



Deusto

Facultad de Ingeniería
Universidad de Deusto

Ingeniaritza Fakultatea
Deustuko Unibertsitatea

Ingeniero en Informática **Informatikako Ingeniaria**

Proyecto fin de carrera
Karrera amaierako proiektua

Resumen

El proyecto se basa en el diseño y desarrollo de una aplicación web que aproveche datos públicos abiertos por el Gobierno Vasco. El Gobierno Vasco publica en sus diferentes sitios de Internet gran variedad de datos acerca de tráfico, estadísticas, formación... En muchas ocasiones estos datos no son fácilmente accesibles ni utilizables por los ciudadanos, por tanto existe la necesidad de aprovechar estos datos y hacerlos fácilmente accesibles según las necesidades de los usuarios.

Este proyecto pretende responder a este hecho, y en concreto, pretende hacer más accesible la información acerca de ciclos formativos coordinados por el área de innovación del Gobierno Vasco, publicada en la web www.euskadinnova.net. Para ello, se ha decidido realizar una aplicación web que incluya, además de un calendario de los ciclos, información extendida acerca de los temas que tratarán y los lugares en los que se realizarán. Esta aplicación mostrará estos datos de manera intuitiva, accesible y agradable, y estará adaptada a la visualización desde distintas plataformas.

Además, este web-mashup -web que utiliza datos y recursos multimedia de diferentes sitios de Internet para crear una aplicación de valor añadido- se englobará dentro de la llamada Web Semántica, es decir, será una aplicación web que contendrá metadatos semánticos y ontologías con el fin de que pueda interoperar con otras webs sin necesidad de operadores humanos.

Resumiendo, se trata del desarrollo de una aplicación web que hará uso de nuevas tecnologías web y hará más accesible y completa la información del área de innovación del Gobierno Vasco.

Descriptores

- Mashup
- Web semántica
- RDF
- Open Data (Datos Abiertos)
- Web móvil

Índice

1.	Introducción.....	1
1.1	Presentación del documento.....	1
1.2	Contextualización del proyecto	2
2.	Definición del proyecto.....	5
2.1	Objetivos.....	5
2.2	Alcance.....	6
3.	Descripción de la realización	7
3.1	Método de desarrollo.....	7
3.2	Fases del proyecto	8
3.2.1	Análisis	8
3.2.2	Estudio del Estado del Arte	9
3.2.3	Diseño de la aplicación web.....	10
3.2.4	Desarrollo de la aplicación web.....	10
3.2.5	Pruebas	10
3.2.6	Paso a producción.....	11
3.2.7	Documentación	11
4.	Análisis.....	13
4.1	Análisis de requisitos.....	13
5.	Estado del Arte.....	15
5.1	Evolución de la Web: de la Web 1.0 a la Web Semántica.....	15
5.2	Web Semántica	19
5.3	Open Data en la Comunidad Autónoma Vasca	22
5.4	Mashups.....	23

6.	Diseño	25
6.1	Selección de fuentes de datos	25
6.2	Selección de formatos de publicación	26
6.3	Arquitectura	27
6.3.1	Módulo de obtención de datos de Euskadi+Innova	29
6.3.2	Módulo de geolocalización de eventos	31
6.3.3	Módulo de actualización de Google Calendar.....	33
6.3.4	Módulo de comprobación de comprobación de eventos finalizados.....	33
6.3.5	Módulo de publicación	34
6.3.6	Módulo de administración.....	34
6.3.7	Módulo de API	34
6.4	Diseño de la interacción con el usuario	35
6.4.1	Interacción con la versión de escritorio del módulo de publicación	36
6.4.2	Interacción con la versión móvil del módulo de publicación	38
6.4.3	Interacción con módulo de administración	40
6.4.4	Interacción con el módulo de API	41
6.5	Diseño de la base de Datos	42
7.	Desarrollo	45
7.1	Herramientas utilizadas	45
7.1.1	Framework de desarrollo web: Django y Python.....	45
7.1.2	Librerías Python utilizadas	48
7.1.3	Módulos Django utilizadas.....	50
7.1.4	Base de datos: PostgreSQL	53
7.1.5	Otras herramientas: JQueryMobile.....	54
7.2	Funcionamiento de Django.....	54

7.3	Estructura del proyecto	55
7.3.1	Resumen de paquetes Python del proyecto	58
7.4	Diagrama de clases.....	59
7.5	Implementación	61
7.5.1	Implementación del modelo de datos	61
7.5.2	Implementación del módulo de obtención de datos de Euskadi+Innova	62
7.5.3	Implementación del módulo de geolocalización de eventos	66
7.5.4	Implementación del módulo de actualización de Google Calendar	69
7.5.5	Implementación del módulo de comprobación de eventos finalizados	69
7.5.6	Implementación del módulo de publicación	70
7.5.7	Implementación del módulo de administración	77
7.5.8	Implementación del módulo de API	78
7.5.9	Implementación de la internacionalización y localización del sitio.....	80
8.	Planificación	81
8.1	Tareas	81
8.2	Equipo real	86
8.3	Plan de trabajo	89
8.4	Cargas reales de trabajo por tarea	90
8.5	Cargas reales de trabajo por recurso.....	98
8.6	Diagrama de Gantt	114
8.7	Diagrama de precedencias	117
9.	Conclusiones y líneas futuras	123
9.1	Conclusiones finales	123
9.2	Líneas futuras.....	124
10.	Bibliografía	127

11.	Agradecimientos	131
	Anexo I: Manual de Usuario	1
	Anexo II: Paso a producción del proyecto	13

Índice de figuras

Figura 3.1: Esquema temporal de las fases de desarrollo del proyecto	8
Figura 5.1: Estructura de la Web 1.0	16
Figura 5.2: Estructura de la Web 2.0	17
Figura 5.3: Mapa de la evolución de la web	18
Figura 5.4: Estructura de un RDF	20
Figura 5.5: Imagen conceptual de la web vista como una red de linked data	21
Figura 6.1: Arquitectura general de la aplicación web	28
Figura 6.2: Diagrama de flujo del módulo de obtención de datos de Euskadi+Innova.....	30
Figura 6.3: Diagrama de flujo del módulo de geolocalización automática.....	32
Figura 6.4: Diagrama de flujo del módulo de actualización de Google Calendar.....	33
Figura 6.5: Diagrama de flujo del módulo de comprobación de eventos finalizados	33
Figura 6.6: Diagrama de flujo básico del módulo de publicación de contenido.....	34
Figura 6.7: Diagrama de flujo del módulo de API	35
Figura 6.8: Mapa conceptual de la navegación de diálogos de la versión de escritorio del módulo de publicación.....	37
Figura 6.9: Mapa conceptual de la navegación de diálogos de la versión móvil del módulo de publicación	39
Figura 6.10: Mapa conceptual de la navegación de diálogos del módulo de administración.....	40
Figura 6.11: Diagrama del modelo relacional de la base de datos.....	42
Figura 6.12: Esquema del modelo relacional de la base de datos	43
Figura 7.1: Tabla de comparación de frameworks web	46
Figura 7.2: Esquema de funcionamiento de Django.....	55
Figura 7.3: Estructura de carpetas del proyecto	56
Figura 7.4: Estructura del paquete de controladores	57

Figura 7.5: Estructura de la aplicación Django.....	57
Figura 7.6: Contenido de las carpetas de la aplicación	58
Figura 7.7: Diagrama de estructura de paquetes de la aplicación.....	59
Figura 7.8: Diagrama de clases del proyecto.....	60
Ilustración 7.9: Implementación del modelo Event.....	61
Figura 7.10: Código de inicialización del módulo de obtención de datos	63
Figura 7.11: Código de conexión del módulo de obtención de datos	64
Figura 7.12: Código de parseo HTML del módulo de obtención de datos.....	65
Figura 7.13: Código de guardado en base de datos del módulo de obtención de datos	66
Figura 7.14: Código de búsqueda de localizaciones frecuentes en el módulo de geolocalización	67
Figura 7.15: Código de búsqueda de direcciones en GMaps Geolocalization API	68
Figura 7.16: Código de autenticación en GData API	69
Figura 7.17: Código de preparación del evento de GCalendar.....	69
Figura 7.18: Código de creación del evento de GCalendar	69
Figura 7.19: Relación entre patrones de URLs y Vistas (urls.py)	70
Figura 7.20: Implementación de la vista de inicio	71
Figura 7.21: Implementación de la vista de evento.....	72
Figura 7.22: Ejemplo de manejo de datos en plantillas	72
Figura 7.23: Script JS de geolocalización del usuario y uso de la API JS de GMaps	75
Figura 7.24: Interfaz de usuario del sub-módulo de visualización del mapa (versión escritorio y móvil)	75
Figura 7.25: Ejemplo de marcado semántico múltiple en plantilla de evento	76
Figura 7.26: Código de los botones sociales de Twitter, Facebook y G+	76
Figura 7.27: Código de la inserción del sistema de comentarios Disqus	77

