



GRADO EN INGENIERÍA INFORMÁTICA

TRABAJO FINAL DE GRADO

Desarrollo de una web corporativa basada en WordPress

Autor:
Marina PIBERNAT SEGARRA

Supervisor:
Manuel GARCÍA DEL OJO
Tutor académico:
Antonio MORALES ESCRIG

Fecha de lectura: 21 de Noviembre de 2014
Curso académico 2013/2014

Resumen

En este documento describimos el proceso de desarrollo de una web corporativa. Partimos de un diseño proporcionado por el cliente para crear una plantilla a medida para el sistema de gestión de contenidos WordPress.

El desarrollo del proyecto está dividido en dos fases. Una primera fase en la que copiamos el diseño y lo implementamos con HTML, CSS y JavaScript. Y una segunda fase en la que añadimos a la web el panel de administración de WordPress utilizando PHP.

Tanto el desarrollo del sistema como el presente documento forman parte del proyecto de final de grado del Grado en Ingeniería Informática. Este trabajo ha sido desarrollado durante una estancia en practicas en *Culturaweb*, una empresa de Castellón dedicada al desarrollo de software. El proyecto ha sido desarrollado para uno de sus clientes: *Tungaloy*, una empresa japonesa dedicada a la producción y venta de metal y herramientas.

Palabras clave

Desarrollo web, PHP, Wordpress, PFG Ingeniería Informática.

Keywords

Web development, PHP, Wordpress, Final Project Computer Engineering Degree.

Índice general

Índice de figuras	7
1. Introducción	11
1.1. Contexto	11
1.2. Antecedentes y motivación	13
2. Descripción del proyecto y objetivos	15
3. Planificación del proyecto	17
3.1. Metodología	17
3.2. Definición de tareas	18
3.3. Planificación temporal de las tareas	20
3.4. Estimación de recursos	21
3.4.1. Recursos software/tecnologías utilizadas	21
3.4.2. Recursos hardware	24
3.4.3. Presupuesto	25
4. Análisis y diseño del sistema	27
4.1. Visión general del sistema	27
4.2. Análisis de requisitos	28
4.2.1. Requisitos de interfaz y navegación	28

4.2.2.	Historias de usuario	28
4.2.3.	Requisitos del entorno de explotación	30
4.2.4.	Requisitos de calidad	31
4.2.5.	Restricciones	31
4.3.	Diseño del sistema	31
5.	Implementación, pruebas y documentación	37
5.1.	Front-end	37
5.2.	Back-end	38
5.2.1.	Introducción a Wordpress	38
5.2.2.	Implementación en Wordpress	41
5.3.	Pruebas	52
5.3.1.	Pruebas de GUI y navegación	52
5.3.2.	Pruebas de funcionalidad o tests de aceptación	52
5.3.3.	Pruebas de compatibilidad	55
5.4.	Documentación	55
6.	Resultados y conclusiones	57
6.1.	Resultado temporal frente a planificación	57
6.2.	Conclusiones	59
	Glosario	61
A.	Ejemplos de código Wordpress	71
A.1.	Creación de <i>custom posts types</i>	71
A.1.1.	Creando el <i>custom post type Events</i>	71
A.2.	Creación de <i>shortcodes</i>	74

A.2.1. <i>Teaser shortcode</i>	75
A.2.2. <i>What's New shortcode</i>	77
A.2.3. <i>Tabs shortcode</i>	79
A.3. Creación de menús personalizados	83
B. Estructura de ficheros	87
C. Diseño proporcionado por el cliente	93
D. Informes quincenales	111
E. Pruebas de aceptación	117

Índice de figuras

1.1. Empresa de acogida	12
1.2. Empresa cliente	12
3.1. Diagrama de Gantt de fases del proyecto	21
3.2. Calendario y horario	21
3.3. Planificación inicial de tareas	22
3.4. Planificación documentación y presentación del TFG	22
3.5. Presupuesto recursos humanos	25
3.6. Presupuesto recursos tecnológicos	25
3.7. Presupuesto total	25
4.1. Diagrama de casos de usos	27
4.2. <i>Header</i>	28
4.3. Partes modificables del <i>footer</i>	29
4.4. Ejemplo de pestaña	29
4.5. Ejemplo de acordeón	29
4.6. Ejemplo de menú desplegable	30
4.7. Captura de parte de la <i>Home</i>	32
4.8. Captura de parte de la <i>Home</i>	32
4.9. Captura de parte de la página <i>Products</i>	33

4.10. Diseño lógico de la página <i>Products</i>	33
4.11. Captura de parte de la página <i>Worldwide Network</i>	34
4.12. Diseño lógico de la página <i>Worldwide Network</i>	34
4.13. Captura de parte de la página <i>Publications</i>	35
4.14. Diseño lógico de la página <i>Publications</i>	35
4.15. Captura de parte de la página <i>About Us</i>	36
4.16. Diseño lógico de la página <i>About Us</i>	36
5.1. <i>Home</i> para escritorio	38
5.2. <i>Home</i> para móviles	39
5.3. Tabla resumen de implementación	42
5.4. Implementación conceptual de la <i>Home</i>	44
5.5. Implementación conceptual de la página <i>About us</i>	46
5.6. Implementación conceptual de la página <i>Industries</i>	47
5.7. Implementación conceptual de la página <i>Products</i>	48
5.8. Implementación conceptual de la página <i>Worldwide Network</i>	49
5.9. Implementación conceptual de la página <i>Publications</i>	50
5.10. Implementación conceptual de la página <i>Contact</i>	51
5.11. Algunas historias de usuario	53
6.1. Planificación temporal inicial vs. Desarrollo temporal real	58
A.1. Código para crear el <i>custom post type events</i>	72
A.2. Ejemplo de <i>metabox</i>	73
A.3. Código para añadir <i>metaboxes</i> al <i>custom post type events</i>	74
A.4. <i>Shortcode teaser</i>	75

A.5. Código HTML para <i>teaser</i>	75
A.6. Código CSS para <i>teaser</i> escrito en LESS	76
A.7. Código PHP para crear el <i>shortcode teaser</i>	76
A.8. <i>Shortcode What's New</i>	77
A.9. Código HTML para <i>What's New shortcode</i>	78
A.10. Código PHP para crear el <i>shortcode What's New</i>	79
A.11. Bucle para leer datos de una consulta a la BDD de WordPress	79
A.12. <i>Shortcode tabs</i>	80
A.13. Código HTML para <i>Tabs shortcode</i>	80
A.14. Código PHP para <i>Tabgroup shortcode</i>	81
A.15. Código PHP para <i>Tab shortcode</i>	82
A.16. Menú personalizable (parte de color gris)	83
A.17. Código HTML para menú	83
A.18. Panel de administración	84
A.19. Código PHP para menú	84
A.20. HTML generado por WordPress	85
B.1. Jerarquía de ficheros completa	89
B.2. Jerarquía de la carpeta <i>assets</i>	90
B.3. Jerarquía de la carpeta <i>framework</i>	91
B.4. Jerarquía de la carpeta <i>languages</i>	91
E.1. Historias de usuario	117

Capítulo 1

Introducción

El desarrollo del proyecto que se presenta en este documento es el resultado del trabajo realizado para la asignatura *EI1054 Prácticas en Empresa y Proyecto de Fin de Grado* del Grado en Ingeniería Informática en la Universidad Jaume I de Castellón.

El objetivo principal de este proyecto es crear una web corporativa.

En este capítulo introductorio describimos el contexto en el que se ha desarrollado el proyecto y la motivación que nos ha llevado a su desarrollo.

En el resto del documento describimos el proceso de desarrollo llevado a cabo. Tomando como base el sistema de gestión de contenidos WordPress, hemos desarrollado un sistema web en PHP totalmente personalizado. Para el desarrollo del sitio hemos partido de un diseño proporcionado por el cliente para desarrollar un sistema que permite gestionar toda la información que la empresa desee poner a disponibilidad del público.

Finalmente, extraemos conclusiones sobre el resultado del proyecto y el proceso de su desarrollo, y comparamos la planificación temporal inicial frente al resultado temporal real analizando las posibles causas de desajustes.

Al final del documento podemos encontrar un glosario donde incluimos términos técnicos utilizados en este escrito.

1.1. Contexto

El PFG (Proyecto de Final de Grado) lo hemos desarrollado durante una estancia en prácticas en una empresa de acogida. Se trata de un proyecto encargado a esta empresa por uno de sus clientes.



Figura 1.1: Empresa de acogida



Figura 1.2: Empresa cliente

Empresa de acogida

La empresa de acogida es *Culturaweb* [10], un negocio de Castellón especializado en el desarrollo de todo tipo de plataformas web y móvil.

Culturaweb es una empresa joven de cuatro trabajadores. Fue creada en 2011 con el objetivo de ayudar a otras empresas a diseñar y construir sus marcas y productos en internet. Entre los servicios que ofrece la empresa encontramos: diseño gráfico, desarrollo de páginas web, aplicaciones para móviles, tiendas virtuales, posicionamiento en buscadores y asesoramiento para *startups*.

La empresa empezó su recorrido realizando trabajos para empresas locales, pero al poco tiempo amplió su mercado y en la actualidad trabaja con agencias publicitarias de Madrid y Barcelona.

La infraestructura técnica de la empresa está formada por una red local, un servidor Debian, un *switch*, un router y un servidor remoto CentOS. Por último, los equipos con los que cuenta la empresa para el desarrollo del trabajo son: cuatro iMacs, un MacBook Pro y un PC con Windows 8, el cual utilizamos durante la estancia en prácticas y para el desarrollo del proyecto.

Empresa cliente

La empresa cliente es *Tungaloy* [36], una empresa internacional especializada en la producción y venta de herramientas de corte de metal, productos de acero, herramientas de ingeniería civil, materiales de fricción y productos industriales.

Tungaloy fue fundada en 1934, con su sede principal en Japón, fue la primera empresa del país en tener éxito en el desarrollo de aleación superduro. La empresa es miembro de la IMC (*International Metalworking Companies*), una de las compañías metalúrgicas más grandes del mundo. La empresa, propiedad de Berkshire Hathaway, actualmente cuenta con plantas y oficinas en más de 25 países.

1.2. Antecedentes y motivación

La empresa cliente ya contaba con una web corporativa pero debido al paso del tiempo y a la necesidad, cada vez más fuerte, de las empresas de mantener una imagen digital cuidada y adecuada, se les hizo imprescindible renovar su web corporativa.

Por un lado, el uso de dispositivos móviles para navegar por Internet está creciendo a un ritmo increíble, y los dispositivos móviles ya generan casi un tercio del tráfico web [25]. Por este motivo cada vez más empresas están renovando sus sitios web para adaptarlas a la multitud de dispositivos desde los que pueden ser consultadas.

Por otro lado, la interacción con los clientes a través de redes sociales se ha vuelto muy importante para las empresas y se hace necesario contar con sitios web que permitan esta interacción con funcionalidad para compartir y comentar.

La motivación de este proyecto es adaptar la web corporativa de Tungaloy a estas nuevas necesidades.

Capítulo 2

Descripción del proyecto y objetivos

Como hemos dicho anteriormente, se trata de un proyecto desarrollado para la empresa japonesa Tungaloy, ésta contactó con la empresa de acogida Culturaweb para la renovación de su página web corporativa. A su vez, Culturaweb ofreció el proyecto a la Universidad Jaume I para que fuese realizado como proyecto de un estudiante de Ingeniería Informática, por este motivo todas las características y requerimientos del sistema, y por tanto del proyecto, han sido determinadas por el cliente.

El cliente pidió que a partir de un diseño que ellos nos proporcionarían se crease un sitio web *responsive*, es decir, que se adapte al tamaño del dispositivo con el que se accede, y completamente gestionable.

El *mockup* del diseño está disponible en el siguiente enlace:

http://www.tungaloy.co.jp/z_responsive_test5

En el *Anexo C Diseño proporcionado por el cliente* hemos incluido impresiones de cada una de las páginas del diseño, por si no se dispone de conexión a Internet o por si en un futuro el enlace deja de estar operativo.

Se trata de un *mockup* creado a partir de imágenes que muestra todas las páginas principales del sitio y ejemplos de algunas páginas secundarias.

El principal **objetivo de este proyecto** es crear una nueva web corporativa para la empresa que tenemos como cliente. Las características que el sitio web debe cumplir son las siguientes:

- Contenido completamente autogestionable, todo el contenido ha de poder ser modificado desde un panel de control sin necesidad de conocimientos de programación o HTML.
- Diseño web adaptable, la apariencia de la web ha de adaptarse al dispositivo que se esté utilizando para visualizarla.
- Gestión de idiomas de forma sencilla.

- Diseño a medida proporcionado por el cliente.

Esta es la única información que nos ha proporcionado el cliente, a partir de ella, en los puntos *4.1. Visión general del sistema* y *4.2. Análisis de requisitos*, se definen los objetivos técnicos concretos y los requisitos del sistema que nos llevarán a conseguir el objetivo principal del proyecto.

Una vez logrado el objetivo y construido el sistema, el cliente espera un beneficio llegando a los siguientes puntos:

- Mejorar la imagen de la empresa y su presencia en Internet.
- Mejorar la usabilidad de la actual web corporativa.
- Mejorar su posicionamiento SEO.
- Mejorar la organización de contenidos.
- Conseguir y fidelizar más clientes.
- Acceder a nuevos mercados.
- Proporcionar un de canal de comunicación.
- Mostrar información de interés siempre actualizada.
- Dar a conocer promociones.
- Destacar sobre la competencia.

No se han proporcionado más datos o requisitos del tipo: *como usuario necesito añadir productos*, estos han de ser extraídos como parte del proyecto a partir del diseño proporcionado. En el apartado *4.1. Análisis y diseño de software* detallamos los requisitos extraídos, que si bien podrían haber sido incluidos en la descripción del proyecto, los hemos incluido en el apartado de análisis ya que son el resultado de un estudio y análisis del prototipo del cliente.

Capítulo 3

Planificación del proyecto

Para toda la planificación y el desarrollo del proyecto diferenciamos **dos grandes partes del proyecto**:

- Parte *front-end*, traducible al español como interfaz, es la parte del software que interactúa con el usuario, haciendo referencia a la parte que visualiza el usuario navegante.
- Parte *back-end*, traducible como motor, es la parte que procesa la entrada desde el *front-end*, también hace referencia a la parte de administración del sitio.

Estas dos partes dividen el desarrollo del proyecto en dos fases. Una primera fase de desarrollo *front-end* en la que replicamos el *mockup* del cliente utilizando HTML, CSS y JavaScript. Y una segunda fase *back-end* en la que agregaremos un panel de administración y una base de datos a la web para que pueda ser modificada por usuarios sin conocimientos técnicos. Para esta segunda fase utilizamos PHP, y MySQL para la base de datos.

Para el desarrollo de la parte *front-end* empezaremos desde cero, en cambio en la parte *back-end* partiremos de una plantilla desarrollada por la empresa. En el *Anexo B Estructura de ficheros* detallamos el contenido de esta plantilla.

3.1. Metodología

En cuanto a la metodología el proyecto se caracteriza por su desarrollo incremental con reuniones con el cliente según la disponibilidad del mismo. Realizamos reuniones con la finalidad de comprobar el avance del proyecto y de discutir impresiones y posibles mejoras.

Aunque el desarrollo del proyecto ha sido llevado a cabo por un único miembro, y por tanto no se ha podido seguir una metodología de desarrollo ágil al completo, hemos utilizado una metodología basada en los principios de desarrollo ágil. Uno de los puntos en los que se hace mayor incidencia en cualquier metodología ágil es en la gestión del trabajo en equipo, nosotros

hemos dejado de lado este aspecto y hemos creado nuestra propia metodología adoptando parte de varios métodos de desarrollo ágil.

Basándonos en los principios del *Manifiesto por el Desarrollo Ágil del Software*[20] en nuestra metodología tenemos especialmente en cuenta los siguientes puntos que coinciden con el manifiesto:

- Colaboración con el cliente. Durante el proceso, mantenemos una comunicación fluida para que conozca el estado y la evolución del proyecto.
- Respuesta ante el cambio sobre seguir un plan. Aceptamos que los requisitos cambien, incluso en etapas tardías del desarrollo.
- Software funcionando sobre documentación extensiva. En lugar de escribir extensa documentación, sobre lo que el sistema permitirá y no permitirá, realizamos entregas continuas de software funcionando, permitiendo al cliente aprender sobre lo que hemos desarrollado hasta el momento, y pudiendo así detectar nuevas necesidades o modificaciones necesarias.

En nuestra metodología el cliente está completamente integrado en el proyecto, gracias a ello tenemos su opinión sobre el estado del proyecto en tiempo real, una información muy valiosa para comprobar que el proyecto va bien en todo momento.

Llevamos la retroalimentación al máximo mediante la elaboración de informes diarios en los que explicamos el estado del proyecto y donde el cliente pide pequeñas modificaciones o nos guía hacia nuevos requisitos, también hemos utilizado reuniones cortas con el cliente para comentar aspectos de mayor envergadura que necesiten comunicación oral. Este método de trabajo nos ayuda a conocer la fase de desarrollo en la que nos encontramos en todo momento y a ajustar los esfuerzos para introducir mejoras en el futuro.

Como en todas las metodologías ágiles, los requisitos son muy flexibles. En lugar de intentar definir todos los requisitos al comienzo del proyecto e invertir esfuerzos después en controlar los cambios en los requisitos, intentamos una aproximación más realista que nos permita adaptarnos a los cambios en cualquier momento. Poniendo más énfasis en la adaptabilidad que en la previsibilidad.

En cuanto a las pruebas y validación realizamos tests de aceptación junto al cliente al añadir nuevas funcionalidades y realizamos otras pruebas finales como detallamos en el punto 5.3. *Pruebas*.

3.2. Definición de tareas

Como hemos dicho anteriormente el proyecto se desarrolla en dos fases: programación *front-end* y programación *back-end*.

En la primera fase, programación *front-end*, copiamos el diseño del cliente y obtenemos una web completa de cara al usuario final pero no modificable, es decir, sin panel de administración. En esta fase tenemos en cuenta los siguientes puntos acordados con el cliente:

- Desarrollo utilizando HTML5, CSS3 y JavaScript.
- Diseño web responsive para PC, tablets y smartphones.
- Mecanismos de posicionamiento SEO: palabras claves y descripciones.
- Compartición en redes sociales: Facebook, Twitter, Google+ y enviar a un amigo.
- Diseño a medida proporcionado por el cliente.
- *Sliders* y carruseles de imágenes.
- Acordeones y otros elementos interactivos para mejorar la experiencia del usuario.
- Compatibilidad con todos los navegadores de PC actuales: IE 7 o superior, Mozilla Firefox, Google Chrome, etc.
- Compatibilidad con todos los navegadores móviles actuales: Safari, Google Chrome, Firefox, etc.

En la segunda fase, programación *back-end*, utilizamos un sistema de gestión de contenidos (CMS) para crear la parte de administración que permitirá al usuario crear y administrar contenidos. El sistema de gestión de contenidos escogido por la empresa de acogida para construir el sitio web de Tungaloy es Wordpress (en el apartado 5.2.1. *Introducción a Wordpress* lo conoceremos más a fondo). Las ventajas de utilizar este CMS son las siguientes:

- Facilidad de uso por personas no técnicas.
- Permite la creación y distribución de contenidos de una forma rápida y visual.
- Puede ser instalado en todos los principales *hostings*.
- Es rápido, fiable y escalable para soportar gran número de visitas.

Además de estas ventajas, la empresa cuenta con experiencia utilizando este CMS y posee un *know-how* que nos permite desarrollar un sistema de calidad de forma rápida y eficiente.

En esta fase unimos la web creada en la primera fase con el panel de administración que nos proporciona Wordpress. Los puntos a tener en cuenta son:

- Creación de un tema *ad-hoc* para Wordpress mediante programación PHP.
- Creación de un sistema multilinguaje: Inglés, Japonés, etc.
- Programación *ad-hoc* de *shortcodes* para gestionar contenido fácilmente.
- Sistema de almacenamiento en caché para mejorar el rendimiento del sitio web.
- Módulo *newsletter*: a través de MailChimp o con el propio sistema del cliente.
- Módulo SEO: administrar la configuración de SEO (título, palabras clave y descripciones) de cada sección.

- Módulo de últimas entradas: permite incluir las últimas entradas del blog por categorías en la barra lateral de cada página.
- Módulo de contacto: permite a los usuarios comunicarse con la empresa, el módulo enviará respuestas automáticas a los usuarios, y las consultas serán enviadas a diferentes destinatarios en función del departamento seleccionado.
- Módulo del blog: con entradas, categorías, comentarios y elementos sociales.
- Módulo de entradas relacionadas: este módulo mejora la navegación entre las entradas del blog y ayuda a reducir el rebote de la página.
- Módulo de feedback: permite a los usuarios hacer comentarios sobre las entradas del blog.
- Módulo anti spam: detecta y bloquea los comentarios de spam.
- Optimización de Wordpress para alto rendimiento: para permitir un gran número de visitantes.

Una vez terminadas las dos fases de desarrollo ya tenemos el sistema completo, entonces, pasamos a desplegar el proyecto en el servidor de producción, para ello llevamos a cabo las siguientes tareas:

- Mover el sistema al *hosting* de producción.
- Redirección 301 para mantener el posicionamiento x: páginas e imágenes.
- Registrar el sitio en Google Analytics y Google Webmaster o creación de un auto sistema estadístico. En el último caso el cliente es dueño de las estadísticas generadas Piwik.
- Creación de *XML Sitemaps* para mejorar la *Search Engine Indexing*.
- Seguimiento del proceso de migración de la página web en los motores de búsquedas. El cliente debe proporcionar todas las contraseñas necesarias para mover el sistema al servidor de producción.

El *hosting* de producción pertenece a la misma empresa cliente, el sistema estará alojado en uno de sus servidores en Japón. La web será gestionada por su propio departamento de informática, al que formaremos sobre el uso del sistema a través de una sesión explicativa.

3.3. Planificación temporal de las tareas

Desarrollamos el proyecto durante 300 horas de trabajo repartidas en días de trabajo de 6 a 8 horas. Empezamos el día 16 de Junio de 2014 y estimamos terminar el día 4 de Agosto, suponiendo que hacemos 8 horas al día, es decir, 40 horas semanales.

En la Figura 3.1 podemos observar un diagrama de Gantt con la estimación del esfuerzo en horas para cada fase. En la Figura 3.3 detallamos la planificación y la mostramos con mayor

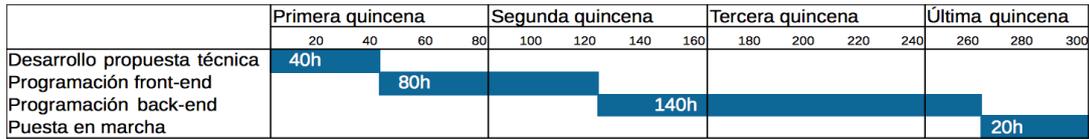


Figura 3.1: Diagrama de Gantt de fases del proyecto

detalle. Indicamos la estimación en horas para cada tarea y las fechas de realización previstas. Para realizar las estimaciones hemos utilizado el método basado en el juicio experto[18]. En la figura vemos que estimamos 40 horas para el desarrollo de la propuesta técnica, incluyendo tareas como la definición del proyecto, los métodos de trabajo, etc. El resto de las 300 horas se prevé dedicarlas al desarrollo del sistema, utilizando 80 horas para la parte de *front-end* y 140 horas para las tareas de *back-end*.

Días semanales: Lunes a Viernes

Horario diario: Variable 6 - 8 horas

Fecha de inicio: 16 de Junio de 2014

Fecha de finalización estimada: 4 de Agosto de 2014

Figura 3.2: Calendario y horario

En el apartado 6.1. *Resultado temporal frente a planificación* contrastamos la planificación inicial de tareas con el tiempo que realmente hemos necesitado para cada tarea.

