OPENRS
(OPEN Real Estate Software)

Ángel Luis Berasuain Ruiz, [(1)]Daniel Molina Cabrera

Calle Pinito del Oro, nº 4, 6ª C, 04009, Almería, España, Teléfono: 659163979, angel.berasuain@gmail.com

[(1)]Escuela Superior de Ingeniería, Avenida de la Universidad, 10, 11519, Puerto Real (Cádiz)

Este proyecto ofrece una alternativa a las herramientas existentes para la gestión de un pequeño negocio inmobiliario. Permite agrupar en un solo software la capacidad de la empresa de gestionar la actividad de su negocio así como de promocionarlo a través de una web corporativa configurable y multilenguaje desde un enfoque totalmente comercial pero siendo una herramienta libre. Además, todas las interfaces son responsivas, por lo que se adaptará a la mayoría de dispositivos existentes.

Palabras Clave: *Inmobiliario, CMS, OPEN, Real Estate, Gestión.*

1. Introducción

1.1. Motivación

En 2013 al autor del proyecto creó un software para la empresa GESTICADIZ (OPENINMOCMS) orientado a la generación de documentación y que satisfacía las necesidades particulares de aquel momento. Con el paso de los años, este software ha quedado obsoleto tanto en tecnologías como en funcionalidades y necesita renovarse.

Actualmente las inmobiliarias disponen de multitud de herramientas para poder gestionar su actividad diaria así como la promoción de su negocio. El principal problema que radica en las mismas es que, o bien son privativas, y conllevan un desembolso importante por parte de la empresa o aquellas que no lo son, dependen de un dominio u empresa externa para su implantación y mantenimiento, no satisfacen todas sus necesidades , tienen muchas que no son de interés o están descentralizadas.

A comienzos de este año se retoma aquella idea pero esta vez con una perspectiva diferente y altura de miras. En lugar de tener en cuenta sólo el punto de vista de GESTICADIZ se ha hecho un estudio exhaustivo del mercado inmobiliario actual (descrito con detalle en la memoria) para generar una herramienta que sirva para cualquier inmobiliaria española pero orientada a la publicidad y al cruce de datos entre los diferentes módulos y la lectura de sus estadísticas generadas y mucho más flexible.

1.2. Alcance

Es un software pensado para satisfacer las necesidades de una inmobiliaria pequeña o mediana. Se dirige principalmente a los agentes inmobiliarios aunque los clientes interactuarán con la zona pública pero tendrán un rol pasivo.

2. Estudio de viabilidad

2.1. Situación actual

Tabla 1. *Comparativa de herramientas del sector.*

	Soluciones hosting		**Soluciones Cloud**		
	Propias	**Wordpress**	**CMS**	**CRM**	**ERP**
Promoción	Depende	Muy buena	Muy buena	Muy Buena	Depende
Gestión	Depende	Regular/Buena	Muy Mala	Buena	Muy buena
Coste	Depende	Muy bueno	Muy bueno	Regular/Malo	Muy Mala
Licencia	Privativo	Libre	Privativo	Privativo	Privativo
Mantenimiento	Regular	Regular	Muy buena	Muy bueno	Muy bueno
Usabilidad	Muy Buena	Regular	Buena	Regular/Buena	Regular
Flexibilidad	Mala	Buena	Buena	Regular/Buena	Regular
Subida a Portales	Muy Malo	Muy Malo	Muy Malo	Muy Buena	Depende

Las principales herramientas con las que rivalizar son las soluciones basadas en Wordpress y CRM. Como referencia, hemos analizado Inmoenter e Inmofactory.

GESTICADIZ cuenta con la antigua aplicación haciendo de backend en un hosting propio y una zona pública programada externamente (ya que el original está obsoleto).

2.2. Necesidades

En la memoria se especifica con detalle las necesidades del sector en general. Para GESTICADIZ, en particular, se detectan las siguientes necesidades en la zona pública:

- Disponer de buscador de inmuebles y visualizarlos conjuntamente en un mapa de Google Maps. Ampliar la información inicial con un detalle.
- Modificar aspecto y contenido con la posibilidad de publicar noticias.
- Diseño responsivo y traducción en varios idiomas de forma flexible y eficiente.

Respecto a la zona privada:

- La introducción de datos iniciales en el sistema es muy laboriosa.
- No existe la posibilidad de configuración de datos del sistema.
- La documentación generada tenía un modelo fijo y no se podía modificar.
- Las copias de seguridad se enviaban por correo y sólo de la base de datos.
- Restricciones implementadas eran poco flexibles con la realidad diaria.