Figura 7.28: Interfaz de usuario del sub-módulo de visualización de un evento (versión escritorio y móvil)	77
Figura 7.29: Implementación del método de filtrado redefinido para el módulo de API	79
Figura 7.30: Código del serializador RDF para Tastypie	80
Figura 8.1: Diagrama de plan de trabajo.....	89
Figura 8.2: Diagrama de cargas reales por tarea (04/04/2011-17/04/2011)	90
Figura 8.3: Diagrama de cargas reales por tarea (18/04/2011-01/05/2011)	91
Figura 8.4: Diagrama de cargas reales por tarea (02/05/2011-15/05/2011)	92
Figura 8.5: Diagrama de cargas reales por tarea (06/06/2011-19/06/2011)	93
Figura 8.6: Diagrama de cargas reales por tarea (20/06/2011-03/07/2011)	94
Figura 8.7: Diagrama de cargas reales por tarea (04/07/2011-17/07/2011)	95
Figura 8.8: Diagrama de cargas reales por tarea (18/07/2011-31/07/2011)	96
Figura 8.9: Diagrama de cargas reales por tarea (01/08/2011-16/08/2011)	97
Figura 8.10: Diagrama de cargas reales para Jefe de Proyecto, Director del Proyecto, Analista y Técnico (04/04/2011-17/04/2011)	98
Figura 8.11: Diagrama de cargas reales para Programador (04/04/2011-17/04/2011)	99
Figura 8.12: Diagrama de cargas reales para Jefe de Proyecto, Director del Proyecto, Analista y Técnico (18/04/2011-01/05/2011)	100
Figura 8.13: Diagrama de cargas reales para Programador (18/04/2011-01/05/2011)	101
Figura 8.14: Diagrama de cargas reales para Jefe de Proyecto, Director del Proyecto, Analista y Técnico (02/05/2011-15/05/2011)	102
Figura 8.15: Diagrama de cargas reales para Programador (02/05/2011-15/05/2011)	103
Figura 8.16: Diagrama de cargas reales para Jefe de Proyecto, Director del Proyecto, Analista y Técnico (06/06/2011-19/06/2011)	104
Figura 8.17: Diagrama de cargas reales para Programador (06/06/2011-19/06/2011)	105
Figura 8.18: Diagrama de cargas reales para Jefe de Proyecto, Director del Proyecto, Analista y Técnico (20/06/2011-03/07/2011)	106

Figura 8.19: Diagrama de cargas reales para Programador (20/06/2011-03/07/2011).....	107
Figura 8.20: Diagrama de cargas reales para Jefe de Proyecto, Director del Proyecto, Analista y Técnico (04/07/2011-17/07/2011)	108
Figura 8.21: Diagrama de cargas reales para Programador (04/07/2011-17/07/2011).....	109
Figura 8.22: Diagrama de cargas reales para Jefe de Proyecto, Director del Proyecto, Analista y Técnico (18/07/2011-31/07/2011)	110
Figura 8.23: Diagrama de cargas reales para Programador (18/07/2011-31/07/2011).....	111
Figura 8.24: Diagrama de cargas reales para Jefe de Proyecto, Director del Proyecto, Analista y Técnico (01/08/2011-16/08/2011)	112
Figura 8.25: Diagrama de cargas reales para Programador (01/08/2011-16/08/2011).....	113
Figura 8.26: Diagrama de Gantt (04/04/2011-05/06/2011)	114
Figura 8.27: Diagrama de Gantt (06/06/2011-31/07/2011).....	115
Figura 8.28: Diagrama de Gantt (01/08/2011-16/08/2011)	116
Figura 8.29: Diagrama de precedencias I	117
Figura 8.30: Diagrama de precedencias II	118
Figura 8.31: Diagrama de precedencias III	119
Figura 8.32: Diagrama de precedencias IV.....	120
Figura 8.33: Diagrama de precedencias V	121
Figura 8.34: Diagrama de precedencias VI.....	122

1. INTRODUCCIÓN

En el primer capítulo de este documento, se ofrecerá una visión general de la estructura del contenido del propio documento, así como una introducción al proyecto realizado y a su contexto.

1.1 PRESENTACIÓN DEL DOCUMENTO

El presente documento describe el proyecto de desarrollo de InnovAgenda.com, una aplicación web en forma de mashup semántico basado en datos públicos publicados por el Área de Innovación del Gobierno Vasco en la web <http://www.euskadinnova.net>.

Este proyecto es el resultado del desarrollo de la propuesta del director del proyecto – Dr. Diego López-de-Ipiña- de la realización de un web-mashup relacionado con datos abiertos por el Gobierno Vasco, y que por una parte hiciera uso de nuevas herramientas de la llamada Web Semántica, y por el otro que fuera adaptable a su visualización en terminales móviles.

La elaboración de este documento es consecuencia de las reuniones entre el equipo y el Director del Proyecto, en las cuales se han ido detallando los aspectos teóricos y técnicos referentes a este proyecto.

El contenido de este documento se estructura en torno a los siguientes puntos:

1. **Introducción:** En el que se explica la estructura del presente documento y se presenta el proyecto y se hace una breve introducción al contexto que lo rodea.
2. **Definición del proyecto:** En el que se establecen los objetivos principales del proyecto, así como su alcance.
3. **Descripción de la realización:** En el que se explican las diferentes fases del desarrollo del proyecto, así como la metodología seguida.
4. **Análisis:** En el que se analizan los requisitos para el diseño y desarrollo del proyecto.
5. **Estado del arte:** En el que se analizan y explican los conceptos que rodean el contexto del proyecto.

6. **Diseño:** En el que se explican las directrices principales en las que se ha basado el diseño de la aplicación web.
7. **Desarrollo:** En el que se muestra cómo es y cómo ha sido desarrollado el proyecto desde el aspecto técnico.
8. **Planificación:** En el que se detalla la forma en la que se ha organizado la realización de las diferentes tareas del proyecto, desde un punto de vista temporal y logístico.
9. **Conclusiones y líneas futuras:** En el que se valoran todos los aspectos referentes a la realización del Proyecto de Fin de Carrera, y en el que se analizan las posibilidades futuras de uso y mejora del proyecto.

Además de estos puntos, el documento incluye referencias a la bibliografía utilizada para el desarrollo de este documento, así como un Manual de Usuario de la aplicación web resultante y la descripción de su paso a producción, a modo de anexos.

1.2 CONTEXTUALIZACIÓN DEL PROYECTO

El Gobierno Vasco publica en diferentes sitios de Internet una gran variedad de información acerca de tráfico -www.trafikoa.net-, meteorología -www.euskalmet.euskadi.net-, innovación -www.euskadinova.net- o sanidad -www.osakidetza.euskadi.net-, entre otros. Muchas de estas páginas web del Gobierno Vasco tienden a tener los siguientes problemas:

- Son páginas poco estructuradas y en las que resulta complicado encontrar información específica, ya que tienden a publicar demasiados datos.
- Son sitios dedicados a la publicación de información, pero que en general no se aprovechan de las nuevas formas de publicación, como la sindicación RSS o los recursos de la Web Semántica.
- Se trata de webs no adaptadas a las necesidades de los usuarios de terminales móviles, cada vez más utilizados, y en los que resulta complicada la consulta de dichos datos.
- Al dedicarse únicamente a la publicación, dejan de lado la posibilidad de feedback y opinión de los usuarios, lo cual desaprovecha la posibilidad de mantener medios de comunicación vía web acerca de los datos publicados entre Gobierno Vasco y ciudadanos, y entre los propios ciudadanos.

En respuesta a los problemas en cuanto a desestructuración y publicación, el Gobierno Vasco lanzó en 2010 la plataforma Open Data Euskadi -opendata.euskadi.net-, en la cual se publica información por temáticas y en diferentes formatos de publicación. La mayor carencia

de este sitio es la poca información de la que dispone por el momento, ya que se trata de un proyecto en crecimiento.

Una de las webs del Gobierno Vasco que aún mantiene algunos de los problemas anteriormente explicados es la web del Área de Innovación del Gobierno Vasco, Euskadi+Innova –euskadinnova.net-. Euskadi+Innova es una web que publica noticias e información general acerca de diferentes áreas de innovación; además de ello, mantiene una agenda de eventos relacionados con ciclos formativos y ponencias acerca de esta temática. Los eventos de esta agenda son actividades con una demanda creciente, debido principalmente a la necesidad de empresas y trabajadores de actualizarse y trabajar en nuevos procesos de innovación. Estos eventos pueden ser consultados en la propia web y vía sindicación RSS, pero en visualizarlos en la web no resulta demasiado sencillo, y la sindicación RSS no sigue los estándares y es incoherente. Además, el sitio desaprovecha la posibilidad de crear una comunidad de asistentes a dichos eventos, y no se adapta a la visualización en dispositivos móviles.

Motivado por el interés creciente por estos eventos, y por dar respuesta a las carencias de la web que los publica, nace el proyecto de desarrollo de un web-mashup que centralice, enriquezca y re-publique la información de dichos eventos utilizando nuevas tecnologías de publicación en Internet y les dé un valor añadido.

2. DEFINICIÓN DEL PROYECTO

En este apartado se definirán los objetivos principales y específicos del Proyecto de Fin de Carrera, así como su alcance, con el objetivo de concretar sus límites de actuación.

2.1 OBJETIVOS

El proyecto de desarrollo definido en el presente documento tiene como **objetivo principal** el desarrollo de un web-mashup -aplicación web que contiene información de diversos sitios web para darles valor añadido- que recoja la información de la agenda de eventos del Área de Innovación del Gobierno Vasco, la centralice, enriquezca e integre con otras plataformas, y la re-publique utilizando tecnologías web semántica y una interfaz agradable y adaptable.