3.4. Estimación de recursos

3.4.1. Recursos software/tecnologías utilizadas

En la parte *front-end* hemos utilizado las siguientes tecnologías:

- HTML5[40]: la quinta revisión del conocido HTML (*HyperText Markup Language*), el lenguaje básico de la *World Wide Web*.
- CSS3[38]: para controlar el estilo y los layouts de múltiples páginas HTML a la vez, separando así la estructura del diseño.
- LESS[22]: es un lenguaje pre-procesador de CSS que lo extiende, añadiéndole características como la creación de variables, funciones y muchas otras técnicas que le permiten hacer CSS más fácil de mantener y extender.
- Bootstrap[6]: un *framework* HTML, CSS y JavaScript para el desarrollo de la parte *front-end* de sitios y aplicaciones web. Contiene plantillas de diseño con tipografías, botones, menús de navegación, pestañas, etc. En este proyecto hemos utilizado los componentes

		Tiempo (h.)	Fecha inicio	Fecha fin
1	DESARROLLO PROPUESTA TÉCNICA	40 h	16/jun	20/jun
1.1	Definir el proyecto	2	16/jun	16/jun
1.2	Definir método de trabajo y documentación	1	16/jun	16/jun
1.3	Identificar alcance y objetivos	3	16/jun	16/jun
1.4	Análisis de requisitos y diseño de software	16	16/jun	18/jun
1.6	Definir tareas y estimar fechas	2	18/jun	18/jun
1.7	Documentar la propuesta del proyecto	16	18/jun	20/jun
2	DESARROLLO TÉCNICO DEL PROYECTO	260 h	20/jun	04/ago
2.1	Programación front-end	(80h)	23/jun	04/jul
2.1.1	Introducción a <i>HTML5</i> , <i>CSS3</i> y <i>JavaScript</i>	16	23/jun	24/jun
2.1.2	Analizar diseño y señalar patrones	2	25/jun	25/jun
2.1.3	Programar el <i>header</i>	6	25/jun	25/jun
2.1.4	Programar el <i>footer</i>	4	26/jun	26/jun
2.1.5	Programar la página " <i>home</i> "	5	26/jun	27/jun
2.1.6	Programar la página " <i>about us</i> "	6	27/jun	27/jun
2.1.7	Programar la página " <i>industries</i> "	1	27/jun	27/jun
2.1.8	Programar la página " <i>products</i> "	5	30/jun	30/jun
2.1.9	Programar la página " <i>worldwide network</i> "	7	30/jun	01/jul
2.1.10	Programar la página " <i>publications</i> "	6	01/jul	02/jul
2.1.11	Programar la página " <i>contact</i> "	2	02/jul	02/jul
2.1.12	Programar funcionalidad " <i>responsive</i> "	2	02/jul	02/jul
2.1.13	Comprobar compatibilidad con distintos navegadores, dispositivos y SO	2	02/jul	02/jul
2.1.14	Revisión y limpieza de código	16	03/jul	04/jul
2.2	Programación back-end	(140h)	07/jul	30/jul
2.2.1	Introducción a <i>PHP</i> y <i>Wordpress</i>	8	07/jul	07/jul
2.2.2	Crear tema <i>ad-hoc</i> para <i>Wordpress</i>	(80h)	08/jul	21/jul
2.2.2.1	Analizar contenidos y señalar patrones	8	08/jul	08/jul
2.2.2.2	Programar los nuevos <i>post types</i> necesarios	8	09/jul	09/jul
2.2.2.3	Programar el <i>header</i> para <i>Wordpress</i>	4	10/jul	10/jul
2.2.2.4	Programar el <i>footer</i> para <i>Wordpress</i>	8	10/jul	11/jul
2.2.2.5	Programar la página " <i>home</i> " para <i>Wordpress</i>	5	11/jul	14/jul
2.2.2.6	Programar la página " <i>about us</i> " para <i>Wordpress</i>	5	14/jul	15/jul
2.2.2.7	Programar la página " <i>industries</i> " para <i>Wordpress</i>	5	15/jul	15/jul
2.2.2.8	Programar la página " <i>products</i> " para <i>Wordpress</i>	5	16/jul	16/jul
2.2.2.9	Programar la página " <i>worldwide network</i> " para <i>Wordpress</i>	10	16/jul	17/jul
2.2.2.10	Programar la página " <i>publications</i> " para <i>Wordpress</i>	7	17/jul	18/jul
2.2.2.11	Programar la página " <i>contact</i> " para <i>Wordpress</i>	6	18/jul	18/jul
2.2.2.12	Diseñar y programar páginas que no constan en el diseño	9	18/jul	21/jul
2.2.3	Crear módulo de últimas entradas	1	19/jul	21/jul
2.2.4	Crear módulo de contacto	3	22/jul	22/jul
2.2.5	Crear módulo del blog	3	22/jul	22/jul
2.2.6	Crear módulo de entradas relacionadas	1	22/jul	22/jul
2.2.7	Crear módulo de <i>feedback</i>	2	22/jul	23/jul
2.2.8	Crear sistema multilinguaje	1	21/jul	23/jul
2.2.9	Crear módulo <i>newsletter</i>	1	23/jul	23/jul
2.2.10	Crear módulo SEO	1	23/jul	23/jul
2.2.11	Crear módulo anti <i>spam</i>	1	23/jul	23/jul
2.2.12	Optimizar <i>Wordpress</i> para alto rendimiento	6	23/jul	24/jul
2.2.13	Revisión, limpieza de código, pruebas y documentación	40	24/jul	31/jul
2.3	Puesta en marcha	(20h)	31/jul	04/ago
2.3.1	Mover el hosting al sistema de producción	4	31/jul	31/jul
2.3.2	Redireccionar 301	4	31/jul	01/ago
2.3.3	Poner en marcha sistema estadístico	4	01/ago	01/ago
2.3.4	Crear <i>XML Sitemaps</i>	4	01/ago	04/ago
2.3.5	Seguir el proceso de migración	4	04/ago	04/ago
	TOTAL	300 h		

Figura 3.3: Planificación inicial de tareas

		Tiempo (h.)
3	DOCUMENTACIÓN Y PRESENTACIÓN DEL TFG	135 h
3.1	Redactar informes quincenales	4
3.2	Redactar la memoria técnica	100
3.3	Preparar la presentación oral	30
3.4	Presentación oral	1

Figura 3.4: Planificación documentación y presentación del TFG

que nos proporciona Bootstrap adaptándolos a los requerimientos de diseño del cliente. Hemos escogido Bootstrap sobre otras opciones porque, aunque no es tan avanzado como otros, es el *framework* de su tipo que ofrece mayor compatibilidad con versiones antiguas de navegadores.

- JavaScript[14]: utilizado para la parte del cliente, es un lenguaje interpretado integrado en las páginas webs y ejecutado por los navegadores modernos.
- JQuery[17]: es una biblioteca JavaScript que permite simplificar la manera de interactuar con los documentos HTML, nos facilita la manipulación del árbol DOM, manejar eventos, etc.

Para la parte *back-end* hemos empleado:

- WordPress: un sistema de gestión de contenidos libre y gratuito (en el apartado 5.2.1. *Introducción a Wordpress* lo conocemos más a fondo).
- PHP[33]: un lenguaje de programación del lado del servidor que utilizaremos para el desarrollo web de contenido dinámico. Es el lenguaje que se utiliza para programar Wordpress.
- MySQL[24]: el sistema de gestión de base de datos que utiliza Wordpress, es un sistema de gestión de base de datos relacionales, multihilos, y multiusuario.

Respecto a las herramientas que hemos utilizado para desarrollar el proyecto son las siguientes:

- Aptana Studio[5]: un entorno de desarrollo integrado para el desarrollo web. Es de software libre y está basado en el IDE Eclipse[31].
- Prepros[28]: una herramienta para el desarrollo web que hemos utilizado para compilar código de LESS a CSS y para la concatenación de ficheros JavaScript.
- Git[12]: un sistema de control de versiones distribuido que permite mantener una gran cantidad de código caracterizándose por su eficiencia. Git realiza el control de versiones en repositorios locales.
- Bitbucket[4]: es un servicio de alojamiento basado en la web para proyectos que utilicen el sistema de control de versiones Mercurial o Git.
- FTP[21]: Protocolo de Transferencia de Archivos, lo hemos utilizado para transferir el proyecto a un servidor y hacerlo disponible para el cliente.

Por último, para la elaboración de este documento hemos hecho uso de las siguientes herramientas:

- Latex[35]: un sistema de composición de textos orientado a la creación de documentos de gran calidad tipográfica. Lo hemos utilizado para la redacción de este documento.

- Cacao[8]: es una herramienta online para la creación de diagramas.
- Libreoffice[30]: una suite ofimática libre y de código abierto. Hemos utilizado sus hojas de cálculo para generar tablas.

3.4.2. Recursos hardware

Para el desarrollo del proyecto nos hemos servido de un ordenador de sobremesa, un servidor local para almacenar el proyecto y una tableta para pruebas. A continuación detallamos las características principales de cada equipo.

Ordenador de sobremesa:

- Procesador: 4th Generation Intel Core i5-4460.
- Sistema operativo: Windows 8.1 Pro.
- Memoria RAM: 8GB1 Dual Channel DDR3 SDRAM a 1600MHz.
- Disco duro: 1TB 7200 RPM SATA Hard Drive 6.0 Gb/s.
- Tarjeta gráfica: NVIDIA GeForce GT 720 1GB DDR3.

Servidor:

- Procesador: Intel Xeon (4 núcleos, 3,1 GHz, 8 MB, 69 W).
- Sistema Debian.
- Memoria RAM 8 GB.

Tableta:

- Procesador Intel a 1.6 GHz.
- Sistema operativo: Android JellyBean 4.2.
- Memoria RAM de 2 GB.
- Capacidad de almacenamiento de 32 GB.
- Gráficos SGX544MP2.

Función	Euros/hora	Horas	Precio
Analista y diseñador	28	40	1.120,00 €
Programador <i>front-end</i>	21	80	1.680,00 €
Programador <i>back-end</i>	26	140	3.640,00 €
TOTAL			6.440,00 €

Figura 3.5: Presupuesto recursos humanos

	Precio de adquisición	Tiempo de vida útil	Horas de uso para el proyecto	Precio por horas de uso
Ordenador de sobre mesa	520,00 €	3 años	300 horas	5,94 €
Hosting de desarrollo	780,00 €	5 años	300 horas	5,34 €
Tableta Android	290,00 €	2 años	20 horas	3,35 €
TOTAL				14,63 €

Figura 3.6: Presupuesto recursos tecnológicos

3.4.3. Presupuesto

El presupuesto estimado para el desarrollo del proyecto incluye las horas de trabajo a dedicar y el precio proporcional a las horas de uso de los equipos que utilizamos según su tiempo de vida útil. Todo el software que utilizamos es libre o gratuito, por tanto no necesitamos licencias que debamos incluir en el presupuesto.

Para establecer el presupuesto necesario para los recursos humanos partimos del precio por hora para cada una de las funciones que desempeñamos. Hemos recogido el precio por hora para cada función de varias estadísticas realizadas por *InfoJobs*[13], *Bureau of Labor Statics*[7] y *U.S. News*[37]. En la Figura 3.5 encontramos en presupuesto estimado para los recursos humanos.

En la Figura 3.6 vemos el presupuesto para los recursos tecnológicos. A partir del precio de los equipos que utilizamos y de la vida útil de cada uno[9], calculamos el precio proporcional según las horas de uso del equipo en el proyecto.

Por último, en la Figura 3.7 podemos observar el presupuesto total, suma de las dos estimaciones anteriores.

	Precio
Recursos humanos	6.440,00 €
Recursos tecnológicos	14,63 €
TOTAL	6.454,63 €

Figura 3.7: Presupuesto total

Capítulo 4

Análisis y diseño del sistema

En este capítulo presentamos el análisis de requisitos del sistema y el diseño lógico del mismo, los detalles técnicos y de implementación quedan para el siguiente capítulo.

4.1. Visión general del sistema

Antes de empezar ofreceremos una visión general del sistema mediante un diagrama de casos de uso. La Figura 4.1 nos da una idea general de la funcionalidad del sistema.

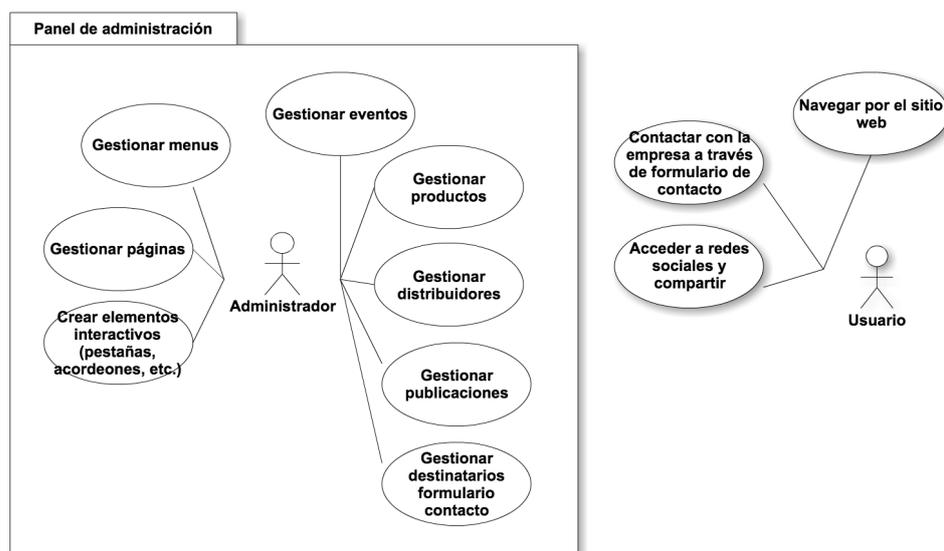


Figura 4.1: Diagrama de casos de usos

Podemos observar la existencia de dos actores: administrador y usuario. Como en cualquier web, el administrador tiene acceso a todas las funciones del panel de administración desde donde puede modificar la web, y el usuario sólo tiene permitido consultar e interactuar con la web.

4.2. Análisis de requisitos

Como parte del proyecto realizamos un análisis de requisitos donde identificamos todas las características o condiciones que el sistema ha de cumplir. De esta manera logramos comprender mejor el problema a resolver y podemos definir la solución informática más adecuada.

Los requisitos los extraemos a partir del análisis del *mockup* proporcionado por el cliente (disponible en el *Anexo B*). Se trata de un proceso de ingeniería inversa en el obtenemos información a partir de un diseño del producto ya construido. En los siguientes apartados presentamos los requisitos obtenidos.

4.2.1. Requisitos de interfaz y navegación

En este proyecto los requisitos de interfaz y navegación están muy claros: la interfaz del sistema tiene que ser exactamente como se muestra en el diseño, del mismo modo, la navegación por el sistema en la parte del usuario ha de copiar la navegación que se muestra en el *mockup*. Además, la interfaz ha de ser *responsive*.

4.2.2. Historias de usuario

Historias de usuarios de la parte del panel de administración:

- **HU1** Como administrador necesito añadir y eliminar enlaces del *header*.
 - Sólo se podrán modificar los enlaces de la parte inferior del *header* (Figura 4.2).
 - La parte superior (logo, iconos de redes sociales, idioma y términos y condiciones) no será modificable.



Figura 4.2: *Header*

- **HU2** Como administrador necesito añadir, modificar y eliminar elementos del *footer*.
 - Sólo se podrán modificar las partes que se señalan en la Figura 4.3.

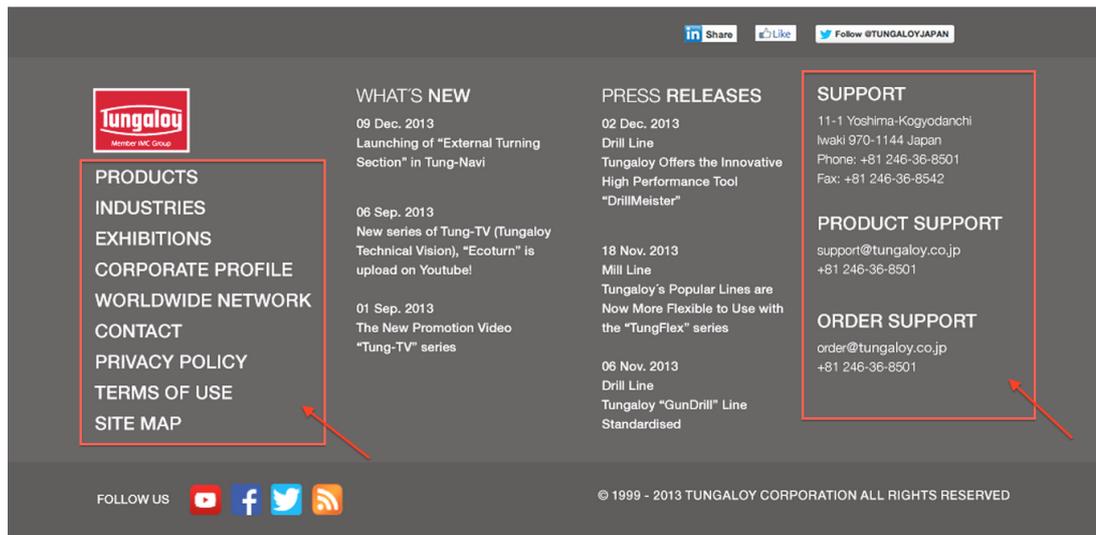


Figura 4.3: Partes modificables del *footer*

- **HU3** Como administrador necesito crear, modificar y eliminar páginas.
 - La modificación de páginas incluye toda la funcionalidad básica de edición de un documento: añadir texto, títulos, enlaces, imágenes, incrustar vídeos, cambiar la alineación de textos, etc.
- **HU4** Como administrador necesito crear pestañas.



Figura 4.4: Ejemplo de pestaña

- **HU5** Como administrador necesito crear acordeones.

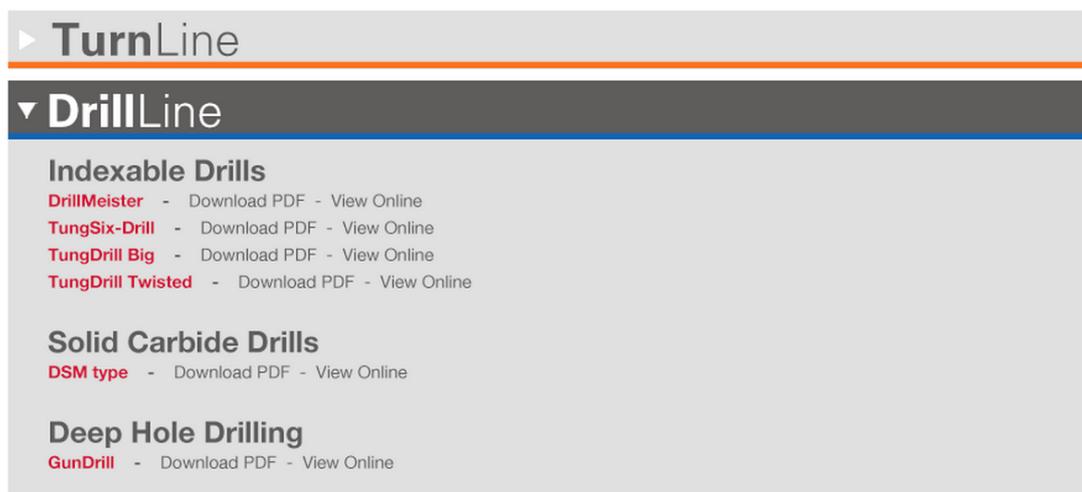


Figura 4.5: Ejemplo de acordeón

- **HU6** Como administrador necesito crear formularios.
- **HU7** Como administrador necesito crear menús desplegables.



Figura 4.6: Ejemplo de menú desplegable

- **HU8** Como administrador necesito crear, modificar y eliminar *sliders* y añadirlos a una página.
- **HU9** Como administrador necesito crear, modificar o eliminar eventos que se muestren automáticamente en la sección de eventos.
- **HU10** Como administrador necesito crear, modificar o añadir productos y que se muestren automáticamente en la sección de productos.
- **HU11** Como administrador necesito añadir, modificar o eliminar distribuidores y subsidiarias que se muestren automáticamente en el mapa de distribuidores.
- **HU12** Como administrador necesito añadir, modificar o eliminar publicaciones que se muestren automáticamente en la sección de publicaciones.
- **HU13** Como administrador necesito añadir, modificar o eliminar destinatarios del formulario de contacto.

Historias de usuarios de la parte del visitante web:

- **HU14** Como usuario necesito contactar con la empresa a través de un formulario que envíe un mensaje al destino seleccionado.
- **HU15** Como usuario necesito acceder a redes sociales de la empresa y compartir sus noticias en mis redes sociales.

4.2.3. Requisitos del entorno de explotación

Respecto al entorno de explotación el hosting ha de cumplir los siguientes requisitos a partir de los cuales desarrollamos el proyecto:

- PHP 5.2.4 o superior.
- MySQL 5.0 o superior.
- Módulo `mod_rewrite` de Apache. Es un módulo del servidor web que permite crear direcciones URL alternativas a las dinámicas generadas por la programación del sitio web, de esta manera podemos hacerlas más legibles y fáciles de recordar.
- Sistema de almacenamiento en caché como APC, Memcached o similar. Sistemas que permiten mejorar el rendimiento de sitios web.

4.2.4. Requisitos de calidad

El sitio web ha de cumplir unos requisitos básicos de calidad:

- Compatible con todos los navegadores de PC actuales: IE 7 o superior, Mozilla Firefox, Google Chrome, Safari, etc.
- Compatible con todos los navegadores móviles actuales: Safari, Google Chrome, Firefox, Opera, etc.
- Optimización para SEO.
- Facilidad de uso por personas no técnicas.
- Rápido, fiable y escalable.
- Seguro.

4.2.5. Restricciones

La principal restricción para el desarrollo del proyecto es temporal, las 300 horas de las que disponemos.

4.3. Diseño del sistema

Analizando el contenido de cada página del *mockup* observamos que el sistema está compuesto tanto de páginas estáticas como de páginas con contenido dinámico. En nuestro diseño, las páginas con contenido estático almacenan toda la información utilizando la funcionalidad que por defecto Wordpress nos ofrece. Para las páginas con contenido dinámico necesitamos crear estructuras de almacenamiento que hagan uso de lecturas en una base de datos para seleccionar el contenido a mostrar.

A continuación detallamos el diseño de cada página con contenido dinámico. Partimos del diseño para crear la estructura lógica, en el punto *5.2.2 Implementación en Wordpress* detallamos cómo usar los recursos del gestor de contenidos Wordpress para desarrollar un sistema con el diseño especificado.

Página: *Home*

En la Figura 4.7 podemos ver la parte del diseño de la *Home* que nos lleva al diseño lógico de la Figura 4.8. Observamos que se necesitan tres entidades independientes: *Slider*, *Noticia* y *Prensa*.

La entidad *Slider* está formada por un nombre como identificador, un conjunto de imágenes y el contenido o texto que se visualiza sobre cada una de ellas.

Respecto a las entidades Noticia y Prensa, aunque a simple vista las dos entidades son iguales las hemos separado porque destinamos cada entidad a una finalidad. En el diseño no se aprecia debido a que los textos que aparecen son de relleno pero la entidad Noticia se destina a la publicación de cualquier tipo de novedad y la entidad Prensa a la publicación de noticias periódicas de carácter más periodístico.



Figura 4.7: Captura de parte de la *Home*

Slider	Novedad	Prensa
- nombre - imágenes - texto	- título - fecha - contenido	- título - fecha - contenido

Figura 4.8: Captura de parte de la *Home*

Página: *Products*

En la Figura 4.9 vemos parte del diseño de la página *Products*, de ella extraemos la estructura lógica detallada en la Figura 4.10. Si observamos la página completa disponible en el *Anexo B Diseño proporcionado por el cliente* la necesidad de cada entidad se hace más evidente.

Sección → **CUTTING**
“Tools”

The application of cutting tools is varied and can generally be categorized into turning, milling and drilling. Tungaloy has an extensive line-up of TAC inserts with a combination of various grades, dimensions and chipbreakers to suit the diverse demands of the marketplace.



Categoría → **MillLine** Back to Top

Tipo → **Indexable Mills**

Subtipo → High Feed Milling
DoFeed series

Producto → DoFeedQuad
EXP and TXP

Face Milling
Dopent
DoOcto DoQuad



Figura 4.9: Captura de parte de la página *Products*

Como podemos apreciar en la Figura 4.10, se trata de una jerarquía vertical en la que cada entidad tiene un único padre y múltiples hijos. Cada elemento contiene determinados atributos necesarios para la correcta visualización de la página. Por ejemplo, la entidad *Categoría* almacena el color de la barra que se muestra a su derecha. Por otro lado, la entidad *Producto* almacena datos que se mostrarán al entrar en el enlace de cada producto.

El resto de elementos del diseño de la Figura 4.9 no necesitan estructuras de datos específicas, los almacenamos como contenido estático de la página ya que no corresponden a ninguna entidad que podamos abstraer conceptualmente.



Figura 4.10: Diseño lógico de la página *Products*

Página: *Worldwide Network*

En la Figura 4.11 observamos la parte del diseño que nos conduce al diseño lógico. En la Figura 4.12 vemos que cada subsidiaria tiene varios distribuidores y cada distribuidor pertenece a una subsidiaria. Cada entidad almacena los datos que vemos en el diseño que se desean mostrar.



Figura 4.11: Captura de parte de la página *Worldwide Network*



Figura 4.12: Diseño lógico de la página *Worldwide Network*

Página: *Publications*

El diseño lógico de esta página es muy parecido al de la página *Products*. En la Figura 4.13 tenemos la parte relevante del *mockup* para el análisis. En la Figura 4.14 el diseño lógico. Apreciamos que también se trata de una jerarquía vertical con múltiples hijos para un sólo padre. La entidad *Acordeón* almacena el color de la línea que se muestra bajo cada desplegable, y la entidad *Publicación* el nombre y el PDF que se muestra al pinchar sobre los enlaces.