2.3. Objetivos generales

- Crear un software válido para cualquier inmobiliaria española.
- Reducir los costes derivados del software de gestión y promoción inmobiliario.
- Fusionar en una única herramienta todas las características esenciales para que un negocio inmobiliario pueda llevar su actividad diaria.
- Disponer de una web corporativa que promocione a la inmobiliaria, ofreciendo sus inmuebles de forma directa y otros servicios relacionados.
- Manipular fácilmente el sistema en múltiples dispositivos y plataformas.
- Tener suficiente flexibilidad en la configuración de todos los datos del sistema así como de la plantilla de diseño web y facilitar la introducción de datos.
- Disponer de un sistema intuitivo, directo, rápido y fácil de usar.
- Ofrecer vías para que el cliente se comunique con la inmobiliaria.
- Generar documentación de manera rápida y flexible y en diversos formatos.
- Poder realizar copias de seguridad completas del sistema.

- Visualizar estadísticas básicas sobre el funcionamiento del negocio.
- Disponer de seguridad a la hora de acceder a los datos.

2.4. Requisitos funcionales

2.4.1 Sitio Web (Zona pública)

- Ofrecer información pública sobre los inmuebles ofertados para venta o alquiler.
- Poder visualizar un detalle de los inmuebles que amplíe la información inicial.
- Enviar una petición de información de un inmueble en particular.
- Buscar los inmuebles atendiendo a diversos criterios de filtrado.
- Publicar noticias u ofertas que la inmobiliaria considere interesantes.
- Disponer de un formulario de contacto para comunicarse con la inmobiliaria.
- Poder crear cualquier tipo de sección en la que se puedan introducir textos e imágenes así como sub-secciones y páginas con contenidos simples.
- Disponer de una sección en la que muestre donde se encuentra ubicada la empresa en Google Maps y diferentes formas de contactar, horarios, etc.
- Ofrecer toda la información anteriormente descrita en varios idiomas.

2.4.2 Software de gestión (Zona privada)

- Gestionar la cartera de clientes de la inmobiliaria.
- Gestionar los inmuebles que se ofrecen y cuales han sido vendidos o alquilados además de diversos datos de información como las subidas y bajadas de precio.
- Subir imágenes y añadir enlaces externos de los inmuebles y publicarlos.
- Adjuntar ficheros a los inmuebles, clientes y demandas registradas.
- Generar las fichas de información y cartel del inmueble con exportación a PDF.
- Importar la información inicial de clientes e inmuebles desde un CSV.
- Registrar las demandas de forma personalizada pudiendo especificar una serie de criterios básicos para que se puedan cruzar con la cartera de inmuebles.
- Gestionar los diferentes usuarios del sistema y asignar sus permisos.

- Asignar características y sitios cercanos a los inmuebles.
- Poder configurar información a asignar: sitios cercanos, características y tipos de inmuebles, plantillas de documentación, etc.
- Realizar copias de seguridad de la base de datos y de todos los ficheros adjuntos.
- Gestionar el contenido y diseño de la cabecera y pie de página así como el logo.
- Gestionar las secciones y su diseño y contenido.
- Habilitar los idiomas que están activos en la plataforma y poder subirlos.
- Gestionar las noticias publicadas en la web corporativa.
- Mostrar un resumen estadístico de la actividad realizada en el sistema.

3. Planificación

3.1. Metodología de desarrollo

Se ha empleado como principal referencia Métrica v.3 y MADEJA en menor medida.

Como modelo de proceso se ha utilizado el espiral, que permite realizar diferentes iteraciones y prototipos para obtener un rápido feedback del entorno.

Además, he creado una propia metodología enfocada en la integración y evaluación de componentes de software, ya que se han utilizado muchos componentes externos.