Los **objetivos específicos** del proyecto se centran en los siguientes puntos:

- Ampliar la información de los eventos obtenidos de www.euskadinnova.net (añadir geolocalización, etc.).
- Integración de la información con diversas plataformas.
- Investigación e utilización de tecnologías web semánticas como RDF, RDFa o Microdatos.
- Desarrollo de la web utilizando HTML5.
- Inclusión de tecnologías Web 2.0 en el sitio resultante, para que sirva de punto de encuentro para organizadores, coordinadores, ponientes y asistentes de dichos eventos.
- Desarrollo de una Interfaz de Programación de Aplicaciones (API) para el acceso a los datos que se publiquen en el mashup.

Las **fases del proyecto** se pueden resumir en los siguientes puntos:

- Análisis de requisitos y estudio de las necesidades del proyecto.

- Investigación acerca de las actuales herramientas, fuentes de datos, formatos y aplicaciones de la Web Semántica.
- Estudio de las herramientas y fuentes de datos aplicables a la aplicación.
- Diseño de la aplicación web y sus componentes.
- Desarrollo de la aplicación.
- Documentación completa de todo el desarrollo e investigación llevados a cabo.

Como **plataforma técnica** para cumplir estos objetivos se ha dispuesto de un equipo Asus U30JC con Ubuntu 11.04 como Sistema Operativo, sobre el cual se han desarrollado las diversas fases del proyecto y en el cual ha corrido el servidor y la base de datos de desarrollo. En la fase de producción el servicio de hosting ha sido proporcionado por www.dotcloud.com. La aplicación ha sido desarrollada bajo el framework de desarrollo web Django y diversas librerías basadas en Python, y la base de datos está basada en PostgreSQL.

2.2 ALCANCE

Los límites de actuación en este proyecto serán los que se establecen a continuación:

- Proporcionar mayor facilidad de consulta y utilización a la información acerca de ciclos formativos proporcionados por el área de innovación del Gobierno Vasco.
- Ampliar y enriquecer de la información acerca de los cursos.
- Dar mayor accesibilidad a los recursos de la web desde diferentes plataformas, tanto móviles como de escritorio.
- Aportar recursos que ayuden a la transformación de la agenda de innovación del Gobierno Vasco en Open Data semántica y reutilizable.
- Añadir al mashup capacidad de interoperabilidad con otras webs gracias a su API y a que publica su contenido en formatos adaptados a la web semántica.
- Proporcionar un área privada para realizar tareas de administración y traducción de los datos de la web de forma sencilla.

3. DESCRIPCIÓN DE LA REALIZACIÓN

En el tercer capítulo se describirá el método de desarrollo utilizado para la consecución de las diferentes fases del proyecto, ofreciendo una visión general de cada una de las fases, que serán explicadas con más detalle en los siguientes capítulos.

3.1 MÉTODO DE DESARROLLO

Debido a que el equipo de trabajo está formado por una sola persona, el método de desarrollo que se ha seguido para el desarrollo del proyecto es un método de desarrollo en cascada, en el cual se ha dividido el trabajo en diferentes fases, las cuales son dependientes de la tarea inmediatamente anterior. Las fases del proyecto son las siguientes:

- Fase de análisis de requisitos.
- Fase de estudio del Estado del Arte.
- Fase de diseño de la aplicación web.
- Fase de desarrollo.
- Fase de pruebas.
- Fase de paso a producción.
- Fase de documentación completa de lo realizado en las diferentes tareas anteriores.

Habiéndose desgranado cada una de las fases en diversas tareas.

Aun así, no se ha seguido rigurosamente este tipo de desarrollo, ya que en muchas ocasiones, en fases posteriores se han realizado cambios (ya sea por nuevas ideas de implantación, o por problemas que requerían otro tipo de soluciones), lo cual obliga a realizar cambios en fases anteriores.

Además, no se puede considerar que la fase de estudio del Estado del Arte empiece y acabe en un espacio temporal limitado, ya que se trata de un trabajo de investigación que, a pesar de tener la mayor carga de trabajo antes de comenzar el diseño de la aplicación, se ha

ido realizando de forma constante a lo largo de las fases de análisis, diseño, desarrollo y pruebas del proyecto.

Por lo tanto se puede decir que se ha seguido un método de desarrollo en cascada, pero abierto a constantes revisiones. El esquema temporal del método de desarrollo de este proyecto responde a este diagrama:

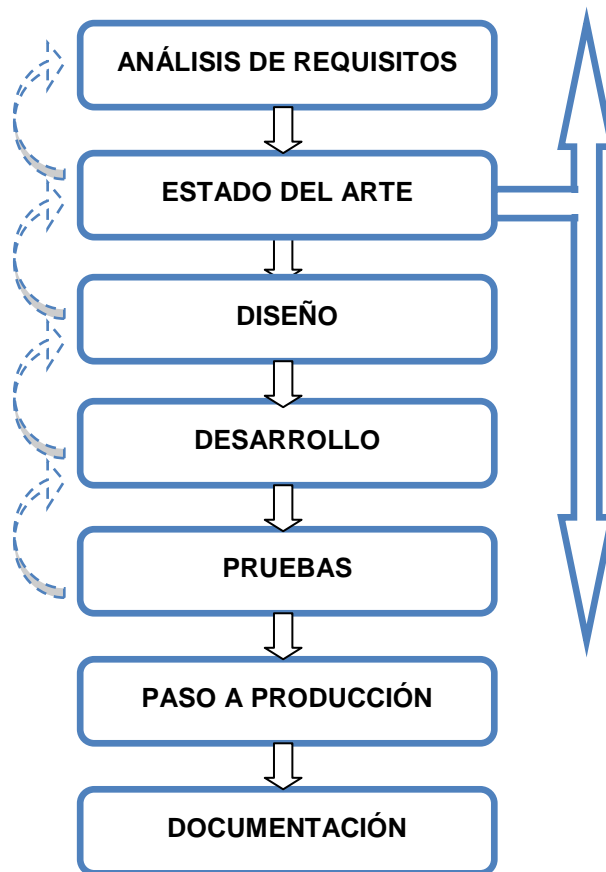


Figura 3.1: Esquema temporal de las fases de desarrollo del proyecto

3.2 FASES DEL PROYECTO

3.2.1 Análisis

En esta fase además de establecerse los objetivos y el alcance del proyecto, se realiza un estudio de los requisitos funcionales del proyecto, teniendo en cuenta el objetivo principal de la propuesta del proyecto y los requisitos necesarios de la aplicación web resultante para satisfacer las necesidades de la propuesta de proyecto y de los potenciales usuarios.

El resultado de este análisis de requisitos es el catálogo de requisitos, documento que resume todos los requisitos, y consensado y aprobado entre la dirección del proyecto y el analista. Este documento está detallado en el Capítulo 4 del presente documento.

3.2.2 Estudio del Estado del Arte

Se trata de una fase centrada especialmente en la investigación y estudio de todas las tecnologías, recursos y conceptos que pueden ser aplicables al proyecto, teniendo en cuenta los requisitos especificados anteriormente.

Es una fase que ha tenido la mayor carga de trabajo al comienzo del proyecto, debido principalmente a la necesidad de contextualizar el problema y de elegir las herramientas a utilizar para el desarrollo del proyecto.

La investigación más relevante realizada durante el estudio del Estado del Arte ha sido la referente a la Web Semántica, un nuevo paradigma para la World Wide Web, que tiene como objetivo principal estructurar el contenido de las páginas web dotándole de semántica, es decir, dotándole de la capacidad de que su significado pueda ser interpretado, en este caso por máquinas, y sea más sencilla la interoperación entre webs sin necesidad de operadores humanos.

El estudio acerca de la Web Semántica se ha basado principalmente en:

- Estudiar la motivación de crear un nuevo paradigma web mediante el estudio de la evolución de la World Wide Web.
- Comprender los conceptos que la rodean (ontología, metadatos, etc.)
- Conocer su funcionamiento interno y las formas de inter-operación entre webs.
- Estudiar los formatos, estándares y protocolos que se utilizan para la publicación y el consumo de datos adaptados a ella.
- Estudio de las tecnologías utilizables para el desarrollo del proyecto teniendo en cuenta la utilización de recursos semánticos.

Además del estudio de la Web Semántica, se ha realizado una investigación acerca de la situación actual de la Open Data o publicación de Datos Abiertos por gobiernos e instituciones públicas, focalizándose especialmente la postura de las instituciones vascas en este aspecto.

Por último, se han ido realizando durante las diversas fases del proyecto estudios de las diferentes herramientas y fuentes de datos aplicables al web-mashup a realizar, como las formas de publicación de calendarios y mapas, o los métodos de geolocalización.

El resultado y la explicación de los diferentes estudios realizados acerca del estado del arte están detallados en el Capítulo 5 de este documento.

3.2.3 Diseño de la aplicación web

Se trata de la fase en la cual, tras el análisis de los requisitos, comprender los conceptos que los rodeaban, y analizar las diferentes herramientas utilizables para la consecución del proyecto, se comienza el diseño de las diferentes funcionalidades y componentes de la aplicación web a desarrollar.

El diseño se ha focalizado en las siguientes tareas:

- Selección de fuentes de datos y formatos de publicación.
- Establecer la arquitectura de los módulos del proyecto.
- Diseñar la estructura de la base de datos.
- Diseñar las diferentes interfaces de usuario.

Todo lo referente al diseño de la aplicación web y sus componentes, está especificado en el Capítulo 6 del presente documento.

3.2.4 Desarrollo de la aplicación web

Es la fase en la que se desarrolla la parte técnica del proyecto, es decir, en la que se desarrollan los diferentes módulos especificados en el diseño realizado en la anterior etapa, con el comportamiento especificado.