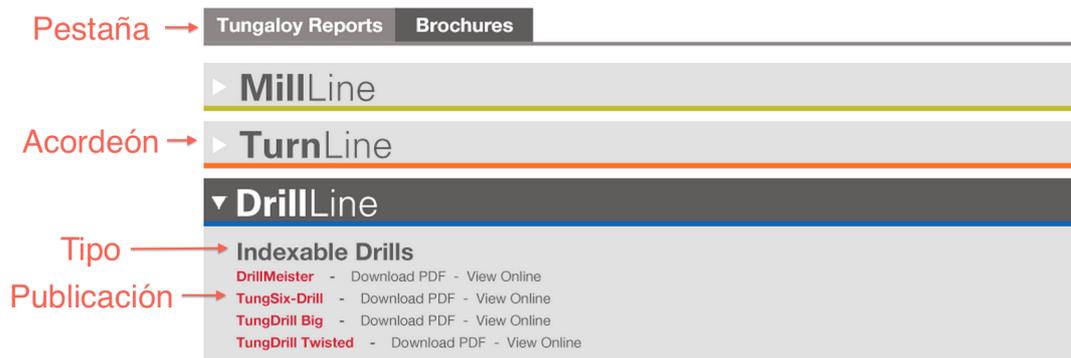


Figura 4.13: Captura de parte de la página *Publications*

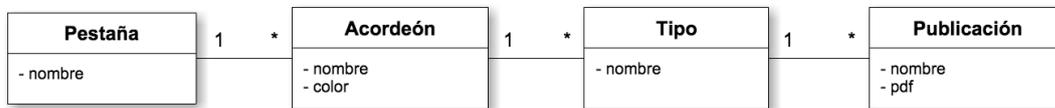


Figura 4.14: Diseño lógico de la página *Publications*

Página: *About Us*

En la página *About Us* encontramos una sección donde se muestran los últimos eventos programados (Figura 4.15), por cuestiones de comodidad deseamos que estos eventos se muestren automáticamente en lugar de tener que ir modificando la página manualmente cada vez que un evento ya ha pasado. Así que creamos una estructura que almacene los eventos y así cuando se muestra la página se mostrarán sólo los próximos eventos. En la Figura 4.16 vemos el diseño lógico necesario, es muy simple, una única entidad que contiene toda la información del evento y la fecha con la que se comprobará si el evento ya ha pasado o no.

EVENTS

“Exhibitions&Seminars”

Evento →

21, 24 May 2014
Metaltech - Malaysia
(Kuala Lumpur)
19, 23 June 2014
ACMEE- India (Chennai)
8, 13 September 2014
IMTS- USA (Chicago)
16, 20 September 2014
AMB- Germany (Stuttgart)
8, 13 September 2014
IMTS- USA (Chicago)
16, 20 September 2014
AMB- Germany (Stuttgart)
19, 23 June 2014
ACMEE- India (Chennai)
21, 24 May 2014
Metaltech - Malaysia
(Kuala Lumpur)



Figura 4.15: Captura de parte de la página *About Us*

Evento
- nombre - fechas - lugar

Figura 4.16: Diseño lógico de la página *About Us*

Capítulo 5

Implementación, pruebas y documentación

5.1. Front-end

La fase de desarrollo *front-end* consiste en la traducción del diseño proporcionado por el cliente, hecho a base de imágenes, a código HTML5, CSS3 y JavaScript. Para ello, utilizamos los componentes que nos ofrece Bootstrap modificando su aspecto mediante LESS y CSS para adaptarlo a nuestras necesidades.

Empezamos con una implementación de todas las páginas sin cuidar demasiado el detalle, de esta manera, como si de los cimientos de una construcción se tratase, el cliente puede ir viendo el desarrollo del esqueleto del sitio y como vamos agregando detalles a este esqueleto inicial.

En esta fase también tenemos especial cuidado con el código escrito para optimizarlo para posicionamiento SEO. Con esta finalidad, tenemos en cuenta los siguientes puntos:

- Escribir HTML válido, es decir código que cumpla con los estándares del *World Wide Web Consortium*.
- Utilizar las nuevas etiquetas de HTML5.
- Utilizar correctamente etiquetas: *title*, *meta description*, *h1*, *h2*, *h3*...
- No usar JavaScript dentro del archivo HTML.
- Usar CSS3 en lugar de JavaScript siempre que se pueda.
- Escribir el mínimo código HTML posible.
- Utilizar nombres semánticos y descriptivos para variables.

Finalmente revisamos todas las páginas para programar la funcionalidad *responsive*. En las Figuras 5.1 y 5.2 encontramos un ejemplo del aspecto de la *Home* para dos tamaños distintos de



Figura 5.1: *Home* para escritorio

pantalla. Se trata de revisar cada página y programar distinto aspecto dependiendo del tamaño de la pantalla.

5.2. Back-end

Para explicar el funcionamiento de la parte *back-end* es preciso ciertas nociones básicas de el CMS WordPress, en el siguiente punto explicamos algunos de los conceptos necesarios para comprender la implementación del sistema.

5.2.1. Introducción a Wordpress

WordPress es un sistema de gestión de contenidos desarrollado en PHP, bajo licencia GPL, para entornos que ejecuten MySql y Apache. Este sistema permite a usuarios finales la gestión de contenidos de un sitio web a través de una interfaz de administración sin necesidad de conocimientos técnicos. Existe un amplio conjunto de temas o plantillas HTML que ofrecen sitios web ya completos a falta de la introducción de contenidos. Por otro lado, WordPress ofrece a los desarrolladores la posibilidad de crear temas o plantillas propias y así desarrollar webs a medida que puedan ser gestionadas desde la interfaz de administración de WordPress. También existen una gran variedad de *plugins* que permiten extender la funcionalidad de WordPress añadiendo

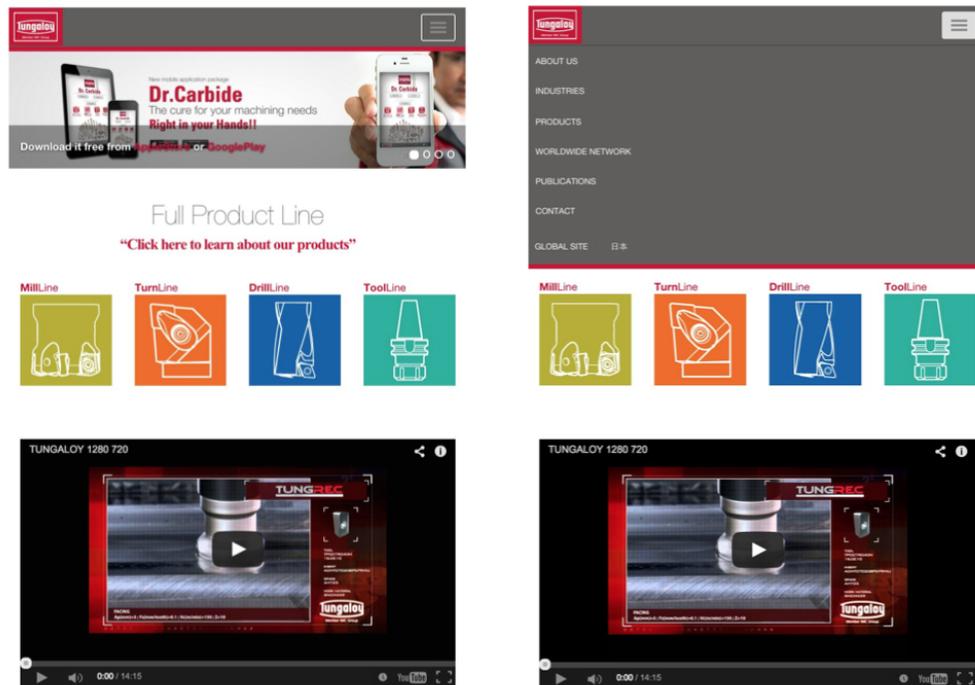


Figura 5.2: *Home* para móviles

características y funciones como: gestión de varios idiomas, servicio anti-spam, funciones para eCommerce, etc. Al igual que con las plantillas, también podemos desarrollar nuestros propios *plugins*.

WordPress nos ofrece varios mecanismos para modificar y gestionar nuestro sitio web, enumeremos y expliquemos los más relevantes o los que a nosotros nos han sido de mayor utilidad:

- *Posts types* y *custom post types*.
- Taxonomias.
- *Shortcodes*.
- Plantillas.
- *Widgets*.

Post types

Un *post type* es un tipo de datos en WordPress. Este tiene cinco tipos de *posts* por defecto[46]:

1. Entradas: normalmente son utilizadas como elemento de contenido básico en los blogs y suelen mostrarse en orden cronológico inverso.

2. Páginas: otro elemento de contenido pero que en lugar de mostrarse junto a los demás de su tipo en determinado orden se muestran individualmente como elemento único, suelen utilizarse para mostrar información estática del sitio. Por ejemplo, podríamos dedicar una página para mostrar todas las entradas que existan o para mostrar información estática como información de contacto. Cada página puede utilizar distintas plantillas para mostrar la información de manera diferentes.
3. Adjuntos: este *post type* almacena los archivos subidos a través de WordPress.
4. Revisiones: pertenecen a un *post type* padre, cada instancia de revisión es un borrador aún no publicado de otro tipo de *post*.
5. Menus de navegación: mantienen información de otros elementos de WordPress.

Además de estos *posts types* se pueden crear nuevos tipos de posts. Los *custom post types* son nuevos tipos de posts personalizados que podemos crear con los campos y características que nosotros queramos. En el *Anexo A Ejemplos de código WordPress* podemos consultar cómo crearlos.

Taxonomías

Las taxonomías[50] permiten agrupar *cosas*, como por ejemplo: entradas, páginas, adjuntos, etc. Una taxonomía es como una etiqueta que le damos a varios elementos para identificarlos, organizarlos y poder hacer búsquedas.

WordPress tiene tres tipos de taxonomías predeterminadas: categorías, etiquetas y categorías de links. También ofrece la posibilidad de crear nuevas taxonomías personalizadas. Llamamos taxonomías cerradas a las taxonomías que no permiten la creación de nuevos elementos y taxonomías abiertas a las que permiten al usuario crear y eliminar elementos.

Shortcodes

Los shortcodes[49] son pequeños códigos que podemos utilizar desde el editor de WordPress. Permiten ahorrar tiempo en labores repetitivas y preprogramar funciones para añadir al contenido de entradas o páginas sin necesidad de que el usuario tenga conocimientos técnicos. Son como tags HTML que usan corchetes ([]) en vez de los símbolos de *mayor y menor qué* (< >). Tienen un aspecto así:

```
[shortcodeldeejemplo]
```

En el *Anexo B Ejemplos de código WordPress* describimos la creación de varios de los *shortcodes* del proyecto.

Plantillas

Wordpress permite asignar fácilmente a las páginas una plantilla[45] de página, de esta manera conseguimos que la página tenga determinada estructura o aspecto.

Widgets

Un *widget*[53] es un pequeñísimo programa que nos da acceso a funciones que usamos normalmente. Este mecanismo nos permite definir una zona de *widget* y colocar en esa zona cualquier tipo de *widget*.

5.2.2. Implementación en Wordpress

Una vez introducidos a los mecanismos que Wordpress nos ofrece para adaptar su panel de administración a nuestra web, vamos a repasar página por página y ver cómo utilizamos estos mecanismos para desarrollar nuestro sitio web. Antes de ello, la Figura 5.3 nos muestra un esquema general de los mecanismos que utilizamos para cada parte de la web.

Página: *Home*

En la Figura 5.4 podemos observar todas las herramientas de Wordpress que empleamos para el desarrollo de la *Home*.

En el *header* hacemos uso del *post type menu*, de esta manera permitimos que desde el panel de administración, en la sección de menus se puedan añadir o quitar elementos. La parte izquierda del *footer* también tiene un menú asociado, de igual manera, se podrá modificar desde la sección de menus del panel de administración. El *post type menu* de Wordpress nos permite añadir elementos, del tipo nombre con su respectivo enlace, es el mecanismo perfecto para hacer modificables estas dos zonas.

En la parte derecha del *footer* encontramos una zona modificable a través de un *widget*. Para hacerla modificable desde el panel señalamos el espacio como zona para *widgets*. Los *widgets* y los menús son los únicos mecanismos que nos permite modificar partes de la página que no son del contenido principal y no pertenecen a ningún *post type*.

Tanto el *header* como el *footer* son elementos que están presentes en todas las páginas de la web, para lograrlo los incluimos en todas las plantillas del tema.

Continuamos con la Figura 5.4, debajo del *header* encontramos un *slider*. Para poder guardar distintos *sliders* e intercambiarlos cuando queramos sin necesidad de crear una copia completa de la página con un *slider* diferente creamos un *custom post type slider*. Este está formado por los campos que se describen en el apartado 4.3. *Diseño del sistema*, un nombre identificativo, todas las imágenes del slider y el contenido que se muestra sobre cada imagen. De esta manera,

POST TYPES	SHORTCODES
<p>Post types por defecto:</p> <ul style="list-style-type: none"> • Entradas → Para noticias (<i>News</i>) • Páginas → Para cada página de la web • Adjuntos → Para imágenes y pdfs • Revisiones → Para borradores de cualquier <i>post type</i> • Menús → Para el <i>header</i> y el <i>footer</i> <p>Custom post types:</p> <ul style="list-style-type: none"> • <i>Press</i> → Para prensa en la <i>Home</i> y en la página de prensa • <i>Sliders</i> → Para <i>sliders</i> en la <i>Home</i> • <i>Events</i> → Eventos de la página <i>About us</i> • <i>Products</i> → Productos de la página <i>Products</i> • <i>Distributors</i> → Para distribuidores y subsidiarias de la página <i>Worldwide network</i> • <i>Publications</i> → Publicaciones en la página <i>Publications</i> 	<ul style="list-style-type: none"> • <i>What's new</i> → Para mostrar las últimas noticias, como en la <i>Home</i> • <i>Press release</i> → Para mostrar la última prensa, como en la <i>Home</i> • <i>Events</i> → Para mostrar los últimos eventos en la página <i>About us</i> • <i>Teaser</i> → Para los títulos con subtítulos en todas las páginas • <i>Dropdown</i> → Para desplegables como el de la página <i>About us</i> • <i>Tabgroup</i> y <i>tab</i> → Para pestañas como las de la página <i>Publications</i> • <i>Acordeongroup</i> y <i>acordeon</i> → Para acordeones como los de la página <i>Publications</i> • Shortcodes de edición general
TAXONOMIAS	WIDGETS
<ul style="list-style-type: none"> • Taxonomía cerrada: <i>Distributor</i> o <i>subsidiarie</i> → para el post type <i>Distributors</i> • Taxonomía abierta para <i>Products</i> • Taxonomía abierta para <i>Publications</i> 	<ul style="list-style-type: none"> • <i>Sidebar</i> → Para modificar el contenido de la barra lateral de la plantilla <i>Sidebar</i> • <i>Footer</i> → Para modificar contenido de contacto
PLUGINS	PLANTILLAS
<ul style="list-style-type: none"> • Contact Form 7 → Para formularios • Antispam Bee → Para evitar SPAM • WordPress SEO → Para mejorar el SEO • WPML → Gestión de idiomas • Regenerate Thumbnails → Para regenerar el tamaño de <i>thumbnails</i> • WP Smush.it → Para reducir tamaño de imágenes 	<ul style="list-style-type: none"> • <i>Homepage</i> • <i>Sidebar</i> → Para páginas con la barra lateral que aparece en la página <i>Products</i> • <i>Products</i> → La plantilla muestra automáticamente • <i>Worldwide Network</i> • <i>Publications</i>

Figura 5.3: Tabla resumen de implementación

configurando el panel para ello, cuando creamos una página podemos indicar si queremos que se le añada un *slider* y cuál de todos los que haya almacenado en el sistema.

Más abajo en el diseño encontramos dos cuadros en los que se muestra los titulares de noticias y prensa, crearemos un *custom post type* para la prensa y utilizaremos el *post type* por defecto *entradas* para las noticias o novedades. A su vez, estas noticias se encuentran dentro de un recuadro que muestra sólo las tres últimas registradas en el sistema con determinado diseño, para mostrar estos recuerdos creamos dos *shortcodes* que realizan una consulta a la base de datos y muestran la información con la apariencia deseada. De esta manera escribiendo en el editor: *[whatsnew]* o *[pressrelease]* obtenemos el recuadro de la figura mostrando las últimas noticias y la última prensa.

Recordemos el motivo de utilizar *post types, shortcodes, widgets, etc.* Podemos lograr toda la apariencia que se muestra en la Figura 5.4 sólo con código HTML y CSS, pero la página no sería editable sin modificar el código, no podría ser modificada sin conocimientos técnicos. Por este motivo adaptamos el código HTML que ya teníamos al panel de administración de Wordpress, para que cualquier usuario sea capaz de editar el contenido de la web.

Página: *About us*

En la Figura 5.5 vemos los mecanismos que utilizamos para adaptar la página *About us* al panel de Wordpress.

Arriba del todo vemos un título que se muestra en color negro, junto a un subtítulo en color rojo, con otra fuente y entre dos rayas. Para que el usuario pueda poner este tipo de títulos en cualquier página creamos el *shortcode teaser*, de esta manera el usuario podría lograr el resultado de la figura escribiendo en el editor:

```
[acordeongroup]
[teaser title='events' subtitle='Exhibitions & Seminars']
```

Destaquemos un pequeño detalle, este tipo de títulos se utilizan en todas las páginas de la web, en todas las páginas el subtítulo aparece siempre completamente rojo, a excepción de este caso en el que el símbolo *℘* aparece en color negro y con el mismo tipo de letra que el título principal. Implementar una solución editable para este pequeño detalle complicaría bastante la solución de modo que le recomendamos al cliente dejar de lado ese pequeño detalle de diseño y poner el subtítulo siempre todo en rojo, con el visto bueno del cliente llevamos a cabo la solución más sencilla. Esto es un ejemplo de la comunicación cliente-desarrollador que nos enseña como el desarrollador ha de aconsejar al cliente para poder llegar a la mejor solución.

Sigamos con la Figura 5.5, a la izquierda vemos una lista de eventos, para almacenar estos eventos creamos un *custom post type events* con la fecha, nombre y lugar. Para mostrarlos con el aspecto del diseño, desarrollamos un *shortcode events* que realiza una consulta a la base de datos de Wordpress y muestra los próximos eventos.

Más abajo en la figura encontramos un desplegable con un botón que muestra una serie de opciones, al seleccionar una opción y clicar sobre el botón el navegador debe redirigirnos a

determinada página. Para que el usuario final pueda crear este tipo de desplegados creamos un *shortcode dropdown*. De esta manera el cliente puede obtener un desplegado funcionando con el siguiente código:

```
[dropdown default='choose something' button='go']  
  [dropdownitem name='name1' url='url1']  
  [dropdownitem name='name2' url='url2']  
[/dropdown]
```

Página: *Industries*

La página *Industries* (Figura 5.6) se puede conseguir mediante el editor de páginas de WordPress y utilizando *shortcode teaser* del que hemos hablado en la sección anterior. Esta página no necesita de la creación de nuevos mecanismos de edición.

Página: *Products*

En la Figura 5.7 observamos la página *Products*.

A la derecha vemos una barra lateral que implementamos mediante un *widget*. Para lograr una página con esta barra lateral el usuario tendrá que seleccionar la plantilla *sidebar* y editar los elementos que desee desde la sección de *widgets* del panel WordPress.

Como podemos apreciar más abajo, la página está compuesta por una lista de productos, para almacenar y gestionar todos los productos creamos un *custom post type products*. También abrimos la posibilidad de crear taxonomías para este nuevo *post type*. Desde el panel de Wordpress se podrán crear categorías o taxonomías indexables. A cada producto se le tendrá que añadir una categoría, ésta determinará el lugar en el que se mostrará el producto. El nivel de la categoría determinará como se muestre ésta. Por ejemplo, las taxonomías de nivel dos se muestran como aparece en la Figura 5.7, con un tamaño más grande y con una barra de color a su lado; las taxonomías de nivel tres aparecen en un tamaño de fuente menor y sin la barra a su derecha; las taxonomías de nivel cuatro en un tamaño aún menor.

Página: *Worldwide Network*

Esta página la encontramos en la Figura 5.8. En ella tenemos un desplegado de países, una zona gris en la que se muestra la subsidiaría de un país y abajo una zona blanca en la que aparecen todos los distribuidores del país. Al cambiar entre los países del desplegado cambian los datos de estas dos zonas.

Para implementar esta página hemos creado un *custom post type distributors* que almacene todos los datos necesarios (en el punto 4.3. *Diseño del sistema* ya detallamos qué datos necesitaríamos guardar). También hemos creado dos taxonomías cerradas para este *cueto post type*: *distributor* y *subsidiarie* y una taxonomía abierta *countries* para que el usuario pueda

EVENTS **shortcode**

— “Exhibitions&Seminars” —

custom post type

21, 24 May 2014
Metaltech - Malaysia
(Kuala Lumpur)

19, 20 June 2014
ACMEE- India (Chennai)

7, 13 September 2014
MTS- USA (Chicago)

16, 20 September 2014
AMB- Germany (Stuttgart)

8, 13 September 2014
IMTS- USA (Chicago)

16, 20 September 2014
AMB- Germany (Stuttgart)

11, 23 June 2014
ACMEE- India (Chennai)

21, 24 May 2014
Metaltech - Malaysia
(Kuala Lumpur)



shortcode

CAREER

— “Opportunities” —

Would you like have a nice future in a global company and take part of our success?

Tungaloy always is hunting talented people to be part of our professional team, if you have the skills and the talent, we want know about you because we have the knowledge and together we can get a brilliant future.

shortcode

Choose your country and explore our career opportunities

Choose your country:




Figura 5.5: Implementación conceptual de la página *About us*



COMPLETE
 “Industrial Coverage”

Every customer has specific needs. We study them in depth and provide the best solution.

<p>AUTOMOTIVE</p>  <p>Tungaloy's engineering teams and factories are located near the major car plants to provide automotive clients with creative solutions that improve their production processes.</p>	<p>HEAVY INDUSTRIES & POWER GENERATION</p>  <p>The high versatility, quality, and robustness of Tungaloy tools allow the power generation industry to take giant strides ahead.</p>
--	---

Figura 5.6: Implementación conceptual de la página *Industries*

crear países que añadir a los distribuidores.

Recordemos que hablamos de taxonomías abiertas y cerradas, las taxonomías abiertas permiten al usuario crear nuevas categorías en cambio con taxonomías cerradas el usuario no puede crear más taxonomías, sólo tiene disponibles las ya creadas.

Cuando el usuario seleccione un país del desplegable en la zona roja se realizará una consulta a la base de datos de todos los *distributors* del país y se mostrará en la zona gris aquel que tenga asignado la taxonomía *subsidiarie* y en la zona blanca todos los que tengan la taxonomía *distributor*.

Página: *Publications*

En la Figura 5.9 vemos el diseño de la página *Publications*. Para esta página creamos las estructuras y algoritmos para que la información se muestre automáticamente como en la página *Products* y también preparamos los mecanismos para que el usuario pueda hacer uso de pestañas y acordeones en otras partes de la web.

Implementamos dos *shortcodes* que permiten crear pestañas y acordeones del aspecto del diseño desde el panel de edición de WordPress. De esta manera, utilizando la siguiente expresión de *shortcodes* podemos obtener unas pestañas como el del diseño:

```
[tabgroup]
  [tab title='Tungaloy Reports'] Tab 1 content goes here. [/tab]
  [tab title='Brochures'] Tab 2 content goes here. [/tab]
[/tabgroup]
```

Taxonomias

1 →

CUTTING “Tools”

The application of cutting tools is varied and can generally be categorized into turning, milling and drilling. Tungaloy has an extensive line-up of TAC inserts with a combination of various grades, dimensions and chipbreakers to suit the diverse demands of the marketplace.



2
3
4

MillLine

Indexable Mills

High Feed Milling
DoFeed series
DoFeedQuad
EXP and TXP

Face Milling

Dopent
DoOcto, DoQuad
Taw, EAWTPW, EPW

Shoulder Milling

Tung-AkuMill
Tung-Tri
RoughingMill series
TungQuad
TungRec
TecMill
DoRec
T/EFE12, DPD09 and EDPD09 types
T/EPS11, 17 types

Profile Milling (Round Insert)

FixRMill
RoundSplit

Slot Milling

SlotMill series
TecSlot

Multi Function Milling Tools

HybridTacMill
EVX

Head Changeable EndMill

TungMeister

Inserts Grades for Milling

BX910
T1115
AH130
T3130 series
DS1000 series
NS740

Custom post type

TurnLine

Turning Tools

General Turning

DominiTurn
EcoTurn
DimpleFx
PSF & Pss
HMM & AH905
Y-Pro
DominiTurn
EcoTurn
TurningA
StreamJet Bar
AL chipbreaker, T-DIA

Heavy Turning

TurnTec



Tungaloy's Turning tools primarily provide our customers with the high productivity and high efficiency that can be expected from an industry.

“Can I help you?”

e-Catalog

Search for Tools

Tung-Navi

Tungaloy Tool Navigator



Insert Converter



Digital Publications



Tungaloy Technical Vision

“contact US”

Widget

Figura 5.7: Implementación conceptual de la página *Products*



WORLDWIDE
"Network"

Custom post type

Choose your country or state :
State (only for US)

Tungaloy Americ Inc.
View Site
3726 N Ventura Drive
Arlington Heights
IL 60004, U.S.A
Phone: +1-888-554-8394
Fax: +1-888-554-8392

Distributors on Florida:

Alternatives for Industry Inc
3726 N Ventura Drive
Arlington Heights
IL 60004, U.S.A.
Phone: +1-888-554-8394
Fax: +1-888-554-8392

Taxonomias
2
Miller Bearings
3726 N Ventura Drive
Arlington Heights
IL 60004, U.S.A.
Phone: +1-888-554-8394
Fax: +1-888-554-8392

Dewil Tool Co Incorporated
3726 N Ventura Drive
Arlington Heights
IL 60004, U.S.A.
Phone: +1-888-554-8394
Fax: +1-888-554-8392

Figura 5.8: Implementación conceptual de la página *Worldwide Network*



Figura 5.9: Implementación conceptual de la página *Publications*

Para añadir el acordeón dentro de la primera pestaña como se muestra en la Figura 5.9 tendríamos que incluir los *shortcodes* necesarios donde pone *Tab 1 content goes here*. Utilizando la siguiente expresión de *shortcodes* podemos obtener un acordeón como las del diseño:

```
[acordeongroup]
  [acordeon title='MillLine'] Acordeon 1 content goes here. [/acordeon]
  [acordeon title='TurnLine'] Acordeon 2 content goes here. [/acordeon]
  [acordeon title='DrillLine'] Acordeon 3 content goes here. [/acordeon]
  [acordeon title='ToolLine'] Acordeon 4 content goes here. [/acordeon]
[/acordeongroup]
```

Ya hemos visto los dos *shortcodes* que hemos creado para usos futuros del usuario en otras páginas. Veamos cómo hemos implementado esta página.