3.2. Planificación

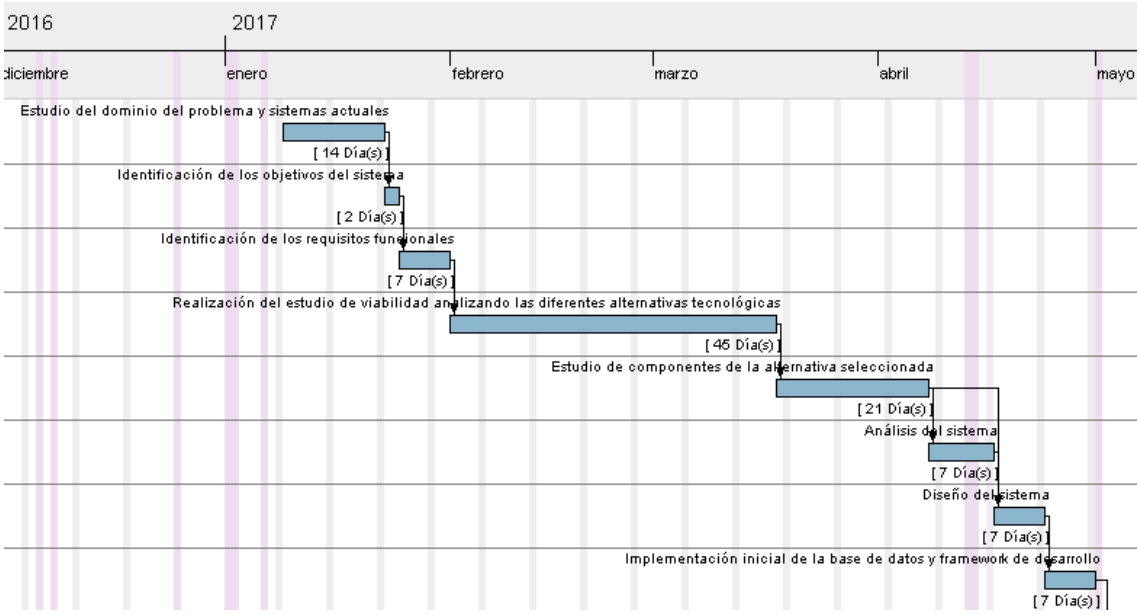

Figura 1. *Diagrama de Gantt de planificación de actividades (parte 1)*

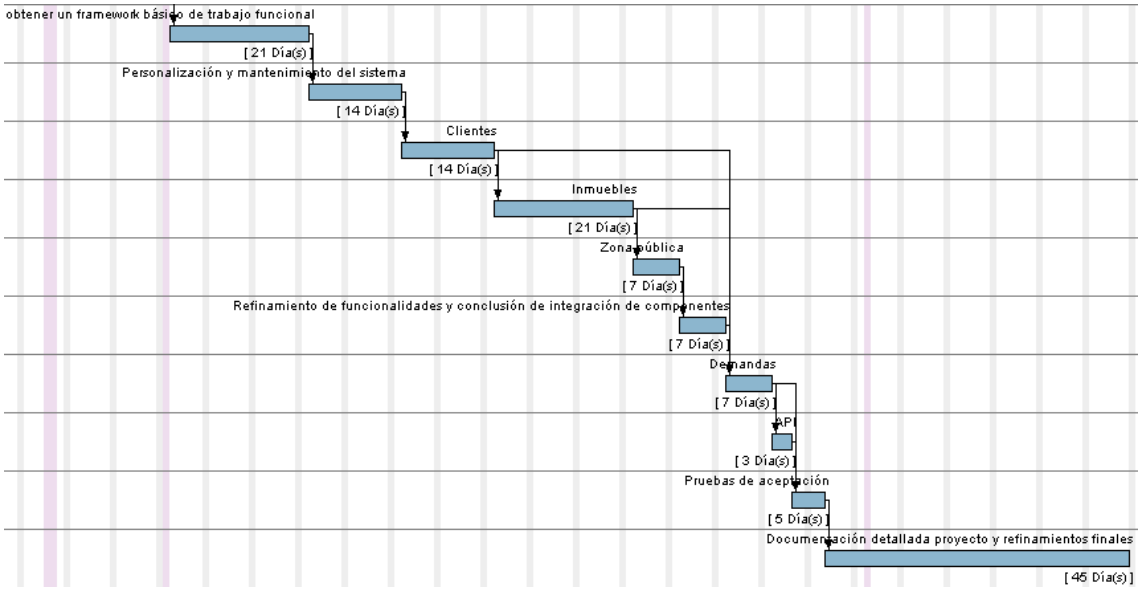

Figura 2. *Diagrama de Gantt de planificación de actividades (parte 2)*

3.3. Costes

Tabla 2. *Previsión de costes basado en cuota de trabajador autónomo (precios sin IVA y IRPF).*

	Cantidad	**Coste unitario**	**Coste total**
Desarrollo	255 d (765 h)	13,75 € / hora	10518,75 €
Hardware y software	Varios	-	957,94 €
Riesgos previstos	15 d (45 h)	13,75 € / hora	618,75 €
Seguros sociales	9 meses	267,03 €	2403,27 €
		Total	14498,71 €

3.4. Organización

Como recursos de hardware se ha utilizado el laptop MSI CX61 2PC (ampliado a 16GB RAM y Disco Duro SSD 250 GB), la Tablet BQ Edison 3 y móvil Alcatel Pixi 4.

Como recursos de software, además de los especificados en la construcción, se han usado Dia v0.97.2, Latex, Trello, Gimp, Gantt Project, y Balsamiq Mockups.

3.5. Riesgos

Se realiza un análisis de riesgos detallado que determina que 15 días afectarán al coste del proyecto y, adicionalmente, otros 15 días afectarán sólo a su duración. Como acciones preventivas se ha ampliado la duración estimada de algunas iteraciones, otras, se han dejado como opciones y se han seguido los criterios de aseguramiento de calidad.

3.6. Aseguramiento calidad

- MADEJA (modelo en W): enfoque de ciclo de vida, prestando especial atención a los productos esperados en cada etapa (documentación y software).
- PCS (Propuesta de cambio de software), Matrices de prueba junto con GIT con despliegue en entorno de pruebas (STAGING) y diario de actividades.