Durante esta fase se realizan las tareas de:

- Concreción la forma en que se va a dar forma al diseño desde un punto de vista de viabilidad técnica: elección de herramientas para el desarrollo de la aplicación, investigación de APIs a utilizar, etc.
- Montaje y puesta en marcha de la Base de Datos.
- Programación de los diferentes módulos del proyecto.

Los detalles del proceso de desarrollo de la aplicación web se concretan en el Capítulo 7 del presente documento.

3.2.5 Pruebas

En esta fase se realizan las pruebas al prototipo desarrollado en la anterior etapa. Estas pruebas tienen como objetivo certificar que lo desarrollado cumple con los requisitos especificados y que ha sido diseñado según lo previsto.

Las pruebas analizan el funcionamiento específico de cada una de las funciones del proyecto, comprobando si existen defectos o incoherencias. En caso de que se encuentren, estas pruebas sirven para corregir errores en fases anteriores.

3.2.6 Paso a producción

Esta fase tiene como objetivo la elección de la infraestructura que va a soportar la aplicación web una vez que ha sido depurada y ha llegado a su versión final, y la realización de las tareas necesarias para la habilitación del proyecto en dicha infraestructura.

Los detalles técnicos del paso a producción del mashup semántico InnovAgenda se especifican en el Anexo II de este documento.

3.2.7 Documentación

Esta fase final tiene como objetivo la obtención del presente documento, es decir, la creación una documentación que describa el proyecto y la forma en que se ha realizado. Cabe comentar que parte de esta tarea se va realizando durante las demás fases, ya que los documentos resultantes de los trabajos realizados en las fases anteriores son incluidos en esta.

4. ANÁLISIS

En el cuarto capítulo se describirán los requisitos funcionales, analizados teniendo en cuenta los requisitos necesarios para satisfacer las necesidades de la propuesta de proyecto y de los potenciales usuarios, y consensuados entre la dirección del proyecto y el analista.

4.1 ANÁLISIS DE REQUISITOS

- La aplicación debe incluir los datos de los eventos de la web de Euskadi+Innova.
- La aplicación debe geolocalizar las direcciones de los eventos recogidos de Euskadi+Innova, y debe incluir un mapa con los eventos geolocalizados.
- La aplicación debe posibilitar la participación de los usuarios en la web mediante un sistema de comentarios para cada evento.
- La aplicación debe posibilitar la compartición de la asistencia de los usuarios en las redes sociales (Facebook y Twitter como mínimo).
- La información de cada evento debe ser publicada en formatos adaptados a la web semántica.
- La aplicación debe proporcionar una API para el acceso a los datos de los eventos.
- La agenda deberá ser exportable a otros servicios de calendario.
- La actualización y publicación de nuevos eventos en la aplicación web deben ser automáticas cuando los eventos sean publicados en la web de Euskadi+Innova.
- La aplicación debe mantener un sitio privado de administración, controlado por un sistema de usuarios y contraseñas, desde el cual poder añadir, modificar y eliminar información de la base de datos de la web de forma sencilla e intuitiva.
- El sitio privado de administración debe incluir una zona para posibilitar la traducción de forma sencilla e intuitiva.

4. ANÁLISIS

- La parte de administración del sitio solo podrá ser accesible por medio de URL que solo los administradores conozcan. Por lo tanto no deberá haber ningún acceso directo a dicha zona desde la visualización normal de la web.
- La aplicación web deberá estar adaptada a su consulta desde terminales móviles como smartphones y tablets. La única parte que no requiere de adaptación es la zona de administración, ya que solo se accederá a ella desde PCs, debido a lo delicado de su contenido.
- La aplicación y todos sus contenidos deberán estar traducidos a euskera y castellano.

5. ESTADO DEL ARTE

En esta sección se resumirán los conceptos más importantes tratados durante los diferentes estudios e investigaciones realizados. De esta manera se tratará de contextualizar el proyecto explicando con detalles los conceptos que lo rodean.

Las líneas de investigación que se han seguido y que se explicarán a continuación son: evolución de la Web hasta llegar a la Web Semántica, conceptos de la Web Semántica y sus herramientas, situación actual de la Open Data en las instituciones vascas, y el concepto de mashup y sus diferentes fuentes de datos posibles.

5.1 EVOLUCIÓN DE LA WEB: DE LA WEB 1.0 A LA WEB SEMÁNTICA

El origen de la Web data del año 1989. Fue una propuesta creada por Sir Tim Berners-Lee para la compartición de informes, planos y otros documentos en el CERN (European Organization for Nuclear Research). La revolución que incluía esta propuesta era que se trataba de un sistema de distribución de información basado en hipertexto o hipermedios enlazados mediante Internet. Para la publicación de información en la web, se diseñó también el lenguaje HTML (HyperText Mark-up Language).

La propuesta fue evolucionando, y para principios de 1993 ya había alrededor de 50 servidores para la World Wide Web, y una serie de navegadores para plataformas NeXT y X Windows, suponiendo aproximadamente el 0,1% total del tráfico de internet. A finales de ese mismo año se había multiplicado por diez el número de servidores y empezaban a publicarse nuevos navegadores para otros sistemas operativos.

A partir de 1994 comenzó el crecimiento exponencial de esta tecnología, junto con la creación de World Wide Web Consortium (W3C), una comunidad internacional impulsada por el MIT (Massachusetts Institute of Technology, una de las instituciones universitarias más importantes del mundo en el ámbito de la tecnología) y el CERN, y en la que se juntaron instituciones y público en general para trabajar conjuntamente en la creación de estándares para la Web, con el objetivo de guiar a la web hacia su máximo potencial.

Durante los primeros años de la popularización de la Web, la plataforma se basaba en un conjunto de páginas estáticas en las que editores y webmasters publicaban contenido, y que

eran visitadas por millones de personas que tenían acceso a Internet. Así pues, se consideraba la web como un lugar de “solo-lectura”. Esta es la llamada “**Web 1.0**”, que basaba su contenido en páginas web vistas como documentos estáticos que no se actualizaban, y que contenían tanto texto como imágenes. Las tecnologías básicas asociadas a esta generación son HTML y GIF (formato de imágenes). Gráficamente, se podría representar la forma de publicación y acceso a la web como algo así:



Figura 5.1: Estructura de la Web 1.0

La forma de entender la web fue evolucionando junto con su crecimiento, y el contenido de las webs dejó de ser meramente estático para añadir componentes dinámicos generados a partir de bases de datos. Tecnologías como ASP posibilitaron esto. Además, se empezó a tomar la estética de las webs como algo importante, y se empezó a usar CSS como tecnología de marcado del estilo. A este tipo de webs que publican contenido de forma dinámica y que contienen una estética más cuidada, se las engloba dentro del término “**Web 1.5**”, que supone una ligera evolución sobre el contenido totalmente estático de las Webs 1.0.

Pero el crecimiento del número de usuarios de la web, la evolución de las tecnologías de publicación web, y las nuevas necesidades de los usuarios, llevaron a Tom O'Reilly a llamar a la creación de un nuevo concepto de web, una evolución de la web del momento, y que él denominó “**Web 2.0**”, dando pie a esa nomenclatura basada en versiones para la web. La propuesta que O'Reilly presentó se basaba en siete conceptos básicos:

- Ver la web como una plataforma para el despliegue de servicios y aplicaciones.
- Aprovechar la inteligencia colectiva, es decir, posibilitar la colaboración.
- Los datos son los nuevos procesadores.
- Webs en beta-permanente, es decir, en constante cambio y evolución.
- Modelos de programación livianos.
- Software multiplataforma.
- Experiencia de usuario enriquecida.

De forma resumida, la idea de O'Reilly se basaba en, por un lado, la democratización de la web, es decir, en que todo usuario de la World Wide Web tuviera la oportunidad de crear nuevo contenido web, creando una web colaborativa e interactiva; y por otro, el empleo de nuevos métodos de distribución de la información, mejorando la experiencia de usuario, y posibilitando la reutilización de información.

Desde ese momento se empezó a ver la web como un sitio colaborativo e interactivo, y se empezaron a adaptar nuevas tecnologías que proporcionaran a las webs de esas capacidades. Estas tecnologías que rodean la web 2.0 son las siguientes:

- HTML dinámico, o DHTML, que designa un conjunto de técnicas para crear sitios web interactivos combinando HTML, JavaScript (lenguaje interpretado para la parte cliente) y hojas de estilo CSS.
- XML, un formato sencillo para el guardado de información, ideal para las nuevas formas de distribución del contenido.
- AJAX, que da la posibilidad de que una web no necesite de su recarga para la comunicación entre cliente y servidor, lo cual permite la creación de aplicaciones web ágiles y dinámicas.