Como podemos observar en la Figura 5.9 esta página está destinada a mostrar publicaciones. Para almacenarlas y gestionarlas creamos un *custom post type publications* con taxonomías abiertas. La implementación es muy parecida a la de la página *Products*. Las taxonomías que se creen serán indexables desde el panel de administración, para mostrarlas, las de primer nivel se mostrarán como pestañas, las de segundo nivel como acordeones y las de tercer nivel como títulos dentro del acordeón.



Figura 5.10: Implementación conceptual de la página *Contact*

Página: *Contact*

La última página aparece en la Figura 5.10. Se trata de un formulario de contacto que debe enviar el mensaje a distintos destinatarios dependiendo de la destinación y del país que se escoja. Para la implementación de este formulario utilizamos el *plugin Contact Form 7*[29]. Este plugin nos permite administrar múltiples formularios de contacto personalizando su aspecto y contenido, permite modificar los mensajes dándoles determinada estructura, programar respuestas automáticas, captchas, filtrado de spam, etc.

Todas las páginas

Por último mencionamos algunos recursos que hemos utilizado para el sitio web completo, sin ser específicos de ninguna página.

Utilizamos los *post type* por defecto *Adjuntos* para almacenar imágenes y PDFs. Y el *post type Revisiones* y para mantener borradores de cualquier *post type* (páginas, entradas, publicaciones, etc.).

También creamos algunos *shortcodes* de edición general para la creación de columnas dentro de una página y la incrustación de vídeos.

Por último, utilizamos *plugins* con funcionalidad varias:

- Antispam Bee[27]: un *plugin* antispam que detecta y elimina los comentarios spam.
- WordPress SEO[16]: permite mejorar el posicionamiento SEO mediante la introducción de palabras clave, etiquetas en imágenes, etc.
- WPML[23]: un *plugin* para la traducción de sitios web WordPress, nos facilita la creación de páginas hermanas en diferentes idiomas.
- Regenerate Thumbnails[2]: permite regenerar las miniaturas de todas las imágenes almacenadas, muy útil cuando por ejemplo modificamos el tamaño por defecto para las miniaturas o cambiamos de tema.
- WP Smush.it[1]: reduce el tamaño de las imágenes y mejora el rendimiento.

5.3. Pruebas

Una vez desarrollado el sistema pasamos a la realización de pruebas. Para garantizar la calidad del sistema desarrollado y la coherencia con respecto a los requisitos que se establecieron que éste debería cumplir, se han llevado a cabo varios tipos de pruebas:

- Pruebas de GUI y navegación.
- Pruebas de funcionalidad o tests de aceptación.
- Pruebas de compatibilidad.

Existen otro tipo de pruebas que podríamos pensar necesarias como pruebas de seguridad, de estrés, de desempeño, de volumen, etc. sin embargo éstas serían pruebas más para verificar el funcionamiento de WordPress que para verificar nuestro sitio web. Este tipo de pruebas ya han sido realizadas por el equipo de desarrollo de WordPress.

5.3.1. Pruebas de GUI y navegación

Para verificar la interacción del usuario con el sistema realizamos pruebas de GUI y navegación. El objetivo es asegurar que la interfaz en la parte *front-end* cumple los requisitos del diseño y que la navegación entre páginas funciona correctamente. También verificamos que toda la web sea *responsive* y se adapte a cualquier tamaño. Estas pruebas consisten en verificar página a página todo el sistema. Las repetimos distintas personas para que no se nos escape ningún detalle.

5.3.2. Pruebas de funcionalidad o tests de aceptación

Para comprobar y testear la funcionalidad del sistema realizamos tests de aceptación. Estas pruebas corresponden a las historias de usuario explicadas en el punto 4.2.2. *Historias de*

usuario. Al añadir nueva funcionalidad al sistema repasamos con el usuario estos tests de aceptación para comprobar que todo funciona correctamente y que el cliente está conforme con el funcionamiento del sistema.

A continuación detallamos los tests de aceptación de algunas de las historias de usuario. En el *Anexo E Pruebas de aceptación* podemos consultar los tests de todas las historias de usuario.

Id	Historia de usuario
HU1	Como administrador necesito añadir y eliminar enlaces del <i>header</i> .
HU2	Como administrador necesito añadir, modificar y eliminar elementos del <i>footer</i> .
HU3	Como administrador necesito crear, modificar y eliminar páginas.
HU4	Como administrador necesito crear pestañas.
HU5	Como administrador necesito crear acordeones.

Figura 5.11: Algunas historias de usuario

PRUEBAS DE ACEPTACIÓN HU1

1. **Añadir enlace:** al añadir un enlace, cuando no existe un enlace igual el sistema, el sistema añade el enlace y podemos verlo en el menú.
2. **Añadir enlace repetido:** al añadir un enlace, cuando ya existe un enlace igual en el sistema, el sistema añade el enlace y podemos verlo en el menú.
3. **Añadir muchos enlaces:** al añadir un enlace, cuando ya hay muchos enlaces y no caben más en la pantalla de la web, el sistema añade el enlace y lo muestra en una segunda línea del menú.
4. **Eliminar enlace:** al eliminar un enlace, el sistema elimina el enlace y ya no podemos verlo en el menú.
5. **Eliminar todos los enlaces:** al eliminar un enlace, cuando es el único enlace en el sistema, el sistema elimina el enlace y se muestra el menú vacío.

PRUEBAS DE ACEPTACIÓN HU2

1. **Añadir enlace:** al añadir un enlace al menú del *footer*, cuando no existe un enlace igual el sistema, el sistema añade el enlace y podemos verlo en el menú.
2. **Añadir enlace repetido:** al añadir un enlace al menú del *footer*, cuando ya existe un enlace igual en el sistema, el sistema añade el enlace y podemos verlo en el menú.
3. **Eliminar enlace:** al eliminar un enlace del menú del *footer*, el sistema elimina el enlace y ya no podemos verlo en el menú.
4. **Eliminar todos los enlaces:** al eliminar un enlace del menú del *footer*, cuando es el único enlace en el sistema, el sistema elimina el enlace y se muestra el menú vacío.

5. **Añadir elemento:** al añadir un elemento en la parte derecha del *footer*, cuando no existe un elemento igual en el sistema, el sistema añade el elemento y podemos verlo en el *footer*.
6. **Añadir elemento repetido:** al añadir un elemento en la parte derecha del *footer*, cuando ya existe un elemento igual en el sistema, el sistema añade el elemento y podemos verlo en el *footer*.
7. **Eliminar elemento:** al eliminar un elemento de la parte derecha del *footer*, el sistema elimina el elemento y ya no podemos verlo en el *footer*.
8. **Eliminar todos los elementos:** al eliminar un elemento de la parte derecha del *footer*, cuando es el único elemento en el sistema, el sistema elimina el elemento y la zona se muestra vacía.

PRUEBAS DE ACEPTACIÓN HU3

La funcionalidad de esta historia de usuario la ofrece cualquier instalación de WordPress por tanto no necesitamos pruebas aceptación para comprobar su funcionamiento.

PRUEBAS DE ACEPTACIÓN HU4

1. **Añadir pestaña:** al añadir una pestaña a una página, cuando todavía no existe ninguna pestaña en el sistema, el sistema muestra una única pestaña en la que el usuario no puede hacer clic.
2. **Añadir dos pestañas:** al añadir una pestaña, cuando ya existe otra pestaña en el sistema, el sistema muestra dos pestañas en las que el usuario puede hacer clic.
3. **Añadir muchas pestañas:** al añadir una pestaña, cuando ya existen muchas pestañas en el sistema y no caben en la pantalla de la web, el sistema muestra la pestaña en una segunda línea de pestañas.
4. **Hacer clic sobre pestaña:** al hacer clic sobre una pestaña, cuando se está mostrando el contenido de otra pestaña, el sistema cambia el contenido mostrado.
5. **Crear más de un conjunto de pestañas:** al crear un conjunto de pestañas, cuando ya existe otro conjunto de pestañas en el sistema, las pestañas continúan cambiando su contenido al hacer clic y se comportan independientemente.

PRUEBAS DE ACEPTACIÓN HU5

1. **Añadir acordeón:** al añadir un acordeón a una página, cuando todavía no existe ningún acordeón en el sistema, el sistema muestra un único acordeón que se despliega y encoge cuando el usuario hace clic sobre él.
2. **Añadir dos acordeones:** al añadir un acordeón, cuando ya existe otro acordeón en el sistema, el sistema muestra dos acordeones que se despliegan y encogen cuando hacemos clic sobre uno de ellos.

3. **Hacer clic sobre acordeón:** al hacer clic sobre un acordeón, cuando otro acordeón está abierto, el sistema despliega el acordeón y encoge el resto de acordeones.
4. **Crear más de un conjunto de acordeones:** al crear un conjunto de acordeones, cuando ya existe otro conjunto de acordeones en el sistema, los acordeones se continúan encogiéndose y desplegando correctamente y se comportan independientemente.

5.3.3. Pruebas de compatibilidad

Para verificar la compatibilidad con todos los navegadores web realizamos pruebas de compatibilidad, tanto de PC como móviles, probamos el sistema con distintos navegadores para detectar posibles pequeñas diferencias de visualización o comportamiento. Realizamos pruebas con Chrome, Firefox e Internet Explorer para Windows y para Mac. También realizamos pruebas desde navegadores de dispositivos móviles.

5.4. Documentación

Respecto a la documentación, durante el proyecto desarrollamos un documento junto al cliente documentando los avances, las propuestas de modificación y la evolución del proyecto.

La estructura de ficheros y el orden de los mismos también pueden considerarse documentación, pues estos son autoexplicativos, como veremos en el *Anexo B. Estructura de ficheros*, la empresa siempre utiliza la misma estructura cuidada que facilita la localización de las partes del código que se desea buscar.

Por último, esta memoria queda a disposición de la empresa a modo de documentación del proyecto.

Capítulo 6

Resultados y conclusiones

En este capítulo presentamos los resultados y conclusiones finales del proyecto: qué hemos conseguido, qué ha ido como esperábamos, qué no ha ido como esperábamos, qué hemos hecho mal, qué podríamos haber hecho mejor, etc.

6.1. Resultado temporal frente a planificación

La planificación temporal frente al resultado final no ha estado del todo ajustada. Planificamos en base a una estimación del esfuerzo necesario para cada tarea con horas de trabajo como métrica. Después del desarrollo del proyecto hemos observado que estimamos más tiempo del que realmente luego necesitamos para la mayoría de las tareas. Como probable causa de este desajuste señalamos el desconocimiento inicial de las tecnologías y la inexperiencia.

Destaquemos que la experiencia, o el conocimiento del que estima, es la característica más importante que influirá en una estimación de esfuerzo. Para realizar una estimación ajustada del esfuerzo de realización de un proyecto software se debería disponer de un histórico de datos de proyectos anteriores que proporcione una base de partida para la estimación. Sin embargo las estimaciones de este proyecto las realizamos sin experiencia y sin datos históricos de otros proyectos. Para disminuir el error en las estimaciones podríamos haber contrastado las estimaciones con otras realizadas por un experto de la empresa con verdadera experiencia y datos sobre anteriores proyectos. Aún así, la estimación se hacía difícil debido a que el histórico de la empresa corresponde a proyectos realizados por un equipo de profesionales especializados en las tecnologías del proyecto. El esfuerzo necesario para que un equipo de expertos desarrollase el proyecto nunca podría ser el mismo al esfuerzo necesario si el proyecto es desarrollado por un único miembro sin experiencia ni conocimientos en las tecnologías concretas a utilizar.

Otro de los posibles errores cometidos durante la planificación y estimación de tiempos podríamos encontrarlo en el afán por ajustar el tiempo total de las tareas a las trescientas horas de trabajo de las que disponíamos. Intentamos repartir el trabajo a realizar entre las trescientas horas sin considerar la posibilidad de que sobrase o faltase tiempo.

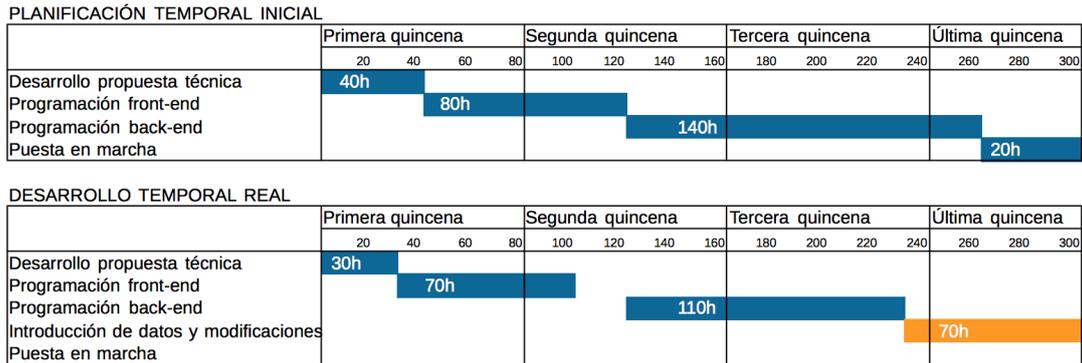


Figura 6.1: Planificación temporal inicial vs. Desarrollo temporal real

Por otro lado, no incluimos en la planificación una fase necesaria antes de la puesta en marcha: la introducción de datos en el sistema. No es una tarea que forme parte del desarrollo de un sistema web, pues ésta se realiza cuando el sistema ya está acabado, pero sí que es una tarea que se ha de realizar antes de la puesta en marcha. Se pactó con el cliente que la tarea estaría incluida dentro del proyecto y por tanto, una vez terminado el sistema y antes de ponerlo en marcha, había que introducir en el sistema todos los datos que nos proporcionase el cliente. El tiempo necesario para llevar a cabo esta tarea tendría que haber sido incluido en la planificación. Aún así, cabe destacar que esta es una tarea de muy difícil estimación ya que existe una enorme dependencia con el cliente: llegados a esta fase, hasta que el cliente no proporcione toda la información a introducir el proyecto queda estancado. Por este mismo motivo no conseguimos llegar a la puesta en marcha del sistema desarrollado.

En la Figura 6.1 podemos observar el diagrama de Gantt de la estimación inicial frente al diagrama que muestra el desarrollo real del proyecto. Como podemos observar, y ya hemos comentado anteriormente, todas las tareas de desarrollo técnico terminaron más rápido de lo previsto. Durante la segunda quincena apreciamos un periodo de tiempo durante el que se detuvo el desarrollo del proyecto, esto se debe a que como el proyecto había avanzado más de lo previsto se dedicaron varios días al desarrollo de otro pequeño proyecto. En el diagrama de desarrollo real hemos incluido en naranja la fase de introducción de datos y modificaciones, vemos como ésta acaba consumiendo todo el tiempo restante del que disponíamos y nos impide acabar con la puesta en marcha del sistema. Durante esta fase también hubo días en los que el proyecto estuvo parado por falta de nuevas indicaciones del cliente, estos días se dedicaron al desarrollo de pequeños proyectos. Dando prioridad al proyecto principal se iban introduciendo los datos y realizando modificaciones conforme el cliente las proporcionaba.

Por último, añadir que las trescientas horas de proyecto las terminamos el 12 de agosto en lugar del 4 de agosto como habíamos estimado. Este dato no tiene mayor relevancia ya que para la estimación supusimos jornadas de 8 horas ya sabiendo que podrían ser variables según las necesidades de la empresa de acogida. Hubo días con jornadas más cortas y como consecuencia acabamos las trescientas horas unos días más tarde de lo estimado inicialmente.

6.2. Conclusiones

El objetivo principal de este proyecto era el de crear un sitio web autogestionable, adaptable a todos los dispositivos, con gestión de idiomas sencilla y siguiendo el diseño proporcionado por el cliente. Una vez terminado el desarrollo del sistema podemos afirmar que hemos conseguido el objetivo. Repasando los requisitos y objetivos concretos que establecimos hemos logrado la consecución de todos ellos.

- Todo el contenido puede ser modificado desde un panel de control sin necesidad de conocimientos de programación o HTML, hemos llegado a este objetivo haciendo uso del panel de administración de WordPress desde el cual podemos modificar la mayoría de los elementos de la web.
- La apariencia de la web se adapta al dispositivo utilizado para visualizarla, hemos logrado este comportamiento utilizando HTML, CSS y etiquetas de Bootstrap específicas para este fin.
- El sitio web permite una gestión de idiomas sencilla, cada página tiene páginas hermanas de los diferentes idiomas. Para conseguir esta funcionalidad multiidioma hemos hecho uso de un *plugin* de WordPress.
- La web que hemos obtenido tras el desarrollo tiene la apariencia exacta del diseño que proporcionó el cliente. Mediante HTML, CSS y Bootstrap hemos replicado el diseño.

Hemos conseguido crear un sitio web al gusto del cliente y añadirle el panel de administración de WordPress para hacer posible su modificación. La utilización de WordPress también nos ha permitido beneficiarnos de la seguridad, fiabilidad y escalabilidad que este gestor de contenidos nos proporciona.

El sistema ha quedado completamente construido pero no hemos llegado a la fase de puesta en marcha. Es cierto que la puesta en marcha no forma parte del desarrollo del sistema y que por tanto podemos considerar el éxito del proyecto, sin embargo es una fase que incluimos en la planificación temporal inicial y que no se ha podido completar, por tanto debemos extraer conclusiones para poder planificar proyectos en el futuro de manera más ajustada.

Después de la realización del proyecto llegamos a la conclusión de que cuando es necesaria una fase de introducción de datos antes de la puesta en marcha, estas dos fases no deberían incluirse en la misma planificación temporal que el desarrollo del proyecto. Sería más adecuado una planificación para estimar la finalización del desarrollo y dejar abiertas las fechas para la puesta en marcha. Estimar el tiempo de introducción de datos es imposible, depende completamente del cliente, por tanto no es útil planificar tiempo ni fechas para esta fase.

Como trabajo futuro para la empresa queda la introducción de los datos restantes del cliente y la puesta en marcha del sistema web. Llegados a ese punto la empresa cliente podrá empezar a beneficiarse de la mejora de imagen y presencia en Internet.

Finalmente, añadir que este proyecto de naturaleza profesional en el ámbito de la Ingeniería del Software me ha ayudado a sintetizar e integrar las competencias y enseñanzas aprendidas durante el grado.

Glosario

A

Apache Es un servidor web HTTP, pág. 38.

APC Es una librería que implementa una caché alternativa para PHP. [32], pág. 30.

Aptana Studio Es un entorno de desarrollo integrado para el desarrollo web. Es de software libre y está basado en el IDE Eclipse. [5], pág. 23.

B

Bitbucket Es un servicio de alojamiento basado en la web para proyectos que utilicen el sistema de control de versiones Mercurial o Git. [4], pág. 23.

Bootstrap Es un *framework* HTML, CSS y JavaScript para el desarrollo de la parte *front-end* de sitios y aplicaciones web. [6], pág. 21.

C

captcha Es una prueba automática para diferenciar computadoras de humanos, pág. 51.

CMS *Content Management System*. Es un sistema de gestión de contenidos, es decir programa informático que permite crear una estructura de soporte para la creación y administración de contenidos, principalmente en páginas web, pág. 19.

CSS3 Es el último estándar de CSS, también conocida como hoja de estilo en cascada, es un lenguaje usado para definir la presentación de un documento estructurado en HTML o XML. [38], pág. 21.

D

DOM Modelo en Objetos para la Representación de Documentos, es una interfaz de programación que proporciona un conjunto estándar de objetos para representar documentos HTML y XML. A través del DOM los programas pueden acceder y modificar el contenido, estructura y estilo de los documentos HTML y XML. [39], pág. 23.

F

framework Es una estructura conceptual y tecnológica de soporte definido. Puede incluir soporte de programas o bibliotecas entre otras herramientas, su finalidad es ayudar a desarrollar y unir los diferentes componentes de un proyecto, pág. 21.

G

Git Es un sistema de control de versiones distribuido que permite mantener una gran cantidad de código caracterizándose por su eficiencia, realiza el control de versiones en repositorios locales. [12], pág. 23.

Google Analytics Es un servicio gratuito de estadísticas de sitios web del buscador de Google, pág. 20.

Google Webmaster Es un servicio de Google que permite a los creadores de páginas web comprobar el estado de la indexación de sus sitios en internet y optimizar su visibilidad, pág. 20.

H

hosting *Web hosting* (en español *alojamiento web*) es el servicio que provee a los usuarios de Internet un sistema para poder almacenar información, imágenes, vídeo, o cualquier contenido accesible vía web, pág. 20.

HTML5 Es la quinta revisión del conocido lenguaje de etiquetas HTML (*HyperText Markup Language*), el lenguaje básico de la *World Wide Web*. [40], pág. 21.

J

JavaScript Es un lenguaje interpretado integrado en las páginas webs y ejecutado por los navegadores modernos. [14], pág. 23.

JQuery Es una biblioteca JavaScript que permite simplificar la manera de interactuar con los documentos HTML, nos facilita la manipulación del árbol DOM, manejar eventos, etc. citebib:jquery, pág. 23.

K

know-how (del inglés *saber-cómo*) o Conocimiento Fundamental es una forma de transferencia de tecnología. Es una expresión anglosajona utilizada en los últimos tiempos en el comercio internacional para denominar los conocimientos preexistentes no siempre académicos, que incluyen: técnicas, información secreta, teorías e incluso datos privados (como clientes o proveedores), pág. 19.

L

LESS Es un lenguaje pre-procesador de CSS que lo extiende, añadiéndole características como la creación de variables, funciones y muchas otras técnicas que le permiten hacer CSS más fácil de mantener y extender. [22], pág. 21.

licencia GPL La Licencia Pública General garantiza a los usuarios finales la libertad de usar, estudiar, compartir y modificar software, pág. 38.

M

MailChimp Herramienta para WordPress que permite incorporar al sitio web un formulario de suscripción y la gestión de suscripciones [34], pág. 19.

Memcached Es un sistema distribuido para caché que se instala en un servidor y funciona como un servicio más de la máquina. [11], pág. 30.

mockup Es un modelo o diseño que se utiliza para mostrar el aspecto que tendrá un sistema, pág. 15.

MySql Es un sistema de gestión de base de datos relacionales, multihilos, y multiusuario. Es el sistema de gestión de base de datos que utiliza Wordpress. [24], pág. 23.

N

newsletter Publicación digital más bien informativa que se distribuye a través del correo electrónico con cierta periodicidad, pág. 19.

P

PHP Es un lenguaje de programación del lado del servidor, es el lenguaje utilizado para programar Wordpress. [33], pág. 23.

Piwik Es una aplicación gratuita y de código abierto que genera estadísticas de sitios web, pág. 20.

plugin También conocido como complemento, es una aplicación que se relaciona con otra para aportarle una función nueva y generalmente muy específica, pág. 39.

Portable Object files Son archivos de texto plano que contienen traducciones. Cada idioma ha de tener su propio archivo *PO* en el que se traduzca cada expresión al idioma del archivo. [51], pág. 72.

post type Es un tipo de datos de Wordpress. [46], pág. 39.

Prepros Es una herramienta para el desarrollo web, entre sus funciones encontramos la compilación de código de LESS a CSS y la concatenación de ficheros JavaScript en tiempo real. [28], pág. 23.

R

Redirección 301 Es un código de respuesta HTTP para indicar una redirección. Indica que la petición actual y todas las futuras deberán ser dirigidas a una dirección dada. Un uso correcto de la redirección 301 influye favorablemente en el posicionamiento de la web. [3], pág. 20.

responsive El término hace referencia a un diseño web adaptativo o adaptable, cuyo objetivo es adaptar la apariencia de las páginas web al dispositivo que se esté utilizando para visualizarla, pág. 15.

S

Search Engine Indexing Es el proceso de recolección, evaluación y almacenamiento de datos por parte de un buscador, el resultado de este procesado de datos le da a cada sitio web una posición en la que aparecerá según la búsqueda que se realice, pág. 20.

SEO *Search Engine Optimization*. Se refiere al posicionamiento en buscadores y optimización en motores de búsqueda, pág. 16.

shortcode Es un pequeño código que podemos utilizar desde el editor de WordPress para preprogramar funciones. [49], pág. 40.

slug Hace referencia a un título en el que se han sustituido los espacios en blanco por guiones y se han eliminado todos los caracteres que no sean letras o números. En Wordpress no puede haber dos slugs iguales y se utilizan a modo de referencias, pág. 72.

T

taxonomía Es una categoría o etiqueta que se le da a una instancia de *post type*. [50], pág. 40.

W

widget Es un pequeñísimo programa que nos da acceso a funciones que usamos normalmente. [53], pág. 41.

WordPress Es un sistema de gestión de contenidos libre y gratuito, pág. 38.

World Wide Web Consortium Es un consorcio internacional que produce recomendaciones para la *World Wide Web*. [41], pág. 37.

wysiwyg Es el acrónimo de *What You See Is What You Get* (en español, “lo que ves es lo que obtienes”). Se aplica a los procesadores de texto y otros editores de texto con formato (como los editores de HTML) que permiten escribir un documento viendo directamente el resultado final, pág. 72.