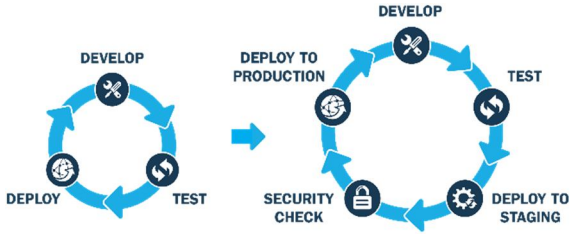

Figura 3. *Evolución de estrategia Develop-Test-Deploy*

4. Análisis del sistema

4.1. Modelo de datos

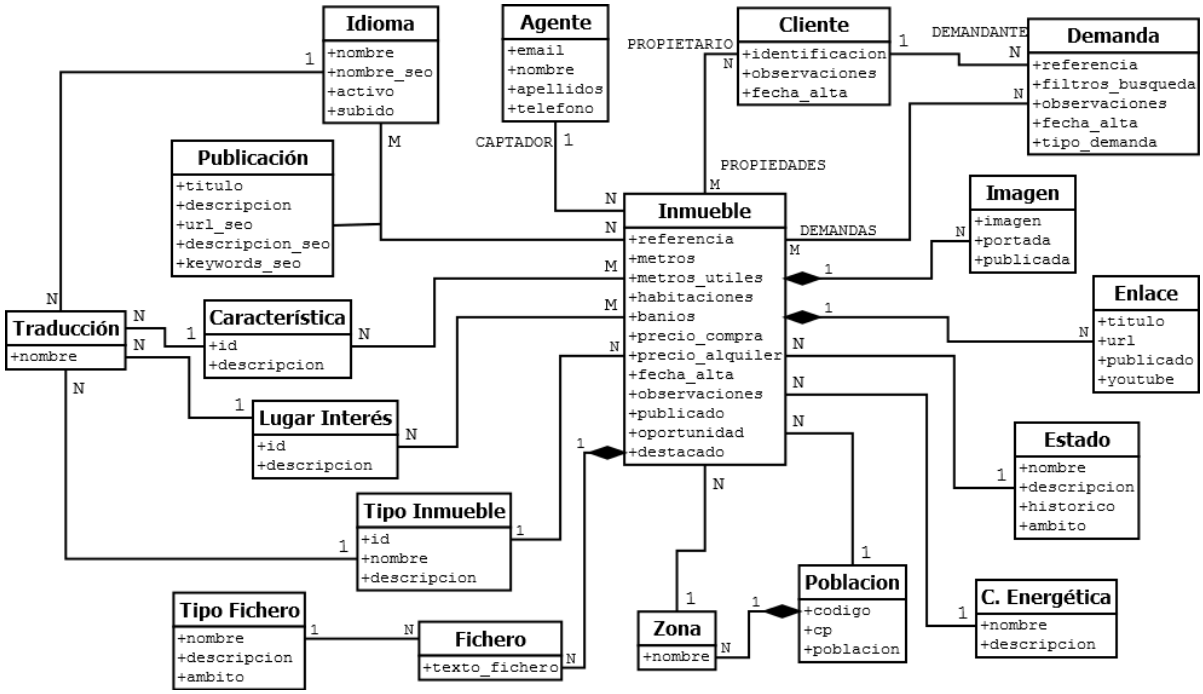

Figura 4. *Diagrama de clases del subsistema de Inmuebles*

4.2. Modelo de casos de uso

4.2.1 Actores

- *Visitante web*: Interactúa únicamente con la zona pública. No necesita registro. Principalmente serán demandantes que buscan un determinado inmueble.
- *Usuario registrado*: Interactúa únicamente con la zona privada:
    - Agente inmobiliario: Su objetivo se centrará en manejar la actividad diaria de la empresa gestionando los datos de las demandas, clientes, inmuebles (y su publicación) y visualizando las estadísticas.
    - Administrador: Su objetivo es configurar los tipos de datos y las preferencias de zona privada y el contenido y formato de la zona pública.

