	31
Figura 10. Actividades desde el modulo alumno (elaboración propia).....	51
Figura 11. Presentación de la interfaz (elaboración propia).....	52
Figura 12. Interfaz de acceso al sistema (elaboración propia) .....	54
Figura 13. Interfaz Datos personales (elaboración propia) .....	56
Figura 14. Interfaz informativo de la asignatura (elaboración propia) .....	57
Figura 15. Interfaz de información de asignaturas (elaboración propia) .....	58
Figura 16. Interfaz observaciones (elaboración propia).....	60
Figura 17. Interfaz calificaciones (elaboración propia) .....	61
Figura 18. Interfaz Calificaciones parciales (elaboración propia) .....	62
Figura 19. Interfaz evaluaciones (elaboración propia).....	63
Figura 20. Interfaz de usuario. Cuestionario (elaboración propia) .....	64
Figura 21. Interfaz Foro de la asignatura (elaboración propia) .....	65
Figura 22. Interfaz foro de la asignatura. Comentarios. (elaboración propia) .....	65

Figura 23. Interfaz de acceso a contenidos (elaboración propia) .....	67
Figura 24. Interfaz de acceso a contenidos (2) (elaboración propia) .....	67
Figura 25. Estructura de Descomposición del Producto (elaboración propia).....	70
Figura 26. Diagrama temporal GANTT (elaboración propia) .....	71