A partir de esta nueva forma de ver la web, la estructura de la relación entre las páginas y los usuarios cambia, rompiendo drásticamente con la de la Web 1.0:

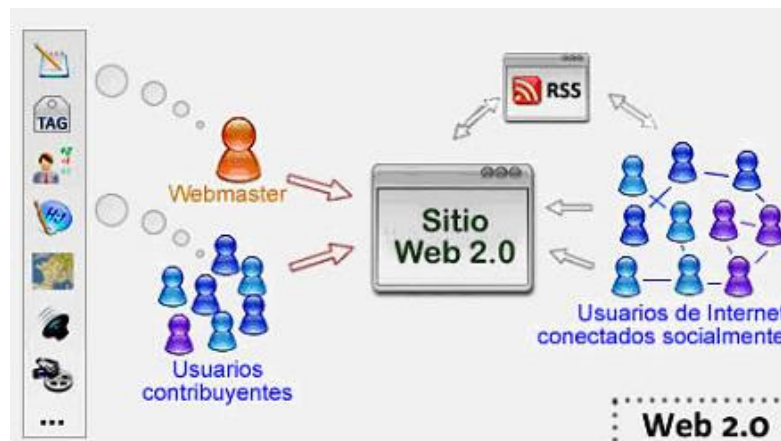


Figura 5.2: Estructura de la Web 2.0

Esta evolución creó una serie de nuevas herramientas: blogs, wikis, podcasts, RSS, redes sociales... Es decir, toda una serie de nuevos servicios, orientados a crear inteligencia colectiva y a socializar la web. Esta nueva visión de la web ha supuesto un crecimiento exponencial de los contenidos en Internet, llegando a convertir la web en una gran base de datos de conocimiento colectivo.

Pero la tendencia evolutiva de la web no para ahí, ya que en medio de este panorama tan favorable en el que una comunidad ingente de usuarios interconectados trabajan

conjuntamente para la publicación de nuevo contenido, existen limitaciones. La limitación más importante es, a la vez, una de las mayores virtudes de la web, y es el enorme tamaño que ha alcanzado. El hecho de que haya tanto contenido, hace que algunas tareas que, por ejemplo, se basen en la búsqueda y comparación de diferentes contenidos de la web, requieran un tiempo excesivo para una persona, o simplemente resulten inabarcables. Incluso desarrollar software capaz de realizar estas tareas es enormemente complicado, debido a que la mayoría del contenido no está adaptado a su comprensión por parte de máquinas, y por tanto, resulta imposible que un software logre comprender y comparar el significado de un contenido dado.

Esta reflexión tiene como resultado el descubrimiento de un problema común para la mayoría del contenido de internet: está expresado en formatos e interfaces comprensibles únicamente por personas, pero no por máquinas.

Cómo solución a este problema se plantea el concepto de “**Web semántica**”, es decir, la siguiente evolución de la web, en la que el contenido de una página está expresado de tal manera que sea comprensible tanto por operadores humanos como por máquinas. Hay quien denomina a este nuevo paradigma la “Web 3.0”, porque supone una nueva evolución del concepto de Web, pero debido a que existe un debate considerable acerca de este término –ya que hay muchos que defienden que la web semántica no es la única evolución posible de la web-, en este documento se referirá a esta evolución por su nombre más descriptivo, “web semántica”.

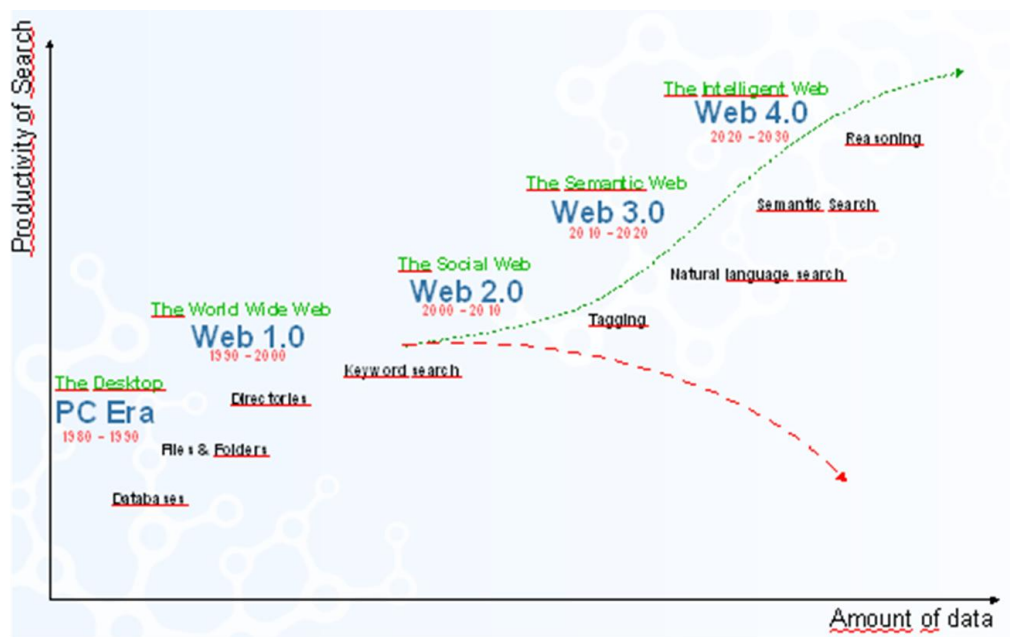


Figura 5.3: Mapa de la evolución de la web

5.2 WEB SEMÁNTICA

La Web Semántica es una Web extendida, dotada de mayor significado, gracias a la cual se pueden obtener soluciones a problemas habituales en la búsqueda de información en la web, motivados principalmente por el tamaño ingente y la naturaleza heterogénea del contenido actualmente publicado en Internet. Esto se realiza gracias a la utilización de una infraestructura común, mediante la cual, es posible compartir, procesar y transferir información de forma sencilla. Esta Web extendida y basada en el significado, se apoya en lenguajes universales que resuelven los problemas ocasionados por una Web carente de semántica en la que, en ocasiones, el acceso a la información se convierte en una tarea difícil y frustrante.

El objetivo principal es mejorar Internet ampliando la interoperabilidad entre los sistemas informáticos sin requerir la actuación de operadores humanos.

En resumen, se trata de un paradigma que se basa en la publicación de datos definidos y vinculados de tal manera que se habilite la creación de agentes inteligentes que recuperen y manipulen la información de forma autónoma y sin necesidad de operadores humanos.

Para obtener esa adecuada definición de los datos, la Web Semántica utiliza esencialmente RDF, SPARQL, y OWL, mecanismos que ayudan a convertir la Web en una infraestructura global en la que es posible compartir, y reutilizar datos y documentos entre diferentes tipos de usuarios.

- RDF proporciona información descriptiva simple sobre los recursos que se encuentran en la Web y que se utiliza, por ejemplo, en catálogos de libros, directorios, colecciones personales de música, fotos, eventos, etc.
- SPARQL es lenguaje de consulta sobre RDF, que permite hacer búsquedas sobre los recursos de la Web Semántica utilizando distintas fuentes de datos.
- OWL es un mecanismo para desarrollar temas o vocabularios específicos en los que asociar esos recursos. Lo que hace OWL es proporcionar un lenguaje para definir ontologías estructuradas que pueden ser utilizadas a través de diferentes sistemas. Las ontologías, que se encargan de definir los términos utilizados para describir y representar un área de conocimiento, son utilizadas por los usuarios, las bases de datos y las aplicaciones que necesitan compartir información específica. Las ontologías incluyen definiciones de conceptos básicos en un campo determinado y la relación entre ellos.

Es decir, RDF sirve para definir los datos a publicar, OWL describe propiedades y relaciones de dichos datos, y SPARQL sirve para realizar consultas sobre lo publicado.

Los RDF están formados en forma de grafo. Y estos grafos constan de tres partes: Sujeto, Predicado y Objeto. El sujeto es la entidad a definir, y el objeto corresponde a un atributo de dicho sujeto, el predicado está definido por OWL, y se trata de la relación de propiedad entre el sujeto y el objeto.

Para visualizar esta forma de definir los datos, se adhiere un pequeño ejemplo de código para la definición de datos en RDF:

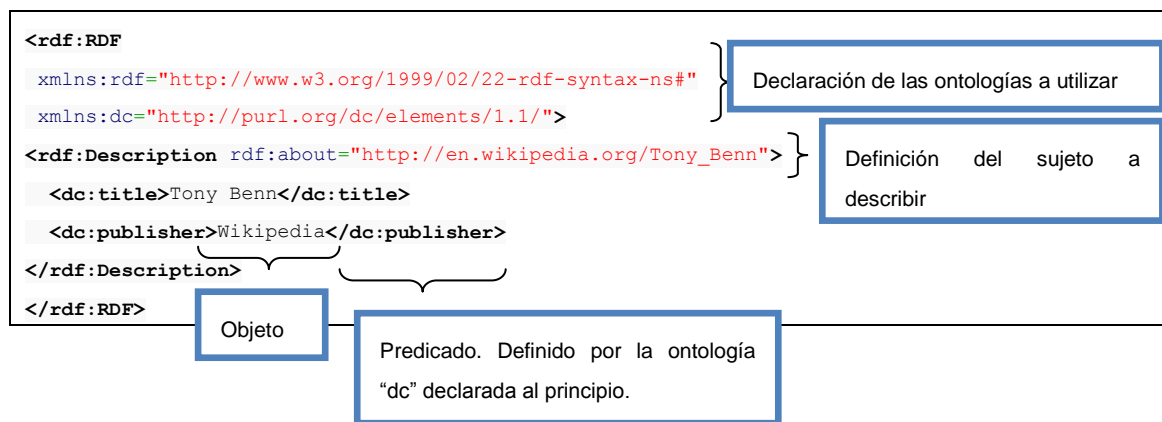


Figura 5.4: Estructura de un RDF

El grafo del RDF representado arriba estaría describiendo al sujeto “http://en.wikipedia.org/Tony_Benn”, que en este caso se trata de una URL. De este sujeto se definen dos propiedades: Title y Publisher. Title y Publisher son dos predicados definidos en la ontología “http://purl.org/dc/elements/1.1/” (que habrá sido definida mediante OWL). Los objetos de este grafo, serán “Tony Benn” y “Wikipedia”, es decir, los valores que toma cada una de las propiedades.