4.2.2 Diagramas

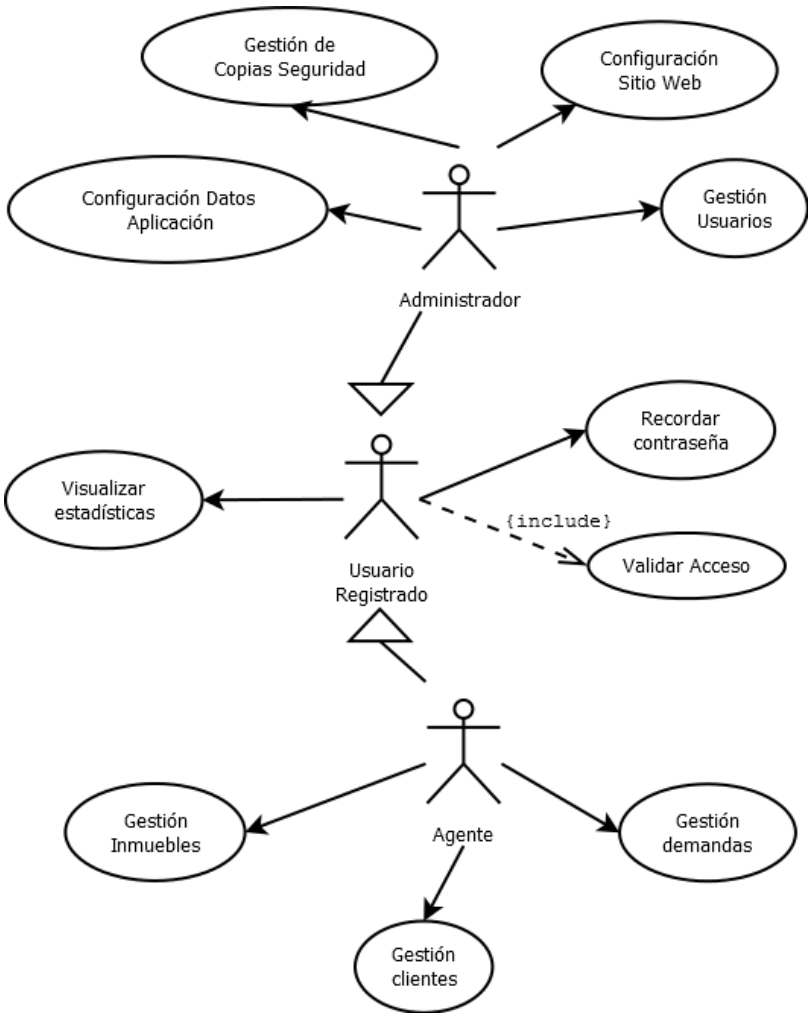

Figura 5. *Diagrama casos de uso (Zona privada)*

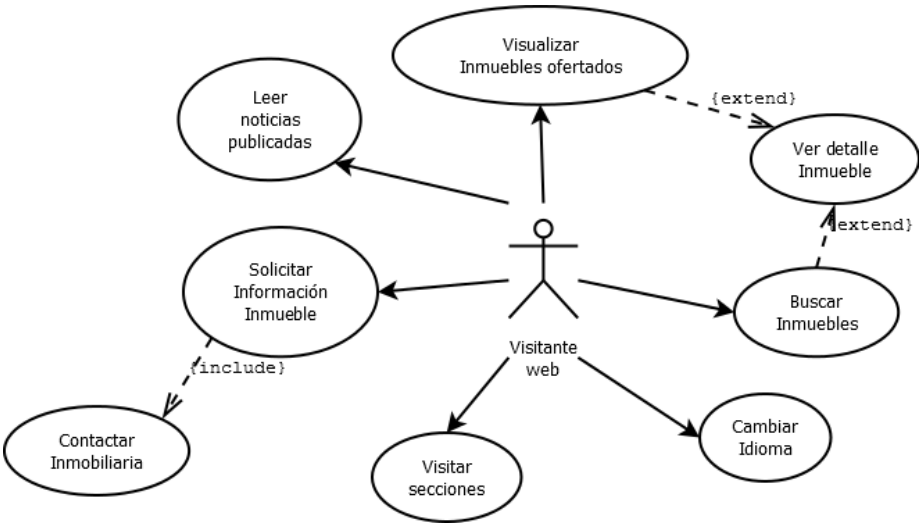

Figura 6. *Diagrama de casos de uso (Zona pública)*

5. Diseño del Sistema

5.1. Arquitectura física

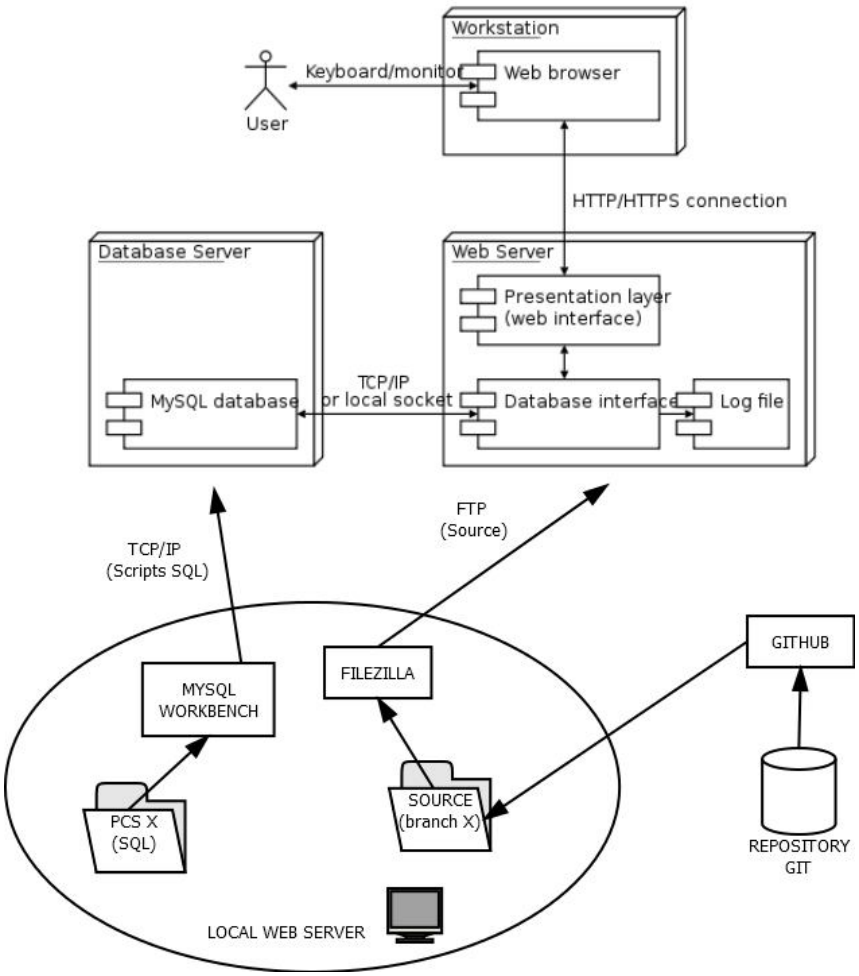

Figura 7. *Diagrama de despliegue*

El anterior diagrama muestra como se ha realizado el proceso de despliegue desde un entorno local a STAGING aplicando las métricas de aseguramiento de calidad.

5.2. Arquitectura lógica

En el sistema se aplican múltiples patrones de diseño como MVC (Modelo-Vista-Controlador), Singleton, Layout (para la visualización de interfaces) entre otros. Algunos se han implementado manualmente y otros ya los aportaba Codeigniter 3.