X

XML Sitemaps Google Sitemaps es una herramienta de Google que permite una mejor búsqueda y posicionamiento en su buscador. El *sitemap* no es más que un archivo con un listado de páginas e información sobre su contenido, actualizaciones, etc. el archivo puede tener varios formatos pero el más utilizado en la actualidad es el XML, pág. 20.

Bibliografía

- [1] Alex Dunae. Wp smush.it. <https://wordpress.org/plugins/wp-smushit/>. [Consulta: 30 de Octubre de 2014].
- [2] Alex Mills. Regenerate thumbnails. <https://wordpress.org/plugins/regenerate-thumbnails/>. [Consulta: 30 de Octubre de 2014].
- [3] Alex Moravek. Seo y el redireccionamiento permanente 301. <http://www.seoagencias.com/posicionamiento-web-redireccionamiento/>. [Consulta: 24 de Octubre de 2014].
- [4] Altassian. Bitbucket. <https://bitbucket.org/>. [Consulta: 24 de Octubre de 2014].
- [5] Appcelerator, Inc. Aptana. <http://www.aptana.com/>. [Consulta: 24 de Octubre de 2014].
- [6] Bootstrap Team. Bootstrap. <http://getbootstrap.com/>. [Consulta: 24 de Octubre de 2014].
- [7] Bureau of Labor Statics. Computer programmers. <http://www.bls.gov/ooh/computer-and-information-technology/computer-programmers.htm#tab-1>. [Consulta: 25 de Noviembre de 2014].
- [8] Cacao Team. Cacao. <https://cacao.com/>. [Consulta: 24 de Octubre de 2014].
- [9] ComputerWorld España. El ciclo de vida útil de un pc ha quedado reducido a tan sólo tres años, según idc. <http://www.computerworld.es/archive/el-ciclo-de-vida-util-de-un-pc-ha-quedado-reducido-a-tan-solo-tres-anos-segun-idc>. [Consulta: 25 de Noviembre de 2014].
- [10] Culturaweb. Culturaweb. <http://www.culturaweb.com/>. [Consulta: 24 de Octubre de 2014].
- [11] Dormando. Memcached. <http://memcached.org/>. [Consulta: 30 de Octubre de 2014].
- [12] Git Team. Git. <http://git-scm.com/>. [Consulta: 24 de Octubre de 2014].
- [13] InfoJobs. ¿cómo es la trayectoria profesional de ingeniero informático? <http://plandecarrera.infojobs.net/puesto-de-trabajo/ingeniero-informatico>. [Consulta: 25 de Noviembre de 2014].
- [14] Javier Eguiluz. Introducción a javascript. <http://librosweb.es/javascript/>. [Consulta: 24 de Octubre de 2014].
- [15] Jonathan Sampson. Unexpected results while styling wordpress menus. http://sampsonblog.com/tag/menu_class. [Consulta: 24 de Octubre de 2014].

- [16] Joost de Valk. Wordpress seo by yoast. <https://wordpress.org/plugins/wordpress-seo/>. [Consulta: 30 de Octubre de 2014].
- [17] jQuery Foundation. jquery. <http://jquery.com/>. [Consulta: 24 de Octubre de 2014].
- [18] Juan Palacio. Mejorando las estimaciones basadas en juicio de expertos. <http://www.navegapolis.net/content/view/347/99/>. [Consulta: 30 de Octubre de 2014].
- [19] Justin Sternberg. Custom metaboxes and fields for wordpress. <https://github.com/WebDevStudios/Custom-Metaboxes-and-Fields-for-WordPress/wiki/Basic-Usage>. [Consulta: 24 de Octubre de 2014].
- [20] Kent Beck. Manifiesto por el desarrollo Ágil de software. <http://agilemanifesto.org/iso/es/>. [Consulta: 24 de Octubre de 2014].
- [21] Kioskea. Protocolo ftp. <http://es.kioskea.net/contents/263-protocolo-ftp-protocolo-de-transferencia-de-archivos>. [Consulta: 24 de Octubre de 2014].
- [22] LESS Team. Less. <http://lesscss.org/>. [Consulta: 24 de Octubre de 2014].
- [23] OnTheGoSystems, Inc. Wpml. <http://wpml.org/es/>. [Consulta: 30 de Octubre de 2014].
- [24] Oracle Corporation. Mysql. <http://www.mysql.com/>. [Consulta: 24 de Octubre de 2014].
- [25] PuroMarketing. Los dispositivos móviles ya generan casi un tercio del tráfico web. <http://www.puromarketing.com/12/18569/dispositivos-moviles-generan-casi-tercio-trafico.html>. [Consulta: 24 de Octubre de 2014].
- [26] Rakhitha Nimesh. Wordpress initialization hooks. <http://code.tutsplus.com/articles/wordpress-initialization-hooks-benefits-and-common-mistakes--wp-34427>. [Consulta: 24 de Octubre de 2014].
- [27] Sergej Muller. Antispam bee. <https://wordpress.org/plugins/antispam-bee/>. [Consulta: 30 de Octubre de 2014].
- [28] Subash Pathak. Prepos app. <http://alphapixels.com/prepros/>. [Consulta: 24 de Octubre de 2014].
- [29] Takayuki Miyoshi. Contact form 7. <https://wordpress.org/plugins/contact-form-7/>. [Consulta: 24 de Octubre de 2014].
- [30] The Document Foundation. Libreoffice. <https://es.libreoffice.org/>. [Consulta: 24 de Octubre de 2014].
- [31] The Eclipse Foundation. Eclipse. <https://www.eclipse.org/>. [Consulta: 30 de Octubre de 2014].
- [32] The PHP Group. Apc. <http://php.net/manual/es/intro.apc.php>. [Consulta: 30 de Octubre de 2014].
- [33] The PHP Group. Php. <http://php.net/>. [Consulta: 24 de Octubre de 2014].
- [34] The Rocket Science Group. Mailchimp. <http://mailchimp.com>. [Consulta: 30 de Octubre de 2014].

- [35] Toni Santo-Regis. Latex. <http://www.latex-project.org/>. [Consulta: 24 de Octubre de 2014].
- [36] Tungaloy. Tungaloy. <http://www.tungaloy.com>. [Consulta: 24 de Octubre de 2014].
- [37] U.S. News. Computer programmer: Salary. <http://money.usnews.com/careers/best-jobs/computer-programmer/salary>. [Consulta: 25 de Noviembre de 2014].
- [38] W3C. Css3. <http://www.w3.org/Style/CSS/current-work>. [Consulta: 24 de Octubre de 2014].
- [39] W3C. Document object model. <http://www.w3.org/DOM/>. [Consulta: 24 de Octubre de 2014].
- [40] W3C. Html5. <http://www.w3.org/TR/html5/>. [Consulta: 24 de Octubre de 2014].
- [41] W3C. World wide consortium. <http://www.w3c.es/>. [Consulta: 24 de Octubre de 2014].
- [42] WebDevStudios. Field types. <https://github.com/webdevstudios/Custom-Metaboxes-and-Fields-for-WordPress/wiki/Field-Types>. [Consulta: 24 de Octubre de 2014].
- [43] WordPress. Function reference. register post type. http://codex.wordpress.org/Function_Reference/register_post_type. [Consulta: 24 de Octubre de 2014].
- [44] WordPress. Functions file explained. http://codex.wordpress.org/Functions_File_Explained. [Consulta: 24 de Octubre de 2014].
- [45] WordPress. Pages. <http://codex.wordpress.org/es:Pages>. [Consulta: 30 de Octubre de 2014].
- [46] WordPress. Post types. http://codex.wordpress.org/Post_Types. [Consulta: 24 de Octubre de 2014].
- [47] WordPress. Query posts. http://codex.wordpress.org/Function_Reference/query_posts. [Consulta: 24 de Octubre de 2014].
- [48] WordPress. Rewrite api. http://codex.wordpress.org/Rewrite_API/add_rewrite_rule. [Consulta: 24 de Octubre de 2014].
- [49] WordPress. Shortcode api. http://codex.wordpress.org/Shortcode_API. [Consulta: 30 de Octubre de 2014].
- [50] Wordpress. Taxonomies. <http://codex.wordpress.org/Taxonomies>. [Consulta: 24 de Octubre de 2014].
- [51] WordPress. Translate wordpress. <https://make.wordpress.org/polyglots/handbook/translating/working-with-core/>. [Consulta: 24 de Octubre de 2014].
- [52] WordPress. Wordpress nav menu. http://codex.wordpress.org/Function_Reference/wp_nav_menu. [Consulta: 24 de Octubre de 2014].
- [53] WordPress. Wordpress widgets. http://codex.wordpress.org/WordPress_Widgets. [Consulta: 24 de Octubre de 2014].

- [54] WordPress StackExchange. Wordpress nav menu's items wrap argument. <http://wordpress.stackexchange.com/questions/19245/any-docs-for-wp-nav-menus-items-wrap-argument>. [Consulta: 24 de Octubre de 2014].

Anexo A

Ejemplos de código Wordpress

En este anexo vemos ejemplos de cómo crear: *custom post types*, *shortcodes* y menús. Utilizamos HTML, CSS y PHP, el lenguaje usado para programar WordPress.

A.1. Creación de *custom posts types*

Para explicar la creación de *customs posts types* vamos ver dos ejemplos en los que mostraremos paso a paso cómo hemos creado el *post type Events*, uno de los más simples, y el *post type Product*, uno de los más complejos.

A.1.1. Creando el *custom post type Events*

Para crear un *custom post type* tenemos que llamar al método *register_post_type()* y pasarle como argumentos el nombre del nuevo *post type* y un vector con los atributos que ha de tener.

El código de la Figura A.1 crea un nuevo *post type* llamado *Events* identificado como *cw_events*. Primero se crea una función encargada de la creación del *post type* y luego (en la línea 14) se indica que esta sea llamada junto a otros procesos de inicialización, justo después de que WordPress termine de cargar[26].

```
1 function cw_events_init() {
2     $args = array(
3         'public' => true,
4         'label' => __('Events', 'culturaweb'),
5         'hierarchical' => true,
6         'supports' => array(
7             'title'
8         ),
9         'rewrite' => array('slug' => 'event'),
10    );
11    register_post_type('cw_events', $args);
```

```

12     flush_rewrite_rules(); // Check out
13 }
14 add_action('init', 'cw_events_init');\label{fig:eje}

```

Figura A.1: Código para crear el *custom post type events*

Como vemos en la línea 1 de la Figura A.1, siguiendo la convención de la empresa, a las funciones les damos el prefijo *cw* haciendo referencia al nombre de la misma (culturaweb), es también una práctica recomendada por WordPress. En la línea 11 vemos la llamada al método que crea el *custom post type*, este recibe dos argumentos: el identificador del *post type* y un vector con las características o atributos que tendrá.

Veamos el significado de cada elemento del vector de argumentos [43]:

1. El primer elemento del vector, *public true*, indica que el nuevo *post type* esté disponible desde el panel de administración de WordPress.
2. El segundo elemento, *label*, indica el nombre que se mostrará en el panel de administración para el *post type*. Para que se muestre diferente dependiendo del idioma hemos utilizado la función de traducción *__(texto, dominio)* que devolverá el texto que hayamos indicado como traducción dentro de un determinado dominio. Las traducciones a cada texto han de ser indicadas en *Portable Object files*.
3. En la línea 5 se indica que se permitirán jerarquías entre eventos, de manera que un evento pueda tener eventos hijos.
4. En el cuarto elemento, el *array supports* indica qué características habilitar para la edición del *post type*. Por ejemplo: *title* hace que tengamos disponible una caja de texto para editar nuestros atributos, aquí no se asigna ningún método de edición, simplemente se dice que cualquiera de sus atributos podrá usar los métodos de edición que aquí se incluyan.
5. Por último, *rewrite* permite modificar algunos parámetros[48], en este caso indicamos que el *slug* para el *post type* sea *event* en lugar del de por defecto.

Al final de la función, en la línea 11, llamamos a la función *flush_rewrite_rules* para forzar el refresco de las reglas de enlaces permanentes. Estas reglas permiten que podamos hacer referencia a *post types* y otros elementos. Al forzar el refresco conseguimos que el nuevo *post type* aparezca en estas reglas y así podamos hacer referencia a él en otras partes de nuestro código.

Una vez creado el *custom post type* tenemos que indicar cómo queremos que este pueda ser editado, es decir, si queremos que nos permita editar determinado atributo con una caja de texto, con una caja para edición de fechas, números, con una caja con funciones de edición avanzadas, etc. Para ello tendremos que crear *metaboxes* personalizados. Un *metabox* nos permite añadir campos extra a la página de edición. En la Figura A.2 podemos ver un *metabox* de ejemplo, en el ejemplo se permite la edición de un título mediante una caja de texto simple y la edición del cuerpo mediante una caja de edición con funciones avanzadas del tipo *wysiwyg*.

En la Figura A.3 encontramos el código que añade las cajas de edición que necesitamos para la edición de nuestro nuevo *post type events*[19].

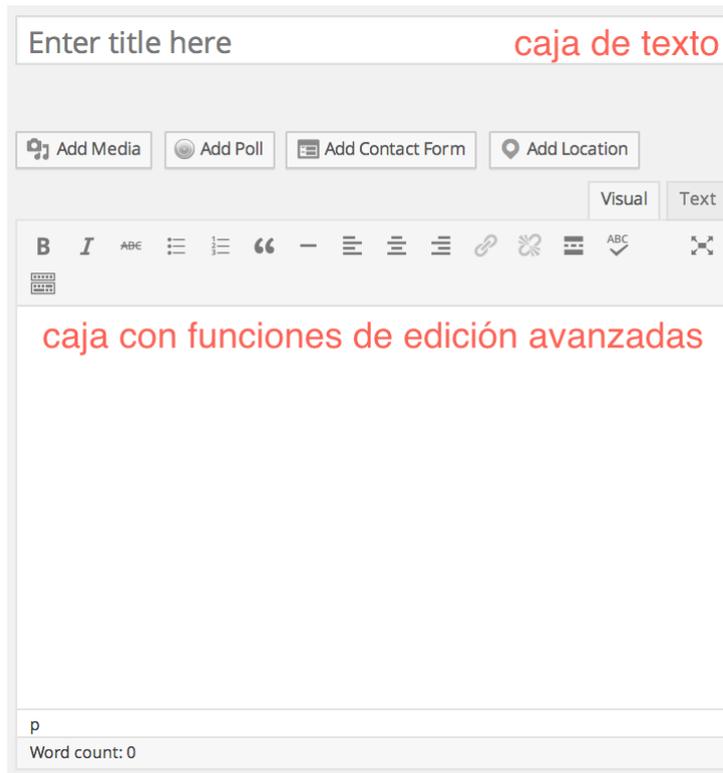


Figura A.2: Ejemplo de *metabox*.

```

1 function cw_event_metaboxes(array $meta_boxes) {
2     // Start with an underscore to hide fields from custom fields
3     list
4     $prefix = '_event_';
5     $meta_boxes['cw_events'] = array(
6         'id'                => 'cw_events',
7         'title'             => __('Events', 'culturaweb'),
8         'pages'             => array('cw_events'), // Post
9         'type'              => 'normal',
10        'context'            => 'high',
11        'priority'           => true,
12        'show_names'         => array(
13            array(
14                'name'        => __('Event Date', 'culturaweb')
15                ,
16                'desc'        => __('Select the event date.', '
17                culturaweb'),
18                'id'          => $prefix . 'cw_event_date',
19                'type'        => 'text_medium',
20            ),
21            array(
22                'name'        => __('Event Place', 'culturaweb')
23                ,
24                'desc'        => __('', 'culturaweb'),
25                'id'          => $prefix . 'cw_event_place',

```

```

22         'type'      => 'text_medium',
23     ),
24     array(
25         'name'      => __('Event Stand', 'culturaweb'),
26         'desc'      => __('', 'culturaweb'),
27         'id'        => $prefix . 'cw_event_stand',
28         'type'      => 'text_medium',
29     ),
30     array(
31         'name'      => __('Event URL', 'culturaweb'),
32         'desc'      => __('', 'culturaweb'),
33         'id'        => $prefix . 'cw_event_url',
34         'type'      => 'text_url',
35     ),
36 ),
37 );
38 return $meta_boxes;
39 }
40 add_filter('cmb_meta_boxes', 'cw_event_metaboxes');

```

Figura A.3: Código para añadir *metaboxes* al *custom post type events*

Como vemos en la línea 1 (Figura A.3) en esta función también se sigue con la convención de la empresa para la definición de nombres de funciones. Más adelante, en la línea 3 definimos un prefijo que se añade al identificador de todos los campos del *metabox* que creemos para este *post type*. En la línea 4 empezamos a crear un vector con la información del *metabox* y a partir de la línea 11 definimos todos los campos de edición de nuestro *metabox*. Para cada campo se establece:

1. El nombre del campo, mediante el parámetro *name* y utilizando una función de traducción para darle el nombre deseado según el idioma.
2. La descripción del campo que se muestra al usuario como ayuda.
3. Un *id* para utilizarlo nosotros dentro del código y hacer referencia al campo del *metabox*.
4. Y por último, el tipo de edición para el campo (caja de texto simple, caja *wysiwyg*, etc.). En este caso hemos utilizado para todos los campos cajas de texto de tamaño mediano especificando el tipo *text_medium* y una caja del tipo *text_url* que automáticamente muestra su contenido con un enlace y comprueba que el texto introducido tenga forma de URL[42].

Finalmente, en la línea 40, añadimos la función que hemos creado para personalizar el *metabox* del *custom post type events* al vector de *metaboxes* de WordPress [19].

A.2. Creación de *shortcodes*

Podemos utilizar *shortcodes* para fines muy variados, por este motivo, en los siguientes puntos hemos explicado la creación de tres *shortcodes* diferentes, uno muy simple y dos algo

más complejos.

A.2.1. *Teaser shortcode*

El primer *shortcode* que vamos a programar simplemente añade formato a un título y un subtítulo. Le hemos llamado *teaser* que en español se traduciría como un título que pretende enganchar al lector. Este *shortcode* nos permite mostrar un título como el de la Figura A.4 escribiendo en el editor de WordPress el siguiente código:

```
[teaser title='título' subtitle='subtítulo']
```



Figura A.4: *Shortcode teaser*

Sin este *shortcode* no podríamos añadir un título como el de la Figura A.4 a nuestro sitio a no ser que utilizásemos una herramienta externa, pasásemos el título a imagen y entonces lo añadiésemos al editor. Este *shortcode* facilita la tarea al usuario y permite conseguir títulos con este formato simplemente mediante la introducción de texto.

Veamos cómo hemos creado el *shortcode*. El código HTML necesario para el resultado de la Figura A.4 es el siguiente:

```
1 <h1 class="text-center text-uppercase">
2   Título
3   <span class="text-big-red">Subtítulo</span>
4   <span class="line-big-red"></span>
5 </h1>
```

Figura A.5: Código HTML para *teaser*

Aunque no es relevante para la creación del *shortcode*, las clases CSS que se utilizan en el código anterior se definen a continuación en la Figura A.6.

```
1 .text-center { text-align: center; }
2
3 h1 {
4   font-family: @font-title;
5   letter-spacing: 5px;
6   line-height: 0.6;
7   margin-bottom: 40px;
```

```

8   margin-top: 60px;
9   text-align: center;
10  text-transform: uppercase;
11
12  .line-big-red {
13    background: @corporate-red;
14    bottom: 10px;
15    height: 1px;
16    left: 10%;
17    position: absolute;
18    width: 80%;
19    z-index: 5;
20  }
21
22  .text-big-red {
23    background: white;
24    color: @corporate-red;
25    display: inline-block;
26    font-family: @font-subtitle;
27    font-size: 55px;
28    font-weight: 400;
29    letter-spacing: -1px;
30    padding: 0 10px;
31    position: relative;
32    text-transform: capitalize;
33    z-index: 10;
34
35    &:before { content: "\201C"; }
36    &:after { content: "\201D"; }
37  }
38
39  @media (max-width: @screen-sm) {
40    font-size: 50px;
41    .text-big-red { font-size: 35px; }
42  }
43 }

```

Figura A.6: Código CSS para *teaser* escrito en LESS

Una vez visto el código que necesitamos para crear el título que queremos hacer disponible mediante *shortcodes*, veamos cómo implementar el *shortcode teaser*.

```

1  function culturaweb_teaser($atts) {
2      extract(shortcode_atts(array('title' => '', 'subtitle' => ''),
3          $atts));
4      return "
5          <h1 class=\"text-center text-uppercase\">$title
6          <span class=\"text-big-red\">$subtitle</span><span
7              class=\"line-big-red\"></span></h1>
8          ";
9  }
10 add_shortcode('teaser', 'culturaweb_teaser');

```

Figura A.7: Código PHP para crear el *shortcode teaser*

En la Figura A.7 vemos la función encargada de interpretar el código `[teaser title="título" subtitle="subtítulo"]` y traducirlo a HTML. En la línea 2 vemos cómo se definen los parámetros que introduce el usuario, `title` y `subtitle`, se les asigna una cadena vacía como valor por defecto. En la línea 3 ya vemos el `return` de la función que devuelve el mismo código HTML de la Figura A.5 pero con las variables que contienen las cadenas a mostrar. Por último en la línea 8 añadimos el `shortcode` con la función `add_shortcode("nombreDelShortcode", "funcionQueDefineElShortcode")`.

A.2.2. *What's New shortcode*

El siguiente *shortcode* que vamos a implementar muestra las tres últimas entradas del *post type* por defecto: entrada, o *post* en inglés. Para ello hacemos una consulta a la base de datos de WordPress y mostramos la información con el aspecto de la Figura A.8.

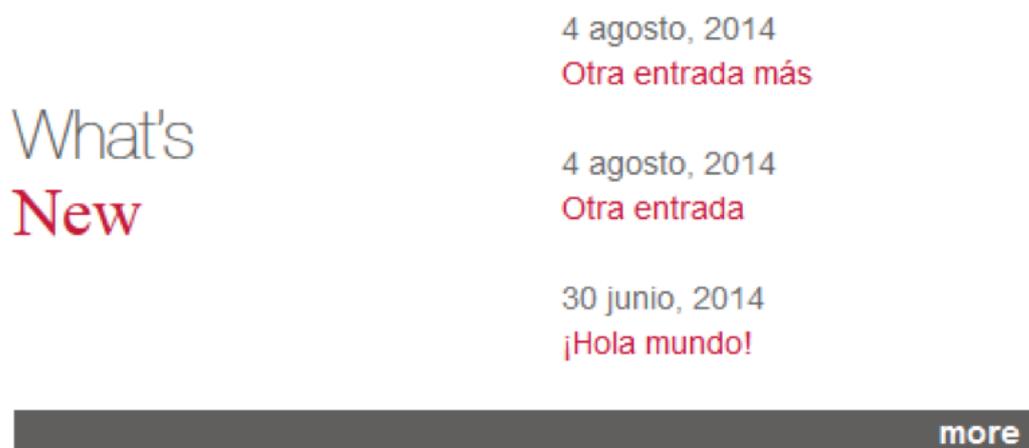


Figura A.8: *Shortcode What's New*

El objetivo de crear este *shortcode* es que el usuario pueda colocar en cualquier lugar del sitio web un cuadro de noticias como el de la Figura A.8, sin un *shortcode* sería imposible hacerlo, en cambio permitimos al usuario esta tarea de modo que pueda lograrlo introduciendo el código que se muestra a continuación en el editor de WordPress.

```
[whatsnew]
```

El código HTML necesario para mostrar un cuadro como el de la Figura A.8 es el siguiente:

```
1 <div class="row">
2   <div class="col-lg-6 col-md-6 col-sm-12">
3     <h2 class="text-left text-capitalize font-tittle">
4       "What 's <span class="text-big-red" >new"</span>
5     </h2>
```

```

6     </div>
7     <div class="col-lg-6 col-md-6 col-sm-12">
8         <ul class="list-unstyled">
9             <li><p><time class="box" datetime="2001-05-15 19:00">May 15</time
10                >
11                Titulo entrada</p></li>
12             <li><p><time class="box" datetime="2001-05-15 19:00">May 15</time
13                >
14                Titulo entrada 2</p></li>
15             <li><p><time class="box" datetime="2001-05-15 19:00">May 15</time
16                >
17                Titulo entrada 3</p></li>
18         </ul>
19     </div>
20 </div>
21 <div class="bar-grey"><a href="#">more</a></div>

```

Figura A.9: Código HTML para *What's New shortcode*

En esta ocasión omitiremos las clases CSS que se utilizan en el HTML de la Figura A.9 ya que no son relevantes para la creación del *shortcode* y lo importante de este no es su resultado visual, sino la consulta que se realiza a la base de datos.