Para publicar contenido semántico en webs, existen dos formas:

- Exportación de los datos a documentos en notaciones de representación RDF.
- Marcado dentro de los HTML de las páginas web que contienen los datos.

Para la primera opción existen diversos formatos de publicación, basados en diferentes notaciones, como N3, N-Triples, Turtle o la más utilizada y generalizada, RDF+XML. Se trata de diferentes notaciones para expresar los mismos grafos.

Para la segunda opción, en cambio, existen tres opciones, de las cuales tan solo una hace uso de RDF. Son las siguientes:

- **RDFa**

RDFa es un conjunto de extensiones de XHTML propuesto por la W3C para la inclusión de semántica en el marcado HTML. Añade una serie de atributos que permiten la declaración de las ontologías a utilizar, así como la definición de los sujetos, predicados y objetos, de igual manera que se hace en los documentos RDF+XML, por ejemplo.

- **Microformatos**

Los microformatos son convenciones sencillas (conocidas como entidades) que se utilizan en las páginas web para describir un tipo de información específico como, por ejemplo, una opinión, un evento, un producto, una empresa o una persona. Cada entidad tiene sus propias propiedades. Para ello, existen una serie de ontologías definidas llamadas microformatos que especifican propiedades para ciertas entidades definidas; ejemplo de ello son “hCard”, “hAtom” o “hCalendar”. Para el marcado se hace uso de las propiedades HTML “class”, “rev” y “rel”.

- **HTML5 Microdata**

Se trata de una especificación de HTML5 para el marcado de contenido semántico en webs. Mediante esta especificación se trata de simplificar el marcado semántico, con la inclusión de una serie de atributos sencillos para describir entidades y propiedades. Existen diversos vocabularios de microdatos que especifican las relaciones para objetos comunes (eventos, personas...). Google, Yahoo y Microsoft han creado un vocabulario para esta especificación llamado Schema.org, que pretenden que se extienda, ya que es soportado por sus tres grandes buscadores, y se presenta como una ayuda al posicionamiento y a la habilitación de búsquedas semánticas.

Una forma de crear ficheros RDF a partir de estos marcados basados en XHTML, es GRDDL, otra de las tecnologías que forman parte del ecosistema de la web semántica. GRDDL es un formato de marcado para archivos basados en XML, como XHTML. Con unas sencillas anotaciones en el documento XHTML, se le dota al documento de la capacidad de transformación de su contenido a RDF. Soporta la transformación de documentos XHTML con marcado basado en microformatos y RDFa.

Como se puede observar, existen multitud de estándares, herramientas y formatos para la publicación de contenido semántico en la web. El objetivo de todo ello es facilitar la adaptación de contenido pasado, así como la creación de contenido ya adaptado.

Si se consiguiera el objetivo de que se normalizara la publicación de contenido semántico, es decir, se consiguiera que la mayoría del contenido de la World Wide Web estuviera publicada de forma entendible para máquinas, el siguiente reto en este sentido sería el asentamiento de la Web como una red de Linked Data, aprovechando las posibilidades de interlinkeo de los formatos semánticos. Con el término Linked Data se hace referencia a la adición de links a contenidos parecidos o relacionados dentro del

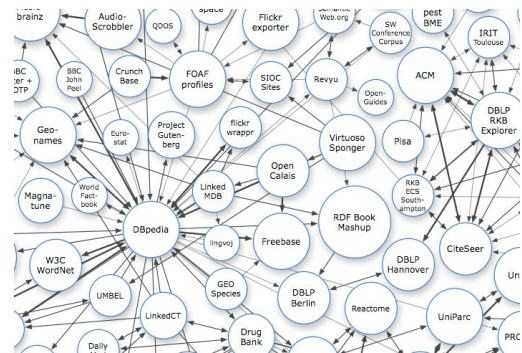


Figura 5.5: Imagen conceptual de la web vista como una red de linked data

ámbito de publicación de contenido semántico. De esta manera, sería posible la adición de técnicas de descubrimiento automático de conceptos relacionados a los agentes inteligentes antes mencionados.

5.3 OPEN DATA EN LA COMUNIDAD AUTÓNOMA VASCA

El término “Open Data” o “Datos Abiertos”, en el entorno de los organismos públicos y gubernamentales, se refiere a la publicación de datos de dominio público por parte instituciones públicas, de forma que estos datos además de visibles, sean reutilizables por los ciudadanos.

Para cumplir con la premisa de que estos datos sean considerados como reutilizables, las webs que publican la información de dominio público deben tener mecanismos de acceso programático a dichos datos, como APIs o Web-Services, o bien mediante la publicación de dichos datos en formatos de web semántica.

El Gobierno Vasco se ha caracterizado en los últimos años por publicar multitud de páginas web relacionadas con los diferentes departamentos y servicios sobre los que trabaja. En ellos se publica mucha información de dominio público, pero el problema común de estas webs suele radicar en su mala estructuración, lo cual convierte en complicado acceder de forma deseable a toda esa información. Por lo tanto, aunque la propuesta de publicación en internet de esa información de dominio público era buena en un inicio, la realidad ha demostrado que no está del todo bien llevada a cabo. Además, no se ofrecen métodos para su reutilización de forma programática, por lo que no podría considerarse open data tal y como se ha definido.

A pesar de que en un principio se podría pensar que el open data en las instituciones vascas no existe, en 2010 se empezaron a dar cambios en este sentido. El Gobierno Vasco, a razón de la gran demanda y a que muchos gobiernos importantes (como el británico o el estadounidense) se habían unido a alternativas de este tipo, decidió llevar a cabo un proyecto de apertura de datos de dominio público siguiendo el modelo de otros gobiernos. Para dicha causa, se habilitó el portal <http://opendata.euskadi.net/>.

Open Data Euskadi es el portal de acceso a dichos datos públicos, que se publican en distintos formatos, incluido RDF, y bajo licencias de propiedad abiertas. En esta plataforma se puede acceder a cientos de ficheros de datos que se clasifican en torno a diferentes categorías, como política, economía, salud, meteorología, medioambiente, urbanismo, ocio, etc. Por el momento consta de pocos datos, pero se trata de un sitio que está en crecimiento, y que va incorporando datos poco a poco.

Según la propia web, los objetivos de Open Data Euskadi son la generación de valor y riqueza gracias a los productos derivados de los datos abiertos, la transparencia de la administración pública, la interoperabilidad entre administraciones, y la ordenación interna de la información de la administración.

En conclusión, se puede decir que la situación del movimiento de datos abiertos en la Comunidad Autónoma Vasca es de crecimiento, se trata de una iniciativa joven, y que por el momento consta de pocos datos, pero que está en constante actualización. Aun así, no se puede dejar de comentar que aunque iniciativas como Open Data Euskadi son positivas, muchas de las páginas web de las instituciones vascas dejan mucho que desear en cuanto a estructura y accesibilidad.

5.4 MASHUPS

Un mashup es una aplicación web que usa y combina datos, presentaciones y funcionalidad procedentes de una o más fuentes para crear nuevos servicios. El término implica integración fácil y rápida, usando a menudo APIs abiertos y fuentes de datos para producir resultados enriquecidos que no fueron la razón original para la que fueron producidos los datos en crudo originales.

La arquitectura de los mashups está siempre compuesta de tres partes:

- El proveedor de contenidos, es decir, las diferentes fuentes de datos.
- El sitio mashup, que es la nueva aplicación web que provee un nuevo servicio utilizando diferente información y de la que no es dueña.
- El navegador del cliente, que es básicamente la interfaz de usuario del mashup. En muchas ocasiones las APIs de los servicios son accesibles vía lenguajes de programación del lado del cliente, como JavaScript.

El contenido usado en mashups puede provenir de diferentes orígenes, como APIs, Web Feeds como RSS o Atom, o técnicas de screen scrapping –parseo del contenido de las webs directamente accediendo a los HTMLs, es decir, acceso a los datos de los sitios que no proveen ni APIs ni sindicación-.

De entre estas fuentes de datos, las más utilizadas son las APIs. Una API es un conjunto de protocolos, rutinas y convenciones que definen la forma en la que se puede invocar una determinada función de un programa desde cualquier otra aplicación. Su función principal es la de proveer un canal de comunicación para que las aplicaciones puedan trabajar unas con otras. Como ejemplos de APIs adaptables a su uso en mashups tenemos la API JavaScript de Google Maps, que permite la adición de mapas a los sitios, la de Facebook, que permite añadir aspectos sociales a los mashups, o la de Flickr, que permite añadir a los mashups fotos alojadas en dicho servicio.

En resumen, los mashups son aplicaciones web que basan su contenido en la información que publican otros servicios web, y que apuestan por el ensamblado, en lugar de por la codificación, es decir, se centran en la utilización de servicios ya existentes, evitando la creación de todo desde cero.

6. DISEÑO

En este capítulo se concretará la estructura interna del proyecto, es decir, se detallará la arquitectura en la que la aplicación web se basa, así como los diseños de interfaces de usuario y de la base de datos.