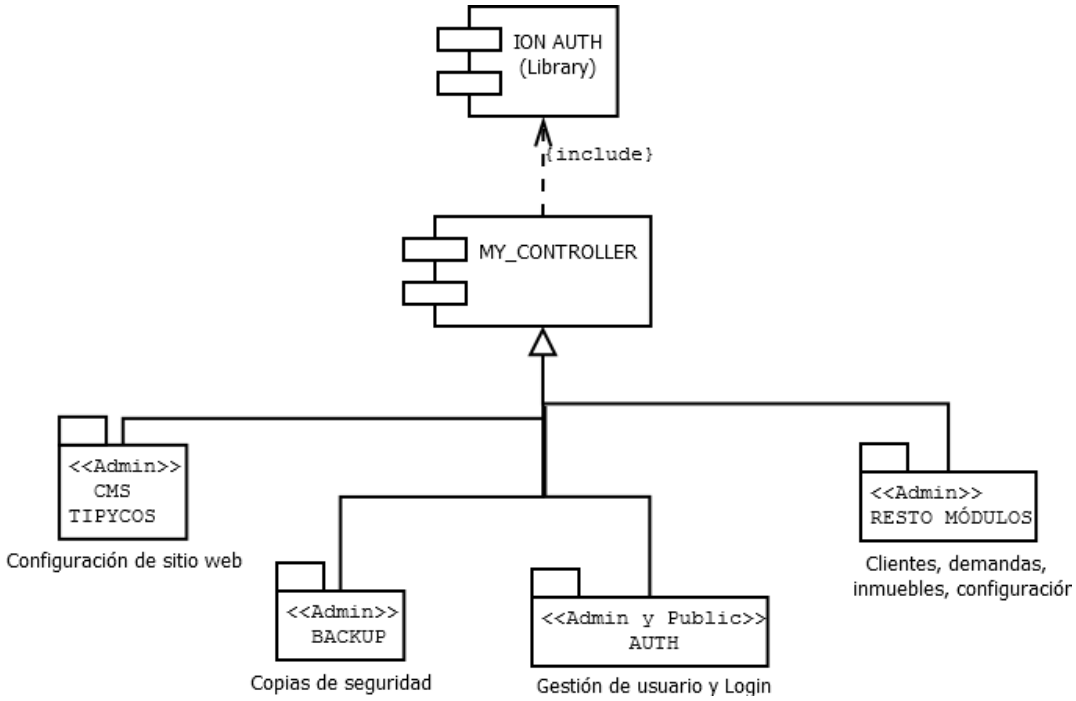

Figura 8. *Diagrama de diseño de software de gestión*

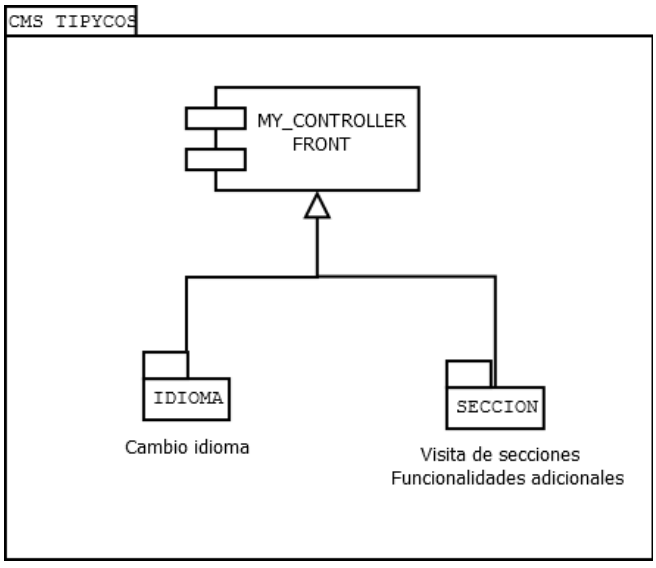

Figura 9. *Diagrama de diseño de sitio web*

El sistema Consta, principalmente, de 3 componentes que ha habido que integrar y adaptar y nos han servido como base para cumplir diversas funcionalidades: CMS TIPYCOS, MY_Backup y ION AUTH (AUTH módulo prefinido basado en la librería).

A nivel de controlador, se han implementado dos tipos diferentes. MY_Controller, para manejar la zona privada (control de acceso, nivel de permisos, etc.) pero con flexibilidad para poder manejar eventos públicos. En cambio, MY_Controller_Front, contiene funcionalidades para realizar un display óptimo de la zona pública.

A nivel de modelo, se utilizan dos capas suplementarias. BASE_Model, componente externo usado para manejar relaciones y con funcionalidades CRUD y MY_Model, implementado manualmente, que añade otras características como el control de errores.

Se ha diseño un nivel de permisos de acceso a ficheros para los componentes por cada módulo del sistema, de forma que cada sección acceda a la información que le interesa.

Además el sistema interactúa con 4 sistemas externos: API de Google Maps v3, Google Analitycs, Recaptcha de Google y Youtube. Otros muchos componentes y librerías tanto del backend como del frontend han sido empleados y se detallan en la memoria.

6. Construcción

6.1 Frameworks

Se han utilizado en el frontend tanto Bootstrap 3 (3.2 y 3.3.1) como Materialize 0.97 junto con Jquery (diferentes versiones). Para el backend se ha usado Codeigniter 3.1.

6.2 Lenguajes de programación

PHP 5.5.12, MySQL 5.6.17, Javascript, HTML 5, CSS 3 y LaTeX.

6.3 Herramientas de desarrollo

MySQL Workbench 6.3.6, PHPMyAdmin 4.1.14, Git 1.9.15 y repositorio en Github (con documentación online), SourceTree 1.6.18, WampServer 2.5, Netbeans IDE 8.1, Doxygen 1.8.13, Composer 1.3.2, Filezilla 3.27.1.

6.4 Buenas prácticas

Sobre **Codeigniter** que afectan a cómo debe estructurarse el código, configuración en entornos, herramientas para depuración y testing, estrategia Thin Controllers – Fat Models, medidas de seguridad, uso de caché, dependencia de paquetes y PSR.

Sobre la **Base de datos**, como el uso de librería PDO o librería CRUD para usarlas como base y el registro de las migraciones en SQL en diferentes PCS.

También se detallan **Otras buenas prácticas** de índole general en la memoria.

7. Pruebas

7.1. Estrategia

Las pruebas pretenden comprobar que el sistema funciona correctamente en cada capa de diseño, así como en cada nivel de abstracción y que se comunica correctamente con los sistemas externos y que los componentes y módulos empleados se adaptan a las necesidades detectadas e interaccionan correctamente con el resto del sistema**.**

7.2. Roles

7.2.1 Pruebas Unitarias

Se han realizado a través de programación en controladores de test usando la librería de testing, con marcas en el código y con el uso del Profiler del framework.

7.2.2 Pruebas de Integración

- Comunicación de los distintos métodos para satisfacer una funcionalidad.
- Comunicación de los componentes de las librerías empleadas en el sistema verificando la comunicación no sólo con el framework si no también, con los modelos donde se iban a utilizar principalmente.
- Comunicación de los artefactos de software con los sistemas externos.
- Comunicación de lo interfaz usando en AJAX.

Dependiendo de la prueba, se ha hecho un uso más intensivo de cada herramienta.

7.2.3 Pruebas de Sistema

Se han simulado a través del propio framework con controladores de prueba o pasando las Matrices de prueba (listas de comprobación) en un navegador.

- Pruebas funcionales.
- Pruebas no funcionales.