Veamos cómo crear el *shortcode*:

```

1 function culturaweb_whats_new($atts) {
2     extract(shortcode_atts(array('title' => ''), $atts));
3
4     $args = array(
5         'numberposts' => 3,
6         'post_type' => 'post',
7     );
8     $query = new WP_Query($args);
9     while ($query->have_posts()) {
10         $query->the_post();
11         $news[] = '<li><p><time datetime="' . get_the_time('c')
12                 . '">' . get_the_date() . '</time></br>' . '<a href
13                 =' . get_the_permalink() . '">' . get_the_title() .
14                 '</a></p></li>';
15     }
16     wp_reset_postdata();
17
18     return '
19         <div class="row">
20             <div class="col-lg-6 col-md-6 col-sm-12">
21                 <h2 class="text-left text-capitalize font-tittle">
22                     ' . __('What's', 'culturaweb') .
23                     ' <span class="text-color-corporate font-subtitle
24                     " >
25                     ' . __('New', 'culturaweb') . '
26                 </span>
27                 </h2>
28             </div>
29         </div>

```

```

25         <div class="col-lg-6 col-md-6 col-sm-12">
26             <ul class="list-unstyled">' . implode("\n", $news)
                . ' </ul>
27         </div>
28     </div>
29     <div class="bar-grey"><a href="#">' . __('more', '
                culturaweb') . '</a></div></br>';
30 }
31 add_shortcode('whats_new', 'culturaweb_whats_new');

```

Figura A.10: Código PHP para crear el *shortcode* *What's New*

En este *shortcode* no necesitamos que se le pasen argumentos, como el título y el subtítulo del *shortcode* del apartado anterior, aún así hemos definido el parámetro *title* por si se le quiere asignar un identificador. En la línea 8 creamos la consulta con los argumentos especificados en el vector *args* de la línea 4. Se trata de una consulta muy sencilla en la que solamente indicamos el tipo de *post* que queremos obtener (línea 6) y la cantidad de filas de la tabla que queremos obtener (línea 5).

Más adelante, desde la línea 9 hasta la línea 13 leemos la información obtenida en la consulta mediante un bucle y tratamos los datos. Los bucles para leer consultas de Wordpress siempre tienen la misma forma:

```

1 while ($query->have_posts()) { // mientras haya filas en la consulta
2     $query->the_post(); // obtiene la siguiente fila de la
        consulta
3     /** Tratamos la información de la fila **/
4 }
5 wp_reset_postdata(); // restaura la variable global $post

```

Figura A.11: Bucle para leer datos de una consulta a la BDD de WordPress

Volvamos a la Figura A.10. Dentro del bucle de lectura de la consulta, en la línea 11, vemos el tratamiento que le hemos dado a cada fila de la consulta. Para cada fila hemos añadido una fila al vector *news* con el código HTML necesario para mostrar la información de la fila. Para generar el HTML con los valores de cada fila hemos utilizado funciones[47] de Wordpress (como *get_the_date()*, *get_the_title()*, etc.) que nos permiten acceder a los campos del *post type* o de la fila. Por último, en la línea 15 empieza el *return* de la función con todo el código estático a cada ejecución y con la función *implode* en la línea 26 que añade el contenido que hayamos añadido al vector *news* con la información dependiente de la base de datos, es decir, con la información obtenida de la consulta. En la línea 19, 21 y 29 se utiliza una función que ya hemos nombrado con anterioridad, función encargada de la traducción de pequeñas cadenas. Por último, al igual que en el *shortcode* anterior, en la última línea se añade el *shortcode* a Wordpress.

A.2.3. *Tabs shortcode*

El último *shortcode* que vamos a programar permite al usuario crear pestañas, un elemento de la interfaz gráfica que permiten cambiar rápidamente lo que se está viendo sin cambiar de

ventana. Las pestañas tendrán el aspecto que se muestra en la Figura A.12.



Figura A.12: *Shortcode tabs*

Para que el usuario pueda crear tantas pestañas como quiera necesitaremos un *shortcode* compuesto. El código a escribir en el editor de WordPress para obtener unas pestañas como las de la Figura A.12 es el siguiente:

```
[tabgroup]
  [tab title='Tab 1'] Tab 1 content goes here. [/tab]
  [tab title='Tab 2'] Tab 2 content goes here. [/tab]
  [tab title='Tab 3'] Tab 3 content goes here. [/tab]
[/tabgroup]
```

Como podemos apreciar, se trata de un *shortcode* compuesto porque necesitaremos crear un *shortcode tabgroup* y otro *shortcode tab*. Si no utilizáramos un *shortcode* compuesto no habría manera de permitir al usuario añadir tantas pestañas como quiera. Fijémonos también en que estos *shortcodes* se abren y se cierran, en los *shortcodes* anteriores sólo teníamos código de apertura, en cambio estos también tienen un código para cerrarlos que utiliza la misma sintaxis pero con una barra “/” antes del nombre del *shortcode*.

El código HTML necesario para implementar las pestañas se muestra en la Figura A.13. Las clases CSS que se utilizan son clases de Bootstrap que, junto a sus librerías JavaScript, nos ofrecen la funcionalidad de cambiar de contenidos al clicar cada pestaña, nosotros hemos modificado estas clases para adaptar su visualización a nuestro diseño.

El cambio de contenidos se realiza a partir de los *ids* del contenido y la referencia de las pestañas, en las líneas 2 y 7 vemos un ejemplo. Al crear el *shortcode* hemos de prestar especial atención a los *ids*, ya que no pueden existir dos elementos HTML con el mismo *id*. Si queremos que el *shortcode* permita la creación de más de un conjunto de pestañas que funcionen correctamente tendremos que asignar *ids* diferentes para cada ejecución del *shortcode*.

```
1 <ul class="nav nav-tabs" role="tablist" data-tabs="tabs">
2   <li class="active"><a href="#tab1" data-toggle="tab">Tab 1</a></li>
3   <li class=""><a href="#tab2" data-toggle="tab">Tab 2</a></li>
4   <li class=""><a href="#tab3" data-toggle="tab">Tab 3</a></li>
5 </ul>
6 <div class="tab-content">
7   <div class="tab-pane active" id="tab1">Tab 1 content goes here.</div>
8   <div class="tab-pane " id="tab2">Tab 2 content goes here.</div>
9   <div class="tab-pane " id="tab3">Tab 3 content goes here.</div>
10 </div>
```

Figura A.13: Código HTML para *Tabs shortcode*

Para programar nuestro *shortcode* compuesto necesitamos crear dos *shortcodes* que trabajen

juntos. Veamos como quedan las funciones que implementan estos dos *shortcodes*:

```
1 function culturaweb_tabgroup($atts, $content = null) {
2     extract(shortcode_atts(array('title' => ''), $atts));
3
4     $randomid = rand();
5     $GLOBALS['tab_count'] = (isset($GLOBALS['tab_count']))
6         ? $GLOBALS['tab_count'] + 1
7         : 0;
8
9     do_shortcode($content);
10
11     if (is_array($GLOBALS['tabs'])) {
12         $i = 0;
13         foreach ($GLOBALS['tabs'] as $tab) {
14             $active = (reset($GLOBALS['tabs']) == $tab) ? '
15                 active' : '';
16
17             $tabs[] = '<li class="' . $active . '"><a href
18                 ="#tab' . $randomid.$i . '" data-toggle="tab
19                 ">' . $tab['title'] . '</a></li>';
20             $panes[] = '<div class="tab-pane ' . $active .
21                 '" id="tab' . $randomid.$i . '">' . $tab['
22                 content'] . '</div>';
23
24             $i++;
25         }
26
27         $return = '
28             <ul class="nav nav-tabs" role="tablist" data-
29                 tabs="tabs">
30                 ' . implode("\n", $tabs) . '
31             </ul>
32             <div class="tab-content">
33                 ' . implode("\n", $panes) . '
34             </div>
35         ';
36     }
37     return $return;
38 }
39 add_shortcode('tabgroup', 'culturaweb_tabgroup');
```

Figura A.14: Código PHP para *Tabgroup shortcode*

```
1 function culturaweb_tab($atts, $content = null) {
2     extract(shortcode_atts(array('title' => ''), $atts));
3     $x = $GLOBALS['tab_count'];
4     $GLOBALS['tabs'][$x] = array(
5         'title' => sprintf($title, $x),
6         'content' => do_shortcode($content)
7     );
8     $GLOBALS['tab_count']++;
9
10 }
```

```
11 add_shortcode('tab', 'culturaweb_tab');
```

Figura A.15: Código PHP para *Tab shortcode*

Veamos primero el *shortcode* más corto: *tab*. En la definición del método de la Figura A.15 (línea 1) encontramos la primera diferencia respecto a los *shortcodes* que hemos visto anteriormente. En la cabecera no definimos solamente el parámetro *atts* para los atributos que nos pasen con el *shortcode*, también se define el parámetro *content*, este parámetro indica que el *shortcode* tendrá código de cierre y que la información contenida entre el código de apertura y el código de cierre será pasada al método a través de esta variable. En la segunda línea del método, como en los *shortcodes* anteriores, definimos los campos que se le podrán pasar como argumentos. Este *shortcode* será siempre utilizado en conjunto al *shortcode tabgroup*, ambos comparten variables utilizando las variables globales de PHP que podemos almacenar en el vector *GLOBALS*. En la línea 3 extraemos de la variable global *tab_count* el número de pestaña para la que se está ejecutando este *shortcode* interno. A continuación, en la línea 4, creamos una variable global para la pestaña donde almacenamos su contenido para que el otro *shortcode* (*tabgroup*) pueda extraerlo y utilizarlo. Después incrementamos la variable global *tab_count* para que este lista con el valor de la posible próxima pestaña, y en la última línea añadimos el *shortcode* a WordPress. Fijémonos en la línea 6, utilizamos la función *do_shortcode()* sobre la variable con el contenido, el objetivo de esto es permitir que dentro del contenido que se muestra en cada pestaña también se puedan utilizar otros *shortcodes*.

Pasemos a la implementación del *shortcode tabgroup* en la Figura A.14. Ya conocemos las primeras líneas. Más adelante, en la línea 4 generamos un número aleatorio para utilizarlo como sufijo en los *ids*, ya que como hemos dicho antes, en HTML no pueden existir dos elementos con el mismo *id* y el *shortcode* ha de poder ejecutarse tantas veces como quiera el usuario. Desde la línea 5 hasta la línea 7 vemos cómo inicializamos la variable global *tab_count* que utilizábamos en el *shortcode* anterior, la inicializamos a 0 si todavía no ha sido utilizada o le sumamos uno si ya la habíamos inicializado con anterioridad.

En la línea 9 utilizamos de nuevo la función *do_shortcode()* para permitir el uso de otros *shortcodes* en el contenido de este *shortcode*, esta función ejecutada en este punto provocará que se ejecuten todos los *shortcodes tab* de dentro de *tabgroup*, es decir, para cada pestaña se ejecutará el código que hemos comentado antes (Figura A.15).

En este punto el *shortcode tab* ya se ha ejecutado para cada pestaña y ha almacenado el título y el contenido de cada pestaña en la variable global *tab*, en la línea 11 comprobamos que la variable tenga contenido. Si la variable global tiene contenido, para cada elemento del vector creamos el código HTML necesario y almacenamos en el vector *tab* para el código necesario para crear las pestañas en HTML y en el vector *panes* el código para el contenido que han de mostrar las pestañas.

Finalmente, en el *return*, como en *shortcodes* explicados anteriormente, insertamos el código HTML estático necesario junto a las llamadas al método *implode* para insertar el código dependiente de cada ejecución que hemos generado previamente en el bucle *for*. Como siempre, al final, en la última línea, añadimos el *shortcode* a WordPress.

A.3. Creación de menús personalizados

En este punto vamos a ver un ejemplo de cómo crear menús personalizables para nuestra plantilla, de modo que los usuarios puedan añadir y eliminar los elementos que quieran.

En nuestro ejemplo queremos crear el menú de la Figura A.16. En este menú el usuario tiene que poder modificar los elementos de la barra gris: añadir y eliminar elementos.

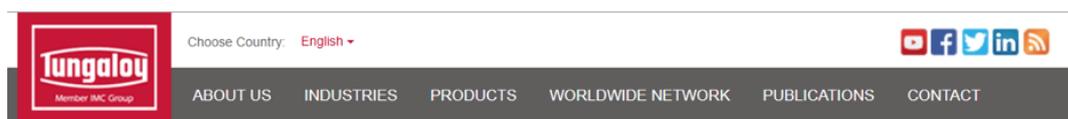


Figura A.16: Menú personalizable (parte de color gris)

El código HTML simplificado para obtener el menú se muestra en la Figura A.17, a este código le faltarían algunas clases y etiquetas más para lograr el aspecto de la Figura A.16 pero vamos a obviarlas por simplicidad.

```
1 <ul class="nav navbar-nav">
2   <li class="active"><a href="#">About us</a></li>
3   <li><a href="#">Industries</a></li>
4   <li><a href="#">Products</a></li>
5   <li><a href="#">Worldwide network</a></li>
6   <li><a href="#">Publications</a></li>
7   <li><a href="#">Contact</a></li>
8 </ul>
```

Figura A.17: Código HTML para menú

Cómo ya podremos imaginar necesitamos un código PHP que añada o elimine elementos `` al código HTML de la Figura A.17. WordPress ofrece funcionalidad para esto. En su panel de administración (Figura A.18) hay un apartado para la gestión de menús, sólo hemos de crear el menú desde el asistente gráfico, desde donde el usuario podrá añadir y eliminar elementos, y escribir el código PHP que imprima en código HTML todos los elementos del menú. Para imprimir los elementos `` de un menú WordPress nos proporciona la función `wp_nav_menu`.

En la Figura A.19 vemos cómo hemos utilizado esta función para leer nuestro menú de la base de datos de WordPress. En la línea 4 llamamos al método encargado de la extracción de datos de menús creados a través de la interfaz gráfica, veamos el significado de cada argumento[52]:

- En la línea 4, al parámetro `menu` le pasamos el `id`, `slug` o nombre (se buscará el menú en este orden) del menú deseado.
- El parámetro `container` nos permite añadir una etiqueta externa al ``, en este caso no añadimos ninguna.
- En la línea 6, `menu_class` añade una clase al ``[15].

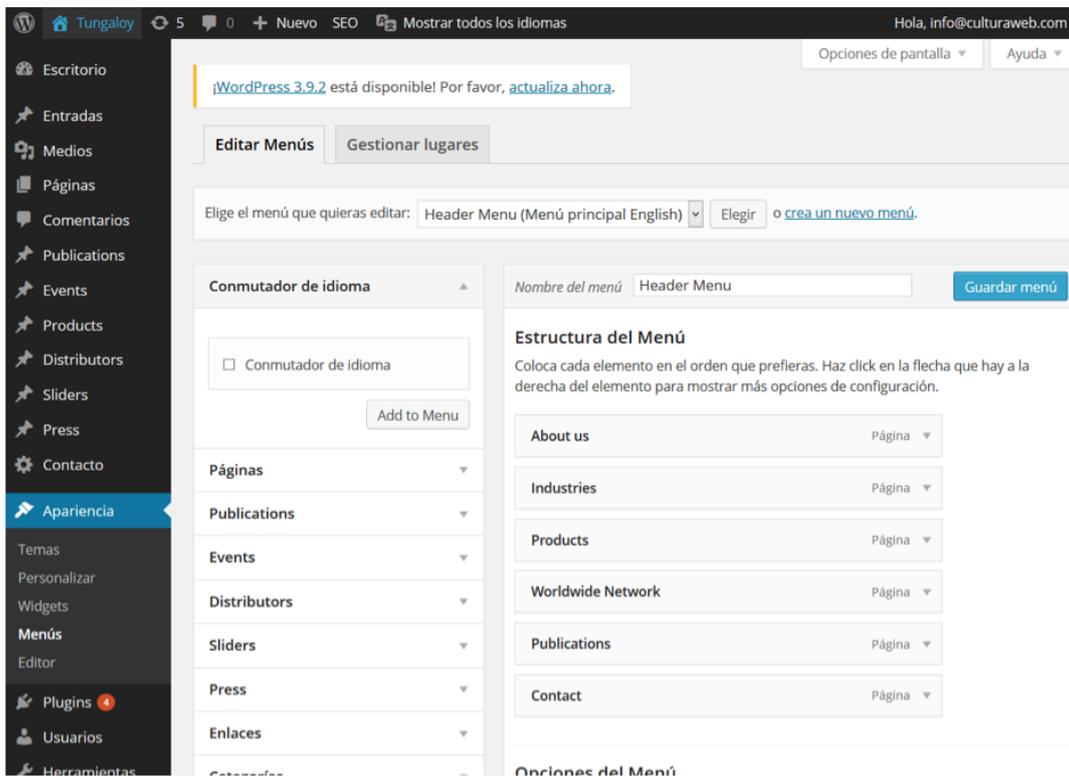


Figura A.18: Panel de administración

- En la línea 7, podemos añadir un *id* al elemento `` de menú si lo deseamos, en nuestro caso tampoco no lo añadiremos.
- Línea 8, *items_wrap* para indicar qué argumentos o elementos se deben imprimir en HTML[54], en este indicamos que sólo se imprimirá el contenido de los ``.
- Finalmente, en la línea 9, indicamos el número de niveles de jerarquía permitido.

```

1 <ul class="nav navbar-nav">
2 <?php
3     wp_nav_menu(array(
4         'menu' => 'Main Menu',
5         'container' => '',
6         'menu_class' => 'primary-navigation',
7         'menu_id' => '',
8         'items_wrap' => '%3$s',
9         'depth' => 1
10    ));
11 <?>
12 </ul>

```

Figura A.19: Código PHP para menú

El código que hemos explicado anteriormente es ejecutado por el servidor de WordPress y como resultado genera el código de la Figura A.20. que es el que realmente muestra nuestro

menú.

```
1 <ul class="nav navbar-nav">
2   <li id="nav-aboutus" class="menu-item menu-item-type-post_type menu-
3     item-object-page current-menu-item page_item page-item-43
4     current_page_item menu-item-79">
5     <a href="http://tungaloy.dev.culturaweb.com/wordpress/en/about-us/"
6       >About us</a>
7   </li>
8   <li id="nav-industries" class="menu-item menu-item-type-post_type
9     menu-item-object-page menu-item-78">
10    <a href="http://tungaloy.dev.culturaweb.com/wordpress/en/industries
11      /">Industries</a>
12  </li>
13  <li id="nav-products" class="menu-item menu-item-type-post_type menu-
14    item-object-page menu-item-77">
15    <a href="http://tungaloy.dev.culturaweb.com/wordpress/en/products/"
16      >Products</a>
17  </li>
18  <li id="nav-worldwidenetwork" class="menu-item menu-item-type-
19    post_type menu-item-object-page menu-item-76">
20    <a href="http://tungaloy.dev.culturaweb.com/wordpress/en/worldwide-
    network/">Worldwide Network</a>
  </li>
  <li id="nav-publications" class="menu-item menu-item-type-post_type
    menu-item-object-page menu-item-75">
    <a href="http://tungaloy.dev.culturaweb.com/wordpress/en/
    publications/">Publications</a>
  </li>
  <li id="nav-contact" class="menu-item menu-item-type-post_type menu-
    item-object-page menu-item-74">
    <a href="http://tungaloy.dev.culturaweb.com/wordpress/en/contact/">
    Contact</a>
  </li>
</ul>
```

Figura A.20: HTML generado por WordPress

Anexo B

Estructura de ficheros

En este anexo mostramos la estructura de ficheros del proyecto. Para simplificar partimos de capturas de la estructura de ficheros de la plantilla vacía que tomamos como base para desarrollar el proyecto. Esto nos será útil como documentación, para documentar dónde se encuentra cada fichero y qué contiene, y para diferenciar el trabajo previo de las aportaciones propias.

En la Figura B.1 encontramos el árbol de archivos del tema del proyecto, primero explicamos el contenido de cada fichero, después pasamos al contenido de las carpetas.

- *functions.php*[44]: el fichero se utiliza para añadir características y funcionalidad, para llamar a funciones de PHP o WordPress y definir funciones propias. El fichero de nuestra plantilla ya incluye todas las dependencias e inicializaciones para que el tema funcione correctamente.
- *404.php*: página de error HTTP 404 o página no encontrada, contiene el HTML que se mostrará como página de error cada vez que no se encuentre un contenido. En nuestra plantilla contiene un mensaje de error simple y sin formato.
- *archive.php*: este documento es la plantilla que muestra el archivo de la web y que lista todos los posts. En nuestra plantilla el fichero consulta todos los *posts* de la base de datos y los muestra en formato plano.
- *category.php*: muestra todos los *posts* de una categoría. El fichero de nuestra plantilla muestra todos los *posts* de una categoría dada en formato plano.
- *comments.php*: contiene las funciones para hacer comentarios de un *post*. El fichero de la plantilla contiene código que muestra todos los comentarios de un *post*.
- *footer.php*: pie común del tema, se muestra al final de todas las páginas, en este fichero se cierra el código HTML. En el fichero de nuestra plantilla únicamente contiene las etiquetas de cerrar el HTML.
- *header.php*: es la cabecera común a todas las páginas del tema, el contenido de este fichero se muestra en todas las páginas, también contiene las etiquetas que dan información a la

web, el enlace a la hoja de estilos, la apertura del *body*, etc. En la plantilla contiene importaciones, dependencias y un menú simple que se muestra a modo de lista.

- *index.php*: este archivo es el que WordPress asignará como página inicial. En nuestra plantilla contiene un título y una llamada al archivo *loop.php* que mostrará los últimos *posts*, todo en formato plano.
- *loop.php*: el *loop* de WordPress es el que muestra el contenido que se ve en cualquier página del sitio web, lee de la base de datos el contenido de una página y lo muestra. El código de nuestra plantilla muestra todos los *posts* del sistema.
- *page.php*: contiene lo necesario para mostrar una página completa, hace llamadas a los ficheros *header.php*, *sidebar.php*, *footer.php*, etc. En nuestra plantilla se hacen todas las importaciones típicas de otros ficheros.
- *pagination.php*: archivo donde se define la habilidad de separar listas de *posts* en varias páginas. En la plantilla se llama a una función de paginación ya creada en el archivo *function.php*.
- *screenshot.png*: es simplemente una imagen del logo de la empresa.
- *search.php*: con esta plantilla se muestran los resultados de las búsquedas realizadas. En nuestra plantilla encontramos código que muestra los resultados de una consulta en la base de datos en formato plano.
- *searchform.php*: en este archivo se define el formulario de búsqueda. El código de nuestra plantilla muestra una caja para introducir texto y un botón simple.
- *sidebar.php*: contiene una barra lateral para la web donde se suelen cargar *widgets*, enlaces de interés o cualquier cosa que el usuario quiera añadir. En nuestra plantilla el fichero contiene una barra lateral compuesta por la importación de dos *widgets* que habría que configurar desde el panel de administración de WordPress.
- *single.php*: este archivo se utiliza para mostrar los *posts* individualmente, es el que se abriría cuando desde un listado de *posts* hiciéramos clic sobre *leer más*. En la plantilla contiene código para mostrar el título de un *post*, la fecha, el contenido y sus categorías, todo se muestra en un formato plano de texto.
- *style.css*: contiene los estilos del tema, determina el aspecto del tema. En la plantilla encontramos este archivo vacío.
- *tag.php*: fichero para mostrar el archivo de etiquetas (taxonomías) para *posts*. En nuestro fichero se llama al *loop*.
- *template-xxx.php*: los ficheros que empiezan con el prefijo *template-* contienen plantillas que se le pueden asignar a cada página. Son ficheros que distribuyen la información de formas distintas y realizan llamadas a los ficheros explicados anteriormente en distintas partes de su código para mostrar información.

La carpeta *assets* que mostramos en la Figura B.2 puede contener una serie arbitraria de ficheros o carpetas para ser utilizados por la aplicación. A partir del nombre de todas sus subcarpetas podemos intuir su contenido. La carpeta *flags* contiene miniaturas de banderas

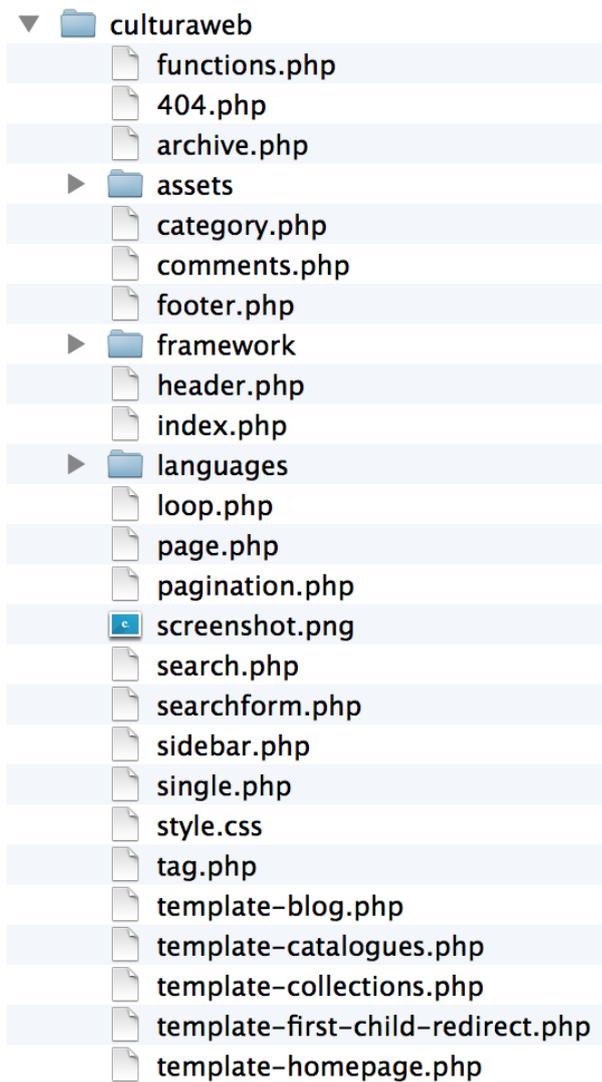


Figura B.1: Jerarquía de ficheros completa

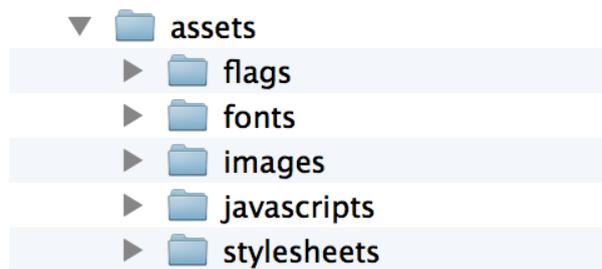


Figura B.2: Jerarquía de la carpeta *assets*

para mostrar en la selección de idiomas, la carpeta *fonts* los archivos de las fuentes que se utilizan en la web, *images* es el lugar donde se almacenan todas las imágenes, *jascripts* para almacenar este tipo de ficheros y *stylesheets* para hojas de estilo. En la plantilla que utilizamos todas estas carpetas se encuentran vacías listas para ser llenadas de contenido.