6.1 SELECCIÓN DE FUENTES DE DATOS

Tras el análisis de las diferentes herramientas utilizables para la consecución del proyecto, se decide que las fuentes de datos para el desarrollo del mashup sean:

- Euskadi+Innova como fuente principal de información, de la cual se deberá recoger toda la información referente a los eventos.
- Google Maps como proveedor de mapas.
 - Razones para su elección:
 - Servicio maduro y utilizado por muchas webs.
 - Información geográfica muy actualizada.
 - API basada en JavaScript fácil de implementar y personalizar.
 - Documentación muy completa.
 - Adaptable a su visualización desde web móvil.
- Google Maps como servicio de geolocalización.
 - Razones para su elección:
 - Servicio de geolocalización basado en servicio web.
 - Respuesta obtenible en varios formatos.
 - Documentación completa.

- Gran variedad de parámetros de búsqueda.

6.2 SELECCIÓN DE FORMATOS DE PUBLICACIÓN

Teniendo en cuenta los requisitos especificados, son necesarias dos decisiones a la hora de diseñar la forma de publicar el contenido: el publicador para el calendario, y la forma en la que publicar la información adaptada a la web semántica.

Para el calendario se ha decidido mantener la publicación de dos calendarios basados en Google Calendar, uno para la información publicada en euskera y otro para la información publicada en castellano.

- Razones para la elección de Google Calendar como herramienta de publicación de calendarios:
 - Calendario con una gran base de usuarios.
 - Provee diferentes formas de exportación de calendarios.
 - Exportación de eventos individuales.
 - API de lectura y publicación de eventos disponible en varios lenguajes de programación.
 - Diversas opciones de publicación de eventos individuales y de calendarios.

En cuanto a los formatos de publicación de datos semánticos, se toma la decisión de publicar en diferentes formatos, para probar las diferentes formas de publicación. Para ello, se decide publicar el contenido con marcado semántico integrado en los HTML –para mejorar el posicionamiento en buscadores (SEO) y habilitar-, y que se dé la opción de exportar dicho contenido a otros formatos. Además, se toma la decisión de adaptar la API especificada en los requisitos del sistema a la web semántica.

- Para el marcado HTML se decide publicar en dos formatos diferentes: RDFa y Micro Datos HTML5 basado en Schema.org.
 - Razones para utilizar Schema.org:
 - Marcado creado y utilizado por los principales buscadores de internet: Google, Yahoo y Microsoft Bing.
 - Marcado sencillo y completo.
 - Razones para utilizar RDFa:
 - Marcado más maduro y estándar.

- Marcado basado en RDF.
- Para la exportación del contenido publicado, se elige dar la opción de exportar en varios formatos: RDF/XML, N-Triples, Turtle y N3, para dar la opción al usuario de obtener la información en el formato que decida.
- Para la publicación de datos semánticos en la API, se elige publicar en RDF+XML, el formato más utilizado basado en el estándar RDF.

En cuanto a las ontologías utilizables para la publicación de los datos utilizando RDF, existían dos alternativas: utilizar ontologías ya desarrolladas y utilizadas, o desarrollar una nueva ontología utilizando OWL.

Debido a que las ontologías RDF Calendar (<http://www.w3.org/TR/rdfcal/>) –ontología para la definición de eventos de un calendario creada por la World Wide Web Consortium (W3C) para un marcado RDF de eventos compatible con iCalendar- para la publicación de eventos, y WGS84 Geo (www.w3.org/2003/01/geo/wgs84_pos) –ontología creada por el W3C para el marcado RDF de posiciones geolocalizadas- cumplen con los requisitos de publicación para este proyecto, se toma la decisión de utilizar dichas ontologías.

6.3 ARQUITECTURA

Teniendo en cuenta la diversidad de fuentes de datos y métodos de publicación, se toma la decisión de estructurar la arquitectura de forma modular. Los módulos serán de dos tipos: dependientes de la intervención de los usuarios y módulos de ejecución en servidor.

Los módulos que se ejecutan cuando los usuarios lo requieren son básicamente los módulos de publicación o modificación del contenido. Estos módulos se basarán en una arquitectura que siga el **paradigma de Modelo-Vista-Controlador (MVC)**, es decir, que en ellos se debe separar la representación de la base de datos (modelo), la interfaz de usuario (vista) y la parte de lógica de programación (controlador).

Se trata de los siguientes:

- Módulo de publicación de contenido, que a su vez contiene varios sub-módulos que se ejecutan según el contenido que el usuario desea visualizar: módulo de publicación de mapas, de publicación de calendarios, de publicación de los eventos en formato para la web semántica, etc.
- Módulo de administración.
- Módulo de la API.

Los módulos de ejecución en servidor, que son los que se ejecutan en el servidor a intervalos regulares y de forma automática, y los encargados de introducir nueva información en la base de datos. Son los siguientes:

- Módulo de obtención de datos de Euskadi+Innova.
- Módulo de geolocalización de eventos.
- Módulo de actualización de Google Calendar.
- Módulo de comprobación de eventos finalizados.

La siguiente ilustración muestra de forma general la arquitectura modular definida anteriormente:



Figura 6.1: Arquitectura general de la aplicación web

A continuación se detallarán las funciones y las actividades que cada uno de los módulos debe llevar a cabo, es decir, la parte “**Controlador**” de los módulos que siguen el paradigma MVC:

6.3.1 Módulo de obtención de datos de Euskadi+Innova

Para la obtención de los datos de los eventos que se publican en www.euskadinnova.net, ha sido necesario un análisis de la estructura de la página web, así como de sus formas de publicación de información.

La web de Euskadi+Innova está dotada de sindicación RSS para el acceso a noticias, agenda, entrevistas, vídeos y ayudas de cada una de las temáticas de las que informa (Innovación Tecnológica, Innovación Social, Transformación Empresarial y Empresa Digital). Se puede acceder a esta sindicación tanto en euskera como en castellano.

Los RSS de los eventos no ofrecen más información que el título, el resumen y la URL de los nuevos eventos. Por lo tanto, además de la información que el RSS ofrece, es necesaria la recogida de más información acerca del evento. Para ello, es necesario un proceso de investigación de la estructura de los HTML en los que se publica la información de cada evento, para su posterior parseo.

Por lo tanto, cuando se ejecute, este módulo se debe encargar de la conexión a cada uno de los RSS de eventos de cada temática, de la comprobación de si los eventos han sido recogidos ya, así como del parseo del HTML de cada evento, tanto en euskera como en castellano.

Una vez obtenida toda la información, este módulo es el encargado de ejecutar los módulos de geolocalización y de actualización de Google Calendar, con el objetivo de terminar de obtener toda la información que se desea guardar acerca del evento en cuestión.

Así pues, el diagrama de flujo resultante para este módulo sería el descrito en la siguiente página:

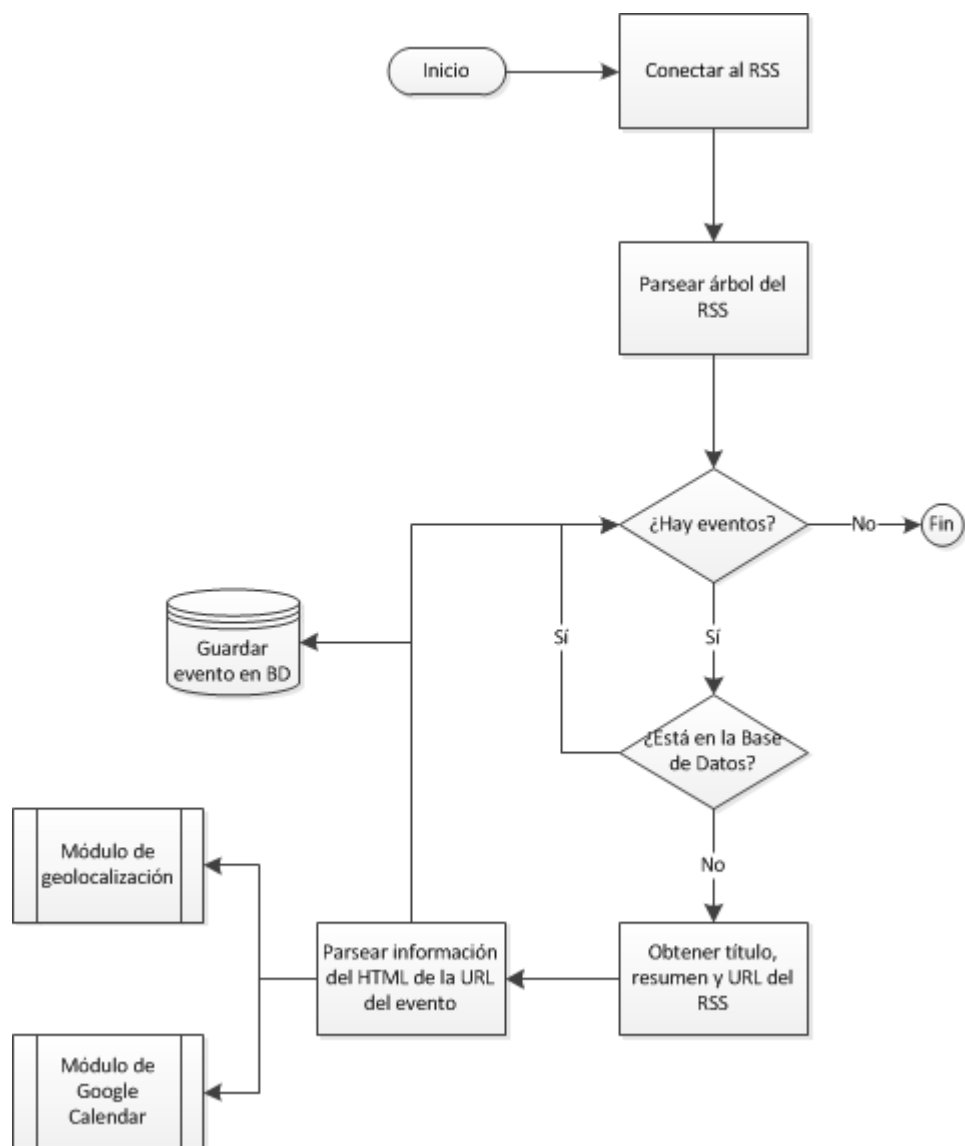


Figura 6.2: Diagrama de flujo del módulo de obtención de datos de Euskadi+Innova

6.3.2 Módulo de geolocalización de eventos

La web de Euskadi+Innova publica la localización de los eventos sin ningún tipo de formato definido y sin geolocalizar (sin especificar sus coordenadas), de hecho, la mayoría de las ocasiones se publica incluyendo nombres de lugares concretos y edificios de empresas, y no direcciones completas. Tras analizar varias de las localizaciones, se ha establecido que muchos lugares aparecían en repetidas ocasiones. Teniendo en cuenta lo complicado de automatizar la geolocalización de direcciones incompletas o de lugares muy concretos, se ha decidido la creación de una pequeña tabla en la base de datos con los lugares más comunes.