- *Pruebas de rendimiento*: A través del profiler se ha comprobado que las peticiones se resuelven en un segundo como máximo salvo excepciones.
- *Pruebas de adaptabilidad de interfaz a múltiples dispositivos*: Diferentes tablets, smartphones y ordenadores personales en diferentes resoluciones.
- *Pruebas de portabilidad*: Mac, Linux y Windows y en varios entornos.
- *Pruebas de carga*: Comprobación de visualización de varios inmuebles en Google Maps y listado de elementos en tablas con datatables.js.
- *Pruebas de concurrencia*: Se realizaban abriendo dos navegadores diferentes ejecutando la misma funcionalidad.

- Pruebas de regresión: se especifican, normalmente, en las matrices de prueba, indicando a qué casos de uso o aspecto concreto afecta un determinado cambio.

7.2.4 Pruebas de aceptación

Se utilizaron múltiples perfiles de usuario para validar que el producto, en sus diferentes ámbitos, estaba operativo para desplegarse en un entorno de producción.

8. Conclusiones

8.1. Dimensiones del proyecto

- Dos frameworks de frontend y un framework de backend (con múltiples componentes de diferentes ámbitos).
- Tres módulos añadidos, personalizados e integrados y siete librerías externas.
- Más de 320 commits realizados, 40 ramas de desarrollo y más de 1000 horas.
- Más de 80000 líneas de código programadas (el sistema más de 190000).
- 74 tablas, 12 vistas, más de 120 índices y 10000 registros en la base de datos.

8.2. Objetivos alcanzados

Se han conseguido los principales objetivos e incluso mejorado las expectativas de algunos de ellos. No obstante, las fichas de visita y su documentación así como la APIs o la subida de inmuebles a portales inmobiliarios quedaron sin realizar.

8.3. Dificultades

- Ámbito, necesidades y herramientas actuales complejo.
- Flexibilidad vs Automatización en funcionalidades y programación.
- Valoración, aprendizaje e integración de componentes.
- Implantación en entornos reales.
- Gran dificultad a la hora de determinar el alcance del sistema.
- Riesgos imprevistos en un calendario con fecha de entrega inamovible.
- Gran variedad de test en múltiples entornos y dispositivos.
- Nivel de auto-exigencia global elevadísimo.
- Desarrollo completo del proyecto totalmente a distancia.

8.4. Lecciones aprendidas

- Se ha actualizado el entorno tecnológico empleado a todos los niveles.
- Se han adquirido buenas prácticas en diferentes ámbitos.

8.5. Trabajo Futuro

- Crear API y módulos en diferentes CMS (principalmente en Wordpress).
- Subida automática a portales inmobiliarios a través de sus APIs.
- Módulo de Acciones (Agenda integrada con Google Calendar).
- Módulo para Clientes (facturas, comunicación directa con inmobiliaria, etc.)

8.6. Valoración personal

- He intentado obtener un producto único en su ámbito y creo que lo he logrado, ya que se han conseguido resultados profesionales que rivalizan con las herramientas comerciales existentes y en algunos casos incluso las supera.
- Los conocimientos y las buenas prácticas adquiridas me van a permitir adaptarme mejor a la realidad del mercado como analista y programador.
- Este proyecto cierra una etapa de mi vida que lleva estancada más de 10 años y ha supuesto un esfuerzo desproporcional en comparación a lo que se supone que debe ser un PFC (OPENINMOCMS ya de por sí era un PFC completo).

9. Referencias bibliográficas

9.1. Bibliografía

1. R.S. Presmann, *Ingeniería del Software, un enfoque práctico*, McGrawHill, Edición 6, (2005).

2. I. Somerville, *Ingeniería de Software*, Pearson, Edición 9, (2011).

3. C. Larman, *UML y Patrones. Una introducción al análisis y diseño orientado a objetos y al proceso unificado*, Pearson, Edición 2, (2003).

4. Miguel López Morales. *Sistema Divandroid*. UCA, 2014.

5. Juan Antonio Caballero Hernández. *Sistema para la evaluación de competencias en LMS mediante servicios web*. UCA, 2014.

9.2. Referencias en internet

6. Ministerio de Administraciones Públicas: *Metodología de Planificación y desarrollo de sistemas de información v.3*, http://administracionelectronica.gob.es.

7. *Marco de Desarrollo de la Junta de Andalucía* (MADEJA): www.madeja.es.

8. Página *web oficial de Codeigniter*, www.codeigniter.com.

9. Página *web oficial de Bootstrap*, www.bootstrap.com.

10. Consulta de errores y dudas de implementación, www.stackoverflow.com.

11. Tutoriales online sobre múltiples tecnologías web, https://www.w3schools.com.

12. Tutoriales online de múltiples tecnologías, https://code.tutsplus.com.

13. Sitio web con diapositivas de diferente temática, https://es.slideshare.net.

10. Agradecimientos

Muchas personas han colaborado en el proyecto: mi tutor, mi mujer, mi hermana, la empresa TIPYCOS entre otros. Pero principalmente, se lo dedico a mis padres.