En la Figura B.3 observamos el contenido de la carpeta *frameworks*. Explicamos su contenido:

- Carpeta *customs*: destinada a incluir todos nuestros *custom post types* y taxonomías. La plantilla ya contiene algunos *posts* personalizados y un fichero para incluir la definición de taxonomías.
- *libraries.php*: para especificar la localización de recursos. En la plantilla contiene una función para cargar recursos de la carpeta *assets*.
- Carpeta *metabox*: contiene archivos para la construcción de las *metaboxes* del panel de administración de WordPress.
- *shortcodes.php*: este archivo es donde programamos todos los *shortcodes* del tema. En nuestra plantilla contiene algunos *shortcodes* básicos para tareas como la incrustación de vídeos en páginas o la creación de columnas.
- *slider.php*: es un *post type* para la creación de un *slider*.
- Carpeta *tinymce*: esta carpeta contiene imágenes en miniatura para crear atajos para los *shortcodes*.

Por último la carpeta *languages* (Figura B.4) contiene archivos de traducción. En la plantilla observamos dos archivos para la traducción al español. El archivo *.po* es un archivo que nosotros escribimos manualmente para la traducción de cadenas, y el archivo *.mo* es una compilación del otro archivo, es el que utiliza el sistema para realizar las traducciones.

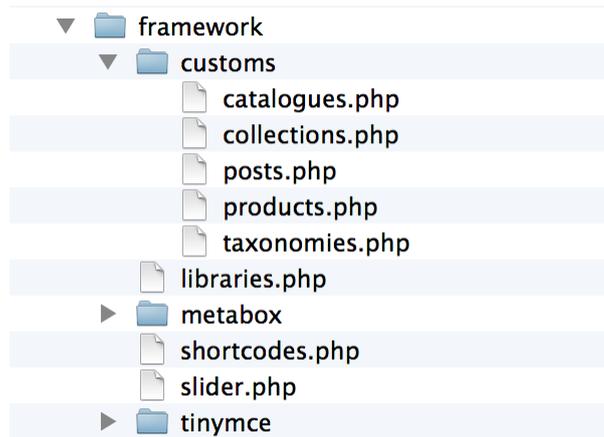


Figura B.3: Jerarquía de la carpeta *framework*

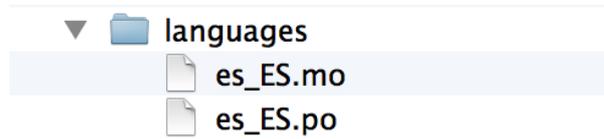


Figura B.4: Jerarquía de la carpeta *languages*

Anexo C

Diseño proporcionado por el cliente

En este anexo incluimos una impresión de las páginas principales del diseño proporcionado por el cliente. Mostramos las siguientes páginas:

1. *Home*
2. *About Us*
3. *Industries*
4. *Products*
5. *Producto DooFeed Series*
6. *Worldwide Network*
7. *Publications*
8. *Contact*



Japanese

TERMS AND CONDITIONS



ABOUT US INDUSTRIES PRODUCTS WORLDWIDE NETWORK PUBLICATIONS CONTACT



MILL LINE TUNGSLLOT

Incredibly secure system for slot-milling with excellent chip control
Watch it working at [Tung-TV](#) channel

Full Product Line

"click here and discover our product lines"

MillLine TurnLine DrillLine ToolLine



"What's New"

09 Dec. 2013
Launching of "External Turning Section" in Tung-Navi

06 Sep. 2013
New series of Tung-TV (Tungaloy Technical Vision) "Ecoturn" is upload on Youtube!

01 Sep 2013
The New Promotion Video "Tung-TV" series

[more](#)

"Press Release"

09 Dec. 2013
Launching of "External Turning Section" in Tung-Navi

06 Sep. 2013
New series of Tung-TV (Tungaloy Technical Vision) "Ecoturn" is upload on Youtube!

01 Sep 2013
The New Promotion Video "Tung-TV" series

[more](#)



[PRODUCTS](#)

[INDUSTRIES](#)

[EXHIBITIONS](#)

[CORPORATE PROFILE](#)

[WORLDWIDE NETWORK](#)

[CONTACT](#)

[PRIVACY POLICY](#)

[TERMS OF USE](#)

[SITE MAP](#)

WHAT'S NEW

09 Dec. 2013

Launching of "External Turning Section" in Tung-Navi

06 Sep. 2013

New series of Tung-TV (Tungaloy Technical Vision), "Ecoturn" is upload on Youtube!

01 Sep. 2013

The New Promotion Video "Tung-TV" series

PRESS RELEASES

02 Dec. 2013

Drill Line
Tungaloy Offers the Innovative High Performance Tool "DrillMeister"

18 Nov. 2013

Mill Line
Tungaloy's Popular Lines are Now More Flexible to Use with the "TungFlex" series

06 Nov. 2013

Drill Line
Tungaloy "GunDrill" Line Standardised

SUPPORT

11-1 Yoshima-Kogyodanchi

Iwaki 970-1144 Japan

Phone: +81 246-36-8501

Fax: +81 246-36-8542

PRODUCT SUPPORT

support@tungaloy.co.jp

+81 246-36-8501

ORDER SUPPORT

order@tungaloy.co.jp

+81 246-36-8501

FOLLOW US



© 1999 - 2013 TUNGALOY CORPORATION ALL RIGHTS RESERVED



COMPANY

“Overview”



Satoshi Kinoshita
President & CEO
Tungaloy Corporation



Jacob Harpaz
Chairman of Tungaloy Corporation
President of IMC Group



Warren Buffett
Chairman, CEO and largest
shareholder of Berkshire Hathaway,
the owner of the IMC Group

Tungaloy: A Member of the IMC Group

In August 2008, Tungaloy became part of the International Metalworking Companies (IMC) Group, owned by Warren Buffett's Berkshire Hathaway Inc. IMC acquired Tungaloy to lead the Japanese market and become a major international supplier of advanced cutting tools.

To achieve these goals, Tungaloy is investing in the latest machinery and automation in its production facilities and strengthening its R&D capabilities.

At Tungaloy, we provide our customers products that serve as industry benchmarks. Our goal is to improve productivity and cycle times to help reduce the overall costs and increase profitability.

OUR “History”



Bringing solutions since 1930.

The two predecessor companies, Shibaura Engineering Works Co., Ltd and Tokyo Electric Company, succeeded in developing a carbide alloy for the first time in Japan. The product was sold on the market as “Tungaloy”.

Building up on a successful Past for the bright Future.

Since its inception in the 1930's when it was Japan's first company to develop a carbide alloy, Tungaloy has a long history of excellence in the development of materials for metal cutting tools and innovative industrial products.

The diverse range of products, including metal-cutting tools, such as indexable inserts and steel products, wear resistant tools, civil engineering tools, and friction materials, are all based upon Tungaloy's expertise in powder metallurgy.

THINK “Green”



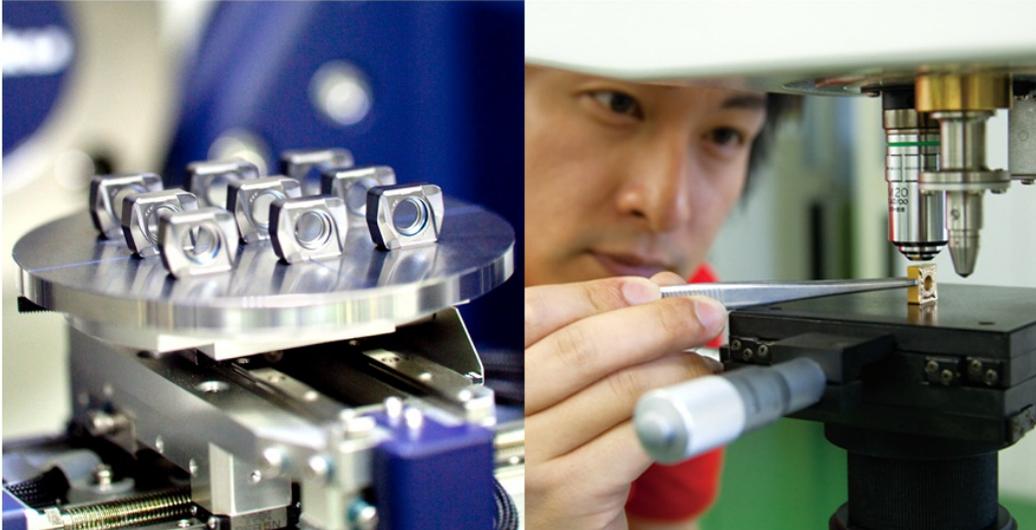
In 1997, Tungaloy acquired ISO: 14001 certification. This was an industry first for the carbide cutting tool manufacturing industry in Japan. The implementation of the environmental standard demonstrates our commitment to a sustainable future and we continually strive to reduce our carbon footprint to enhance our environmental credentials.



[click here to read about our think green](#)



RESEARCH — “& Development” —



Since 1929, Tungaloy has been the pioneer in materials research and the development of cemented carbides.

Tungaloy develops cutting-edge technologies by taking advantage of the latest techniques and equipment as well as a close relationship with customers worldwide in all industry sectors.

As the market leader in the development of powder metallurgy and innovative coating technology for high quality cutting tools, Tungaloy staff have the experience, expertise, and the latest equipment to ensure they remain the benchmark in the industry.

EVENTS — “Exhibitions&Seminars” —

21, 24 May 2014
Metaltech - Malaysia
(Kuala Lumpur)

19, 23 June 2014
ACMEE- India (Chennai)

8, 13 September 2014
IMTS- USA (Chicago)

16, 20 September 2014
AMB- Germany (Stuttgart)

8, 13 September 2014
IMTS- USA (Chicago)

16, 20 September 2014
AMB- Germany (Stuttgart)

19, 23 June 2014
ACMEE- India (Chennai)

21, 24 May 2014
Metaltech - Malaysia
(Kuala Lumpur)



CAREER

“Opportunities”

Would you like have a nice future in a global company and take part of our success?

Tungaloy always is hunting talented people to be part of our professional team, if you have the skills and the talent, we want know about you because we have the knowledge and together we can get a brilliant future.



Choose your country and explore our career opportunities

Choose your country ▾

search



Share

Like

Follow @TUNGALOYJAPAN



PRODUCTS

INDUSTRIES

EXHIBITIONS

CORPORATE PROFILE

WORLDWIDE NETWORK

CONTACT

PRIVACY POLICY

TERMS OF USE

SITE MAP

WHAT'S NEW

09 Dec. 2013

Launching of "External Turning Section" in Tung-Nav

06 Sep. 2013

New series of Tung-TV (Tungaloy Technical Vision), "Ecoturn" is upload on Youtube!

01 Sep. 2013

The New Promotion Video "Tung-TV" series

PRESS RELEASES

02 Dec. 2013

Drill Line

Tungaloy Offers the Innovative High Performance Tool "DrillMeister"

18 Nov. 2013

Mill Line

Tungaloy's Popular Lines are Now More Flexible to Use with the "TungFlex" series

06 Nov. 2013

Drill Line

Tungaloy "GunDrill" Line Standardised

SUPPORT

11-1 Yoshima-Kogyodanchi

Iwaki 970-1144 Japan

Phone: +81 246-36-8501

Fax: +81 246-36-8542

PRODUCT SUPPORT

support@tungaloy.co.jp

+81 246-36-8501

ORDER SUPPORT

order@tungaloy.co.jp

+81 246-36-8501

FOLLOW US



© 1999 - 2013 TUNGALOY CORPORATION ALL RIGHTS RESERVED

COMPLETE “Industrial Coverage”

Every customer has specific needs. We study them in depth and provide the best solution.

AUTOMOTIVE



Tungaloy's engineering teams and factories are located near the major car plants to provide automotive clients with creative solutions that improve their production processes.

HEAVY INDUSTRIES & POWER GENERATION



The high versatility, quality, and robustness of Tungaloy tools allow the power generation industry to take giant strides ahead.

AEROSPACE



Tungaloy is well-known for its high quality carbide grades, which are most suitable for machining exotic materials and hardened steels.

OIL & GAS



Tungaloy offers a wide range of powerful tools for the 3 main segments of the oil and gas industry: upstream, midstream, and downstream.

MACHINE TOOL



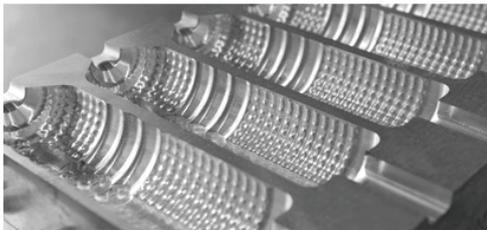
Tungaloy works with its customers in this industry, in a comprehensive technological partnership that leads to creative solutions.

MINIATURE PARTS



Tungaloy provides a wide variety of products suitable for swiss type machines, dedicated for small parts. This results in an efficient machining process and reduced cost.

DIE & MOULD



Tungaloy specializes in creating unique technological solutions, specifically for the die and mould industry.

BEARING



Tungaloy supplies a wide selection of CBN inserts for hard turning operations required in the bearing industry.



[PRODUCTS](#)

[INDUSTRIES](#)

[EXHIBITIONS](#)

[CORPORATE PROFILE](#)

[WORLDWIDE NETWORK](#)

[CONTACT](#)

[PRIVACY POLICY](#)

[TERMS OF USE](#)

[SITE MAP](#)

WHAT'S NEW

09 Dec. 2013

Launching of "External Turning Section" in Tung-Navii

06 Sep. 2013

New series of Tung-TV (Tungaloy Technical Vision), "Ecoturn" is upload on Youtube!

01 Sep. 2013

The New Promotion Video "Tung-TV" series

PRESS RELEASES

02 Dec. 2013

Drill Line
Tungaloy Offers the Innovative High Performance Tool "DrillMeister"

18 Nov. 2013

Mill Line
Tungaloy's Popular Lines are Now More Flexible to Use with the "TungFlex" series

06 Nov. 2013

Drill Line
Tungaloy "GunDrill" Line Standardised

SUPPORT

11-1 Yoshima-Kogyodanchi

Iwaki 970-1144 Japan

Phone: +81 246-36-8501

Fax: +81 246-36-8542

PRODUCT SUPPORT

support@tungaloy.co.jp

+81 246-36-8501

ORDER SUPPORT

order@tungaloy.co.jp

+81 246-36-8501

FOLLOW US



© 1999 - 2013 TUNGALOY CORPORATION ALL RIGHTS RESERVED



Japanese

TERMS AND CONDITIONS



ABOUT US INDUSTRIES PRODUCTS WORLDWIDE NETWORK PUBLICATIONS CONTACT

CUTTING TOOLS INDUSTRIAL PRODUCTS

with innovative, interchangeable head

read more

CUTTING “Tools”

The application of cutting tools is varied and can generally be categorized into turning, milling and drilling. Tungaloy has an extensive line-up of TAC inserts with a combination of various grades, dimensions and chipbreakers to suit the diverse demands of the marketplace.

MillLine TurnLine DrillLine ToolLine



“Can I help you?”

e-Catalog

Search for Tools

TUNG-Navi

Tungaloy Tool Navigator

MillLine

[Back to Top](#)

Indexable Mills

High Feed Milling

DoFeed series

DoFeedQuad

EXP and TXP

Face Milling

Dopent

DoOcto, DoQuad

Taw, EAWTPW, EPW

Shoulder Milling

Tung-AluMill

Tung-Tri

RoughingMill series

TungQuad

TungRec

TecMill

DoRec

T/EFE12, DPD09 andEDPD09 types

T/EPS11, 17 types

Profile Milling (Round Insert)

FixRMill

RoundSplit

Slot Milling

SlotMill series

TecSlot

Multi Function Milling Tools

HybridTacMill

EVX

Head Changeable EndMill

TungMeister

Inserts Grades for Milling

BX910

T1115

AH130

T3130 series

DS1000 series

NS740



Tungaloy's Milling tools provide our customers with original and innovative technology-based products that meet the high productivity and high efficiency demands of industry. We can offer a variety of milling tools for an extremely diverse range of applications on all material types.



Insert Converter



Digital Publications



Tungaloy Technical Vision

“contact
US”

TurnLine

[Back to Top](#)

Turning Tools

General Turning

[DominiTurn](#)
[EcoTurn](#)
[DimpleFx](#)
[PSF & Pss](#)
[HMM & AH905](#)
[Y-Pro](#)
[DominiTurn](#)
[EcoTurn](#)
[TurningA](#)
[StreamJet Bar](#)
[AL chipbreaker, T-DIA](#)



Tungaloy's Turning tools primarily provide our customers with the high productivity and high efficiency that can be expected from an industry leading manufacturer. The wide range of turning products is available for a variety of applications with extremely high quality guaranteed. Tungaloy continually invests in innovative production technology and staff development to create groundbreaking new products that continually surpasses our customer expectations.

Heavy Turning

[TurnTec](#)
[THS & TUS](#)
[TurnFeed](#)

Grooving & Parting

[EasyCut](#)
[TungCut](#)
[TetraCut](#)
[GTGN type](#)
[MY-T series](#)
[TGTS, TGTT % SFT-type series](#)

Threading

[TungThread](#)

Small Part Machining

[TinyTurn](#)
[JS, JRP, JPP, JSP](#)
[J-Series](#)

Inserts Grades for Turning

Carbide Grades

[NS9530 cermet, GT9530 coated cermet](#)
[T9100 series](#)
[T6100 series](#)
[AH600 series](#)
[AH905](#)
[T5100 series](#)
[AH725](#)
[GT/NS700 series](#)

CBN Grades

[Mini T-CBN](#)
[BX910](#)
[BXM series](#)

DrillLine

[Back to Top](#)

Indexable Drills

DrillMeister
TungSix-Drill
TungDrill Big
TungDrill Twisted

Solid Carbide Drills

DSM type

Deep Hole Drilling

GunDrill



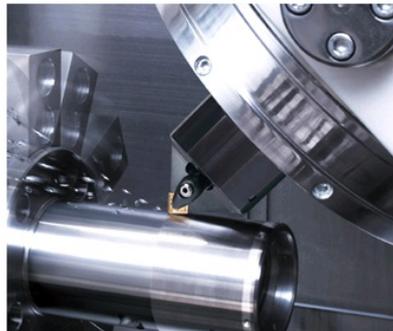
Tungaloy's Holemaking tools ensure our customers can cost effectively, efficiently and productively produce holes to impeccable quality levels with remarkable accuracy. The wide range of drills include solid carbide drills, indexable insert drills, gun drills and reamers that are all available in a vast array of products and size variations to guarantee holemaking with the utmost in quality.

ToolLine

[Back to Top](#)

Tooling systems

TungCap
TungHold BT interface series
TungHold



Tungaloy's Holemaking tools ensure our customers can cost effectively, efficiently and productively produce holes to impeccable quality levels with remarkable accuracy. The wide range of drills include solid carbide drills, indexable insert drills, gun drills and reamers that are all available in a vast array of products and size variations to guarantee holemaking with the utmost in quality.

INDUSTRIAL

“Products”

The application of cutting tools is varied and can generally be categorized into turning, milling and drilling. Tungaloy has an extensive line-up of TAC inserts with a combination of various grades, dimensions and chipbreakers to suit the diverse demands of the marketplace.

Industrial Products

Friction Materials

Wear Resistant Tools

PCB Drills

Civil Engineering and Building Construct Tools



[Back to Top](#)

[Share](#)

[Like](#)

[Follow @TUNGALOYJAPAN](#)



[PRODUCTS](#)

[INDUSTRIES](#)

[EXHIBITIONS](#)

[CORPORATE PROFILE](#)

[WORLDWIDE NETWORK](#)

[CONTACT](#)

[PRIVACY POLICY](#)

[TERMS OF USE](#)

[SITE MAP](#)

WHAT'S NEW

09 Dec. 2013
Launching of "External Turning Section" in Tung-Navii

06 Sep. 2013
New series of Tung-TV (Tungaloy Technical Vision), "Ecoturn" is upload on Youtube!

01 Sep. 2013
The New Promotion Video "Tung-TV" series

PRESS RELEASES

02 Dec. 2013
Drill Line
Tungaloy Offers the Innovative High Performance Tool "DrillMeister"

18 Nov. 2013
Mill Line
Tungaloy's Popular Lines are Now More Flexible to Use with the "TungFlex" series

06 Nov. 2013
Drill Line
Tungaloy "GunDrill" Line Standardised

SUPPORT

11-1 Yoshima-Kogyodanchi
Iwaki 970-1144 Japan
Phone: +81 246-36-8501
Fax: +81 246-36-8542

PRODUCT SUPPORT

support@tungaloy.co.jp
+81 246-36-8501

ORDER SUPPORT

order@tungaloy.co.jp
+81 246-36-8501

FOLLOW US



© 1999 - 2013 TUNGALOY CORPORATION ALL RIGHTS RESERVED



Japanese

TERMS AND CONDITIONS



ABOUT US INDUSTRIES PRODUCTS WORLDWIDE NETWORK PUBLICATIONS CONTACT

CUTTING TOOLS INDUSTRIAL PRODUCTS

MillLine DOFEED SERIES TUNGALOY

New generation of high feed cutters
offering incredible productivity

- Excellent chip evacuation: Holes for air blast to reduce edge chipping caused by re-cutting chips
- Remarkable chip control: Large inclination improves chip flow
- High density inserts for improved productivity
- Reduce chattering with low cutting forces

- DoFeed** [download PDF](#) - [View online](#)
- DoFeed Mini** [download PDF](#) - [View online](#)
- DoFeed Quad** [download PDF](#) - [View online](#)

“Can I help you?”

e-Catalog

Search for **Tools**

TUNG-Navi

Tungaloy
Tool Navigator



Insert Converter



Digital Publications



Tungaloy
Technical Vision

“contact
US”





[PRODUCTS](#)

[INDUSTRIES](#)

[EXHIBITIONS](#)

[CORPORATE PROFILE](#)

[WORLDWIDE NETWORK](#)

[CONTACT](#)

[PRIVACY POLICY](#)

[TERMS OF USE](#)

[SITE MAP](#)

WHAT'S NEW

09 Dec. 2013

Launching of "External Turning Section" in Tung-Navii

06 Sep. 2013

New series of Tung-TV (Tungaloy Technical Vision), "Ecoturn" is upload on Youtube!

01 Sep. 2013

The New Promotion Video "Tung-TV" series

PRESS RELEASES

02 Dec. 2013

Drill Line
Tungaloy Offers the Innovative High Performance Tool "DrillMeister"

18 Nov. 2013

Mill Line
Tungaloy's Popular Lines are Now More Flexible to Use with the "TungFlex" series

06 Nov. 2013

Drill Line
Tungaloy "GunDrill" Line Standardised

SUPPORT

11-1 Yoshima-Kogyodanchi
Iwaki 970-1144 Japan
Phone: +81 246-36-8501
Fax: +81 246-36-8542

PRODUCT SUPPORT

support@tungaloy.co.jp
+81 246-36-8501

ORDER SUPPORT

order@tungaloy.co.jp
+81 246-36-8501

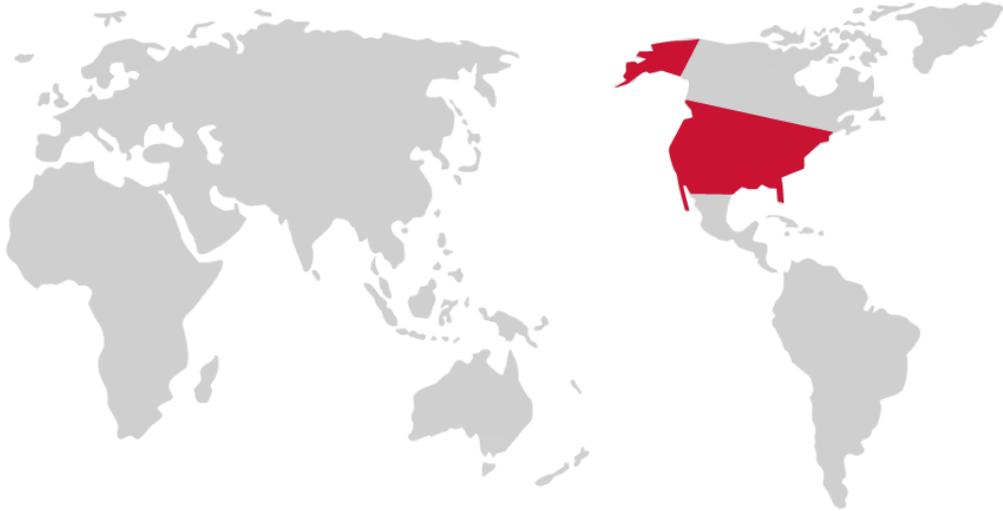
FOLLOW US



© 1999 - 2013 TUNGALOY CORPORATION ALL RIGHTS RESERVED



WORLDWIDE “Network”



Choose your country or state ▾

State (only for US)

Tungaloy Americ Inc.

[View Site](#)

3726 N Ventura Drive
Arlington Heights
IL 60004, U.S.A.
Phone: +1-888-554-8394
Fax: +1-888-554-8392

Distributors on Florida:

Alternatives for Industry Inc

3726 N Ventura Drive
Arlington Heights
IL 60004, U.S.A.
Phone: +1-888-554-8394
Fax: +1-888-554-8392

Miller Bearings

3726 N Ventura Drive
Arlington Heights
IL 60004, U.S.A.
Phone: +1-888-554-8394
Fax: +1-888-554-8392

Dewil Tool Co Incorporated

3726 N Ventura Drive
Arlington Heights
IL 60004, U.S.A.
Phone: +1-888-554-8394
Fax: +1-888-554-8392



- PRODUCTS
- INDUSTRIES
- EXHIBITIONS
- CORPORATE PROFILE
- WORLDWIDE NETWORK
- CONTACT
- PRIVACY POLICY
- TERMS OF USE
- SITE MAP

WHAT'S NEW

09 Dec. 2013
Launching of “External Turning Section” in Tung-Navi

06 Sep. 2013
New series of Tung-TV (Tungaloy Technical Vision), “Ecoturn” is upload on Youtube!

01 Sep. 2013
The New Promotion Video “Tung-TV” series

PRESS RELEASES

02 Dec. 2013
Drill Line
Tungaloy Offers the Innovative High Performance Tool “DrillMeister”

18 Nov. 2013
Mill Line
Tungaloy’s Popular Lines are Now More Flexible to Use with the “TungFlex” series

06 Nov. 2013
Drill Line
Tungaloy “GunDrill” Line Standardised

SUPPORT

11-1 Yoshima-Kogyodanchi
Iwaki 970-1144 Japan
Phone: +81 246-36-8501
Fax: +81 246-36-8542

PRODUCT SUPPORT

support@tungaloy.co.jp
+81 246-36-8501

ORDER SUPPORT

order@tungaloy.co.jp
+81 246-36-8501



PUBLICATIONS

“Reports & Brochures”

Tungaloy Reports Brochures

▶ MillLine

▶ TurnLine

▼ DrillLine

Indexable Drills

- DrillMeister** - Download PDF - View Online
- TungSix-Drill** - Download PDF - View Online
- TungDrill Big** - Download PDF - View Online
- TungDrill Twisted** - Download PDF - View Online

Solid Carbide Drills

- DSM type** - Download PDF - View Online

Deep Hole Drilling

- GunDrill** - Download PDF - View Online

▶ ToolLine



- PRODUCTS
- INDUSTRIES
- EXHIBITIONS
- CORPORATE PROFILE
- WORLDWIDE NETWORK
- CONTACT
- PRIVACY POLICY
- TERMS OF USE
- SITE MAP

WHAT'S NEW

09 Dec. 2013
Launching of "External Turning Section" in Tung-Navi

06 Sep. 2013
New series of Tung-TV (Tungaloy Technical Vision), "Ecoturn" is upload on Youtube!

01 Sep. 2013
The New Promotion Video "Tung-TV" series

PRESS RELEASES

02 Dec. 2013
Drill Line
Tungaloy Offers the Innovative High Performance Tool "DrillMeister"

18 Nov. 2013
Mill Line
Tungaloy's Popular Lines are Now More Flexible to Use with the "TungFlex" series

06 Nov. 2013
Drill Line
Tungaloy "GunDrill" Line Standardised

SUPPORT

11-1 Yoshima-Kogyodanchi
Iwaki 970-1144 Japan
Phone: +81 246-36-8501
Fax: +81 246-36-8542

PRODUCT SUPPORT

support@tungaloy.co.jp
+81 246-36-8501

ORDER SUPPORT

order@tungaloy.co.jp
+81 246-36-8501



CONTACT

“Inquiry Form”

Company Name Country

Contact Person E-mail

Phone Fax

Destination Choose your Country

Message

11 - 1 Yoshima - Kogyodanchi, Iwaki 970-1144 Japan
Phone: +81 246-36-8501
Fax: +81 246-36-8542

Please be sure to fill in items marked



- PRODUCTS
- INDUSTRIES
- EXHIBITIONS
- CORPORATE PROFILE
- WORLDWIDE NETWORK
- CONTACT
- PRIVACY POLICY
- TERMS OF USE
- SITE MAP

WHAT'S NEW

09 Dec. 2013
Launching of “External Turning Section” in Tung-Navi

06 Sep. 2013
New series of Tung-TV (Tungaloy Technical Vision), “EcoTurn” is upload on Youtube!

01 Sep. 2013
The New Promotion Video “Tung-TV” series

PRESS RELEASES

02 Dec. 2013
Drill Line
Tungaloy Offers the Innovative High Performance Tool “DrillMeister”

18 Nov. 2013
Mill Line
Tungaloy’s Popular Lines are Now More Flexible to Use with the “TungFlex” series

06 Nov. 2013
Drill Line
Tungaloy “GunDrill” Line Standardised

SUPPORT

11-1 Yoshima-Kogyodanchi Iwaki 970-1144 Japan
Phone: +81 246-36-8501
Fax: +81 246-36-8542

PRODUCT SUPPORT

support@tungaloy.co.jp
+81 246-36-8501

ORDER SUPPORT

order@tungaloy.co.jp
+81 246-36-8501



Anexo D

Informes quincenales

En este anexo incluimos los informes quincenales desarrollados durante el proyecto, en estos se puede observar la evolución del proyecto y el trabajo desarrollado durante cada quincena de la estancia en prácticas.

El primer informe quincenal que se realizó consistió en la propuesta técnica, todo el contenido de la propuesta está ya incluido y extendido en este documento, por tanto no lo hemos incluido en el anexo.

Informe de la segunda quincena

Marina Pibernat Segarra

Del 01/07/2014 al 15/07/2014

1. Quincena anterior

La quincena anterior la dedicamos a la realización de la propuesta técnica analizando las características del proyecto y las necesidades del cliente. Una vez realizado el análisis y la planificación del proyecto empecé con la programación *front-end*. Empecé el desarrollo con HTML5, CSS3 junto a Less, Bootstrap, PHP y Javascript. Utilicé el IDE Aptana junto a Git y Bitbucket para el control de versiones.

2. Quincena actual

Durante esta quincena he terminado la programación *front-end* a falta de pequeños detalles que necesitan ser comentados con el cliente. Al acabar esta fase de programación front-end antes de lo previsto, he empezado con la programación *back-end*. Para la programación back-end he empezado a utilizar Wordpress y PHP.

Como el proyecto estaba bastante avanzado y la empresa necesitaba una aplicación sencilla para Android también he dedicado varios días al desarrollo de ésta *app* dejando de lado por unos días el proyecto principal. He utilizado el entorno de desarrollo Android Studio con Git para el control de versiones.

3. Quincena siguiente

Modificando las previsiones iniciales debido al adelanto del proyecto, en la quincena próxima está previsto terminar con la programación *back-end* y presentar un primer prototipo al cliente.

4. Observaciones

La planificación inicial no ha resultado ser demasiado acertada. Sobrestime el tiempo necesario para cada tarea, posiblemente debido a la inexperiencia y al desconocimiento inicial de las

tecnologías. Otro posible error en la planificación podría ser el intento de ajustar las tareas a las trescientas horas.

También tendría que haber tenido en cuenta el tiempo necesario para realizar modificaciones al proyecto después de una primera entrega al cliente.

Informe de la tercera quincena

Marina Pibernat Segarra

Del 15/07/2014 al 29/07/2014

1. Quincena anterior

La quincena anterior terminamos la programación *front-end* con la creación en HTML5, CSS3 junto a less, Bootstrap, PHP y Javascript de todas las páginas que incluía el diseño proporcionado por el cliente. También empezamos con la programación *back-end*.

2. Quincena actual

Durante esta quincena he terminado la programación *back-end* utilizando Wordpress y PHP. Ha consistido en la creación de una plantilla Wordpress a partir de la versión front-end que desarrollamos anteriormente y en la creación de *custom post types* y *shortcodes* para facilitar al futuro usuario la gestión del sitio web. He utilizado el IDE Aptana y FTP para almacenar los cambios en un servidor.

Una vez terminada la programación *back-end* he empezado a introducir datos del cliente a la web. Para ello hemos utilizado una carpeta compartida Dropbox dónde el cliente va subiendo toda la información que debemos introducir con una estructura acordada.

En esta quincena también he dedicado dos días al desarrollo de un sitio web sencillo a partir de una plantilla, y durante una tarde he dado una *master class* de Java a los miembros de la empresa para introducirlos al lenguaje.

3. Quincena siguiente

El proyecto está prácticamente acabado a falta de pequeños detalles que el cliente nos pida que modifiquemos. La introducción de datos será a lo que necesitaremos dedicar más tiempo. También nos encargaremos de revisar cada detalle para comprobar que todo funciona correctamente y que el código no necesita modificaciones.

4. Observaciones

Llegados a este punto del proyecto necesitamos mucha comunicación con el cliente. Esto hace que el desarrollo del proyecto no sea todo lo fluido que nos gustaría debido a la ligera dificultad para la comunicación. Muchas veces la comunicación escrita no es suficiente y la diferencia horaria entre España y Japón, junto con la dificultad para que dos empresas encuentren un momento para reunirse hacen que el proyecto se quede estancado algún que otro día.

Informe de la cuarta quincena

Marina Pibernat Segarra

Del 29/07/2014 al 12/08/2014

1. Quincena anterior

La quincena anterior la dedicamos al desarrollo de la mayor parte de la programación *back-end* y a la introducción de datos proporcionados por el cliente al sitio web.

2. Quincena actual

Durante esta quincena he realizado pequeñas modificaciones que el cliente nos ha ido pidiendo, tanto de de la parte *front-end* como *back-end*. Por ejemplo, el cliente ha pedido que añadamos al sitio un selector de idiomas: en la parte de *front-end*, hemos añadido el selector con el aspecto deseado por el cliente y en la parte de *back-end*, lo hemos integrado con los idiomas de *WordPress* para que los idiomas aparezcan en el selector automáticamente.

En esta quincena el cliente ha continuado pasándonos información para subir a la web, he subido toda la información que nos ha proporcionado creando algún pequeño *script* en *Python* para automatizar la subida de determinados datos a partir de hojas *XML*.

Como la cantidad de trabajo a hacer dependía mucho de la información que nos fuese pasando el cliente, ha habido días en los que el proyecto no podía avanzar. He invertido estos días en otras tareas como el desarrollo de un *minisite* para la promoción de un producto mediante un concurso.

3. Observaciones

Ésta ha sido la última quincena. Mirando la planificación inicial no hemos llegado a la última parte planificada de *puesta en marcha del proyecto*. Tampoco hemos tenido tiempo para realizar todas las modificaciones pedidas por el cliente. Como ya comentamos en un informe anterior, esto no fué incluido en la planificación, este tiempo necesario debería haberse tenido en cuenta, con la dificultad para la estimación debido a la gran dependencia del cliente en estas fases finales de modificaciones y subida de información.

Anexo E

Pruebas de aceptación

Id	Historia de usuario
HU1	Como administrador necesito añadir y eliminar enlaces del <i>header</i> .
HU2	Como administrador necesito añadir, modificar y eliminar elementos del <i>footer</i> .
HU3	Como administrador necesito crear, modificar y eliminar páginas.
HU4	Como administrador necesito crear pestañas.
HU5	Como administrador necesito crear acordeones.
HU6	Como administrador necesito crear formularios.
HU7	Como administrador necesito crear menús desplegables.
HU8	Como administrador necesito crear, modificar y eliminar <i>sliders</i> y añadirlos a una página.
HU9	Como administrador necesito crear, modificar o eliminar eventos que se muestren automáticamente en la sección de eventos.
HU10	Como administrador necesito crear, modificar o añadir productos y que se muestren automáticamente en la sección de productos.
HU11	Como administrador necesito añadir, modificar o eliminar distribuidores y subsidiarias que se muestren automáticamente en el mapa de distribuidores.
HU12	Como administrador necesito añadir, modificar o eliminar publicaciones que se muestren automáticamente en la sección de publicaciones.
HU13	Como administrador necesito añadir, modificar o eliminar destinatarios del formulario de contacto.
HU14	Como usuario necesito contactar con la empresa a través de un formulario que envíe un mensaje al destino seleccionado.
HU15	Como usuario necesito acceder a redes sociales de la empresa y compartir sus noticias en mis redes sociales.

Figura E.1: Historias de usuario

PRUEBAS DE ACEPTACIÓN HU1

1. **Añadir enlace:** al añadir un enlace, cuando no existe un enlace igual el sistema, el sistema añade el enlace y podemos verlo en el menú.

2. **Añadir enlace repetido:** al añadir un enlace, cuando ya existe un enlace igual en el sistema, el sistema añade el enlace y podemos verlo en el menú.
3. **Añadir muchos enlaces:** al añadir un enlace, cuando ya hay muchos enlaces y no caben más en la pantalla de la web, el sistema añade el enlace y lo muestra en una segunda línea del menú.
4. **Eliminar enlace:** al eliminar un enlace, el sistema elimina el enlace y ya no podemos verlo en el menú.
5. **Eliminar todos los enlaces:** al eliminar un enlace, cuando es el único enlace en el sistema, el sistema elimina el enlace y se muestra el menú vacío.

PRUEBAS DE ACEPTACIÓN HU2

1. **Añadir enlace:** al añadir un enlace al menú del *footer*, cuando no existe un enlace igual en el sistema, el sistema añade el enlace y podemos verlo en el menú.
2. **Añadir enlace repetido:** al añadir un enlace al menú del *footer*, cuando ya existe un enlace igual en el sistema, el sistema añade el enlace y podemos verlo en el menú.
3. **Eliminar enlace:** al eliminar un enlace del menú del *footer*, el sistema elimina el enlace y ya no podemos verlo en el menú.
4. **Eliminar todos los enlaces:** al eliminar un enlace del menú del *footer*, cuando es el único enlace en el sistema, el sistema elimina el enlace y se muestra el menú vacío.
5. **Añadir elemento:** al añadir un elemento en la parte derecha del *footer*, cuando no existe un elemento igual en el sistema, el sistema añade el elemento y podemos verlo en el *footer*.
6. **Añadir elemento repetido:** al añadir un elemento en la parte derecha del *footer*, cuando ya existe un elemento igual en el sistema, el sistema añade el elemento y podemos verlo en el *footer*.
7. **Eliminar elemento:** al eliminar un elemento de la parte derecha del *footer*, el sistema elimina el elemento y ya no podemos verlo en el *footer*.
8. **Eliminar todos los elementos:** al eliminar un elemento de la parte derecha del *footer*, cuando es el único elemento en el sistema, el sistema elimina el elemento y la zona se muestra vacía.

PRUEBAS DE ACEPTACIÓN HU3

La funcionalidad de esta historia de usuario la ofrece cualquier instalación de WordPress por tanto no necesitamos pruebas aceptación para comprobar su funcionamiento.

PRUEBAS DE ACEPTACIÓN HU4

1. **Añadir pestaña:** al añadir una pestaña a una página, cuando todavía no existe ninguna pestaña en el sistema, el sistema muestra una única pestaña en la que el usuario no puede hacer clic.

2. **Añadir dos pestañas:** al añadir una pestaña, cuando ya existe otra pestaña en el sistema, el sistema muestra dos pestañas en las que el usuario puede hacer clic.
3. **Añadir muchas pestañas:** al añadir una pestaña, cuando ya existen muchas pestañas en el sistema y no caben en la pantalla de la web, el sistema muestra la pestaña en una segunda línea de pestañas.
4. **Hacer clic sobre pestaña:** al hacer clic sobre una pestaña, cuando se está mostrando el contenido de otra pestaña, el sistema cambia el contenido mostrado.
5. **Crear más de un conjunto de pestañas:** al crear un conjunto de pestañas, cuando ya existe otro conjunto de pestañas en el sistema, las pestañas continúan cambiando su contenido al hacer clic y se comportan independientemente.

PRUEBAS DE ACEPTACIÓN HU5

1. **Añadir acordeón:** al añadir un acordeón a una página, cuando todavía no existe ningún acordeón en el sistema, el sistema muestra un único acordeón que se despliega y encoge cuando el usuario hace clic sobre él.
2. **Añadir dos acordeones:** al añadir un acordeón, cuando ya existe otro acordeón en el sistema, el sistema muestra dos acordeones que se despliegan y encogen cuando hacemos clic sobre uno de ellos.
3. **Hacer clic sobre acordeón:** al hacer clic sobre un acordeón, cuando otro acordeón está abierto, el sistema despliega el acordeón y encoge el resto de acordeones.
4. **Crear más de un conjunto de acordeones:** al crear un conjunto de acordeones, cuando ya existe otro conjunto de acordeones en el sistema, los acordeones se continúan encogiéndose y desplegando correctamente y se comportan independientemente.

PRUEBAS DE ACEPTACIÓN HU6

1. **Añadir formulario:** al añadir un formulario a una página, cuando todavía no existe ningún formulario igual en el sistema, el sistema muestra el formulario compuesto por cajas de introducción de texto y al menos un botón.
2. **Enviar datos correctos:** al pulsar sobre el botón para enviar los datos, cuando todos los datos introducidos en el formulario son correctos, el sistema envía los datos al destino especificado y muestra un mensaje indicando que los datos han sido enviados correctamente.
3. **Enviar datos incorrectos:** al pulsar sobre el botón para enviar los datos, cuando alguno de los datos introducidos en el formulario es incorrecto, el sistema no envía los datos y muestra un mensaje indicando que alguno de los datos introducidos no es correcto.
4. **Enviar datos incompletos obligatorios:** al pulsar sobre el botón para enviar los datos, cuando falta algún dato marcado como obligatorio por introducir en el formulario, el sistema no envía los datos y muestra un mensaje indicando que faltan datos requeridos por introducir.

5. **Enviar datos incompletos no obligatorios:** al pulsar sobre el botón para enviar los datos, cuando falta algún dato no marcado como obligatorio por introducir en el formulario, el sistema envía los datos y muestra un mensaje indicando que los datos han sido enviados correctamente.

PRUEBAS DE ACEPTACIÓN HU7

1. **Añadir menú desplegable:** al añadir un menú desplegable a una página, cuando todavía no existe ningún menú desplegable en el sistema, el sistema muestra el menú desplegable junto a un botón.
2. **Seleccionar elemento del desplegable:** al pulsar sobre el botón del desplegable, cuando hemos seleccionado alguna de sus opciones, el sistema nos redirige a la página establecida.
3. **Añadir más de un menú desplegable:** al añadir un menú desplegable a una página, cuando ya existe otro menú desplegable en el sistema, los desplegables continúan pasando las pruebas anteriores y se comportan independientemente.

PRUEBAS DE ACEPTACIÓN HU8

1. **Crear *slider*:** al crear un *slider*, cuando no existe un *slider* igual en el sistema, el sistema crea el *slider*, lo almacena y lo muestra en la lista de *sliders*.
2. **Crear *slider* con datos repetidos:** al crear un *slider*, cuando ya existe un *slider* con los mismos datos en el sistema, el sistema crea el *slider*, lo almacena y lo muestra en la lista de *sliders*.
3. **Modificar *slider*:** al modificar un *slider*, cuando este ya existía en el sistema, el sistema almacena los datos modificados y muestra el *slider* con los nuevos datos en la lista de *sliders* y en las páginas que lo utilicen.
4. **Eliminar *slider*:** al eliminar un *slider*, cuando este ya existe en el sistema, el sistema elimina los datos del *slider* y no lo muestra en la lista de *sliders* ni en las páginas que anteriormente lo utilizaran.
5. **Añadir *slider* a una página:** al añadir un *slider* a una página, cuando la página no tiene aún ningún *slider* asignado, el sistema asigna el *slider* a la página y lo muestra después del *header*.
6. **Añadir *slider* a una página con *slider*:** al añadir un *slider* a una página, cuando la página ya tiene un *slider* asignado, el sistema asigna el nuevo *slider* a la página, desvincula el *slider* anterior y muestra el nuevo *slider* después del *header*.
7. **Eliminar *slider* de una página:** al eliminar un *slider* de una página, cuando la página tiene un *slider* asignado, el sistema desvincula el *slider* de la página y ya no muestra el *slider* en la página pero continua mostrándolo en la lista de *sliders*.

PRUEBAS DE ACEPTACIÓN HU9

1. **Crear evento:** al crear un evento, cuando no existe un evento igual en el sistema, el sistema crea el evento, lo almacena y lo muestra en la lista de eventos del panel de administración.
2. **Crear evento con datos repetidos:** al crear un evento, cuando ya existe un evento con los mismos datos en el sistema, el sistema crea el evento, lo almacena y lo muestra en la lista de eventos del panel de administración.
3. **Crear evento pasado:** al crear un evento, cuando la fecha del evento es anterior a la actual, el sistema crea el evento, lo almacena y lo muestra en la lista de eventos del panel de administración pero no lo muestra en la lista pública de eventos.
4. **Crear evento futuro:** al crear un evento, cuando la fecha del evento es posterior a la actual, el sistema crea el evento, lo almacena y lo muestra en la lista de eventos del panel de administración y en la lista pública de eventos.
5. **Modificar evento:** al modificar un evento, cuando este ya existía en el sistema, el sistema almacena los datos modificados y muestra el evento con los nuevos datos en la lista de eventos del panel de administración y en la lista pública de eventos.
6. **Eliminar evento:** al eliminar un evento, cuando este ya existe en el sistema, el sistema elimina los datos del evento y no lo muestra en la lista de eventos del panel de administración ni en la lista pública de eventos.

PRUEBAS DE ACEPTACIÓN HU10

1. **Crear producto:** al crear un producto, cuando no existe un producto igual en el sistema, el sistema crea el producto, lo almacena y lo muestra en la lista de productos del panel de administración y en el lugar correcto de la lista pública de productos.
2. **Crear producto con datos repetidos:** al crear un producto, cuando ya existe un producto con los mismos datos en el sistema, el sistema crea el producto, lo almacena y lo muestra en la lista de productos del panel de administración y en la lista pública de productos.
3. **Modificar producto:** al modificar un producto, cuando este ya existe en el sistema, el sistema almacena los datos modificados y muestra el producto con los nuevos datos en la lista de productos del panel de administración y en la lista pública de productos.
4. **Eliminar producto:** al eliminar un producto, cuando este ya existe en el sistema, el sistema elimina los datos del producto y no lo muestra en la lista de productos del panel de administración ni en la lista pública de productos.

PRUEBAS DE ACEPTACIÓN HU11

1. **Crear distribuidor:** al crear un distribuidor, cuando no existe un distribuidor igual en el sistema, el sistema crea el distribuidor, lo almacena y lo muestra en la lista de

distribuidores del panel de administración y en el lugar correcto de la lista pública de distribuidores.

2. **Crear distribuidor con datos repetidos:** al crear un distribuidor, cuando ya existe un distribuidor con los mismos datos en el sistema, el sistema crea el distribuidor, lo almacena y lo muestra en la lista de distribuidores del panel de administración y en la lista pública de distribuidores.
3. **Modificar distribuidor:** al modificar un distribuidor, cuando este ya existe en el sistema, el sistema almacena los datos modificados y muestra el distribuidor con los nuevos datos en la lista de distribuidores del panel de administración y en la lista pública de distribuidores.
4. **Eliminar distribuidor:** al eliminar un distribuidor, cuando este ya existe en el sistema, el sistema elimina los datos del distribuidor y no lo muestra en la lista de distribuidores del panel de administración ni en la lista pública de distribuidores.

PRUEBAS DE ACEPTACIÓN HU12

1. **Crear publicación:** al crear una publicación, cuando no existe una publicación igual en el sistema, el sistema crea la publicación, la almacena y la muestra en la lista de publicaciones del panel de administración y en el lugar correcto de la lista pública de publicaciones.
2. **Crear publicación con datos repetidos:** al crear una publicación, cuando ya existe una publicación con los mismos datos en el sistema, el sistema crea la publicación, la almacena y la muestra en la lista de productos del panel de administración y en la lista pública de publicaciones.
3. **Modificar publicación:** al modificar una publicación, cuando este ya existe en el sistema, el sistema almacena los datos modificados y muestra la publicación con los nuevos datos en la lista de publicaciones del panel de administración y en la lista pública de publicaciones.
4. **Eliminar publicación:** al eliminar una publicación, cuando este ya existe en el sistema, el sistema elimina los datos de la publicación y no la muestra en la lista de publicaciones del panel de administración ni en la lista pública de publicaciones.

PRUEBAS DE ACEPTACIÓN HU13

1. **Añadir destinatario:** al añadir un destinatario a un formulario, cuando se envían datos desde el formulario, el sistema envía los datos al destinatario añadido.
2. **Modificar destinatario:** al modificar un destinatario de un formulario, cuando se envían datos desde el formulario, el sistema envía los datos al nuevo destinatario añadido.
3. **Eliminar destinatario:** al eliminar un destinatario de un formulario, cuando se envían datos desde el formulario, el sistema no envía los datos al destinatario eliminado.

PRUEBAS DE ACEPTACIÓN HU14

La funcionalidad de esta historia de usuario ya la hemos probado con los tests de aceptación de la *HU6 Como administrador necesito crear formularios*.

PRUEBAS DE ACEPTACIÓN HU15

1. **Botones *header*:** al hacer clic sobre los botones de redes sociales situados en el *header*, el sistema redirige la página a la red social correspondiente de la empresa.
2. **Botones *footer*:** al hacer clic sobre los botones de redes sociales situados en el *footer*, el sistema redirige la página a nuestra cuenta de la red social para seguir o compartir.