Estos lugares son los Parques Tecnológicos de Bizkaia, Gipuzkoa y Araba, las oficinas de SPRI (Sociedad Para la Transformación Competitiva), las oficinas de Empresa Digitala en Vitoria-Gasteiz, Zamudio, Donostia-San Sebastián y Arrasate-Mondragón, y el Polo de Innovación Garaia.

Así pues, el módulo de geolocalización lo primero que deberá hacer será buscar coincidencias entre la localización obtenida y los lugares más comunes. En caso de no encontrar ninguna coincidencia, se pasará a la búsqueda con el Webservice proporcionado por la API de Google Maps. La búsqueda con el Webservice de geolocalización se realizará introduciendo la dirección completa obtenida de Euskadi+Innova. En caso de error, la última alternativa es comprobar si la dirección incluye el código postal (muchas de ellas lo incluyen), y en caso afirmativo, comprobar las coordenadas geográficas asociadas a dicho código postal mediante la API de Google Maps de nuevo.

En caso de no conseguir la geolocalización automática del evento, se debe mandar una notificación a los usuarios administradores de la aplicación, es decir, a los usuarios que tienen acceso a la zona de administración del sitio.

Así pues, el diagrama de flujo resultante para este módulo sería el que se especifica en la página siguiente:

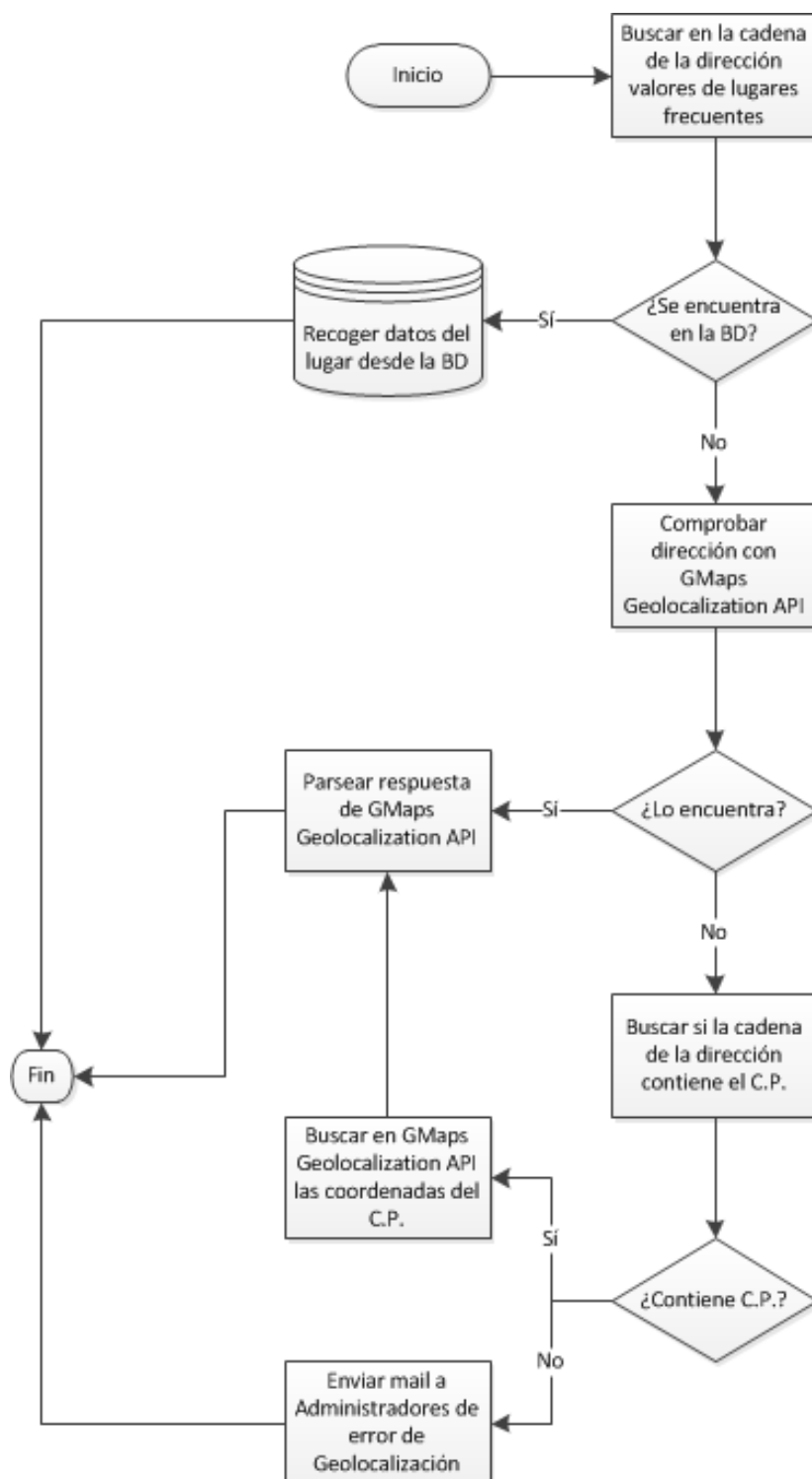


Figura 6.3: Diagrama de flujo del módulo de geolocalización automática

6.3.3 Módulo de actualización de Google Calendar

La actualización de cada uno de los calendarios de Google Calendar con la información de los eventos se realizará gracias a la API de Google Data, que ofrece manejo de calendarios asociados a las cuentas de Google.

Mediante esta API, el módulo se conectará con la cuenta Google de InnovAgenda, y subirá la información en cada uno de los idiomas a cada calendario, previa adaptación de dicha información a formatos legibles por Google Calendar.

Por lo tanto, el diagrama de flujo de este módulo será el siguiente:

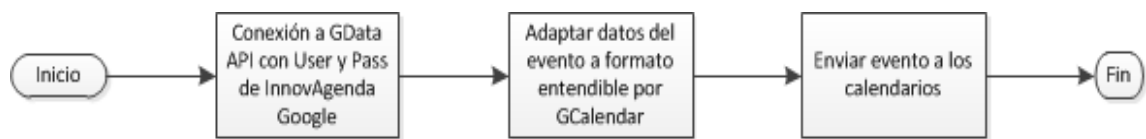


Figura 6.4: Diagrama de flujo del módulo de actualización de Google Calendar

6.3.4 Módulo de comprobación de comprobación de eventos finalizados

Este módulo es el encargado de comprobar si alguno de los eventos de la base de datos ha finalizado ya. Para ello, cuando se ejecute, deberá acceder a todos los eventos de la base de datos que aún no hayan finalizado (habrá un valor booleano en la tabla que lo indique), y en caso de encontrar alguno cuya fecha y hora de finalización sean anteriores a la fecha actual, lo marca como finalizado.

Así pues, el diagrama de flujo es el siguiente:

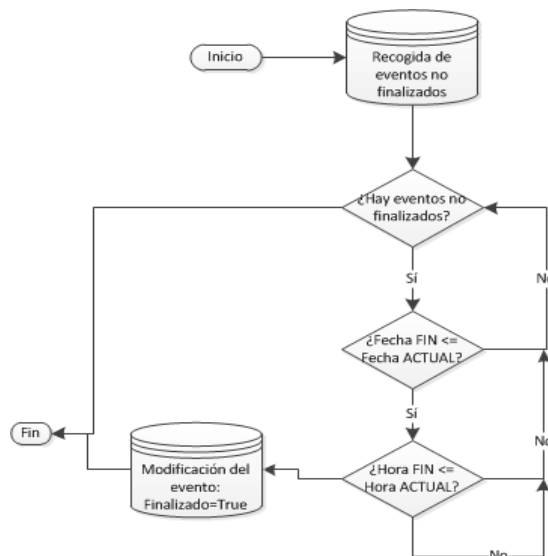


Figura 6.5: Diagrama de flujo del módulo de comprobación de eventos finalizados

6.3.5 Módulo de publicación

El módulo de publicación de contenido web es el encargado de publicar la información al usuario según lo requiera en cada momento. Su tarea principal es descubrir desde qué tipo de plataforma accede el usuario a la web, para adaptar la visualización a sus necesidades.

De manera que hasta el momento su diagrama de flujo sería el siguiente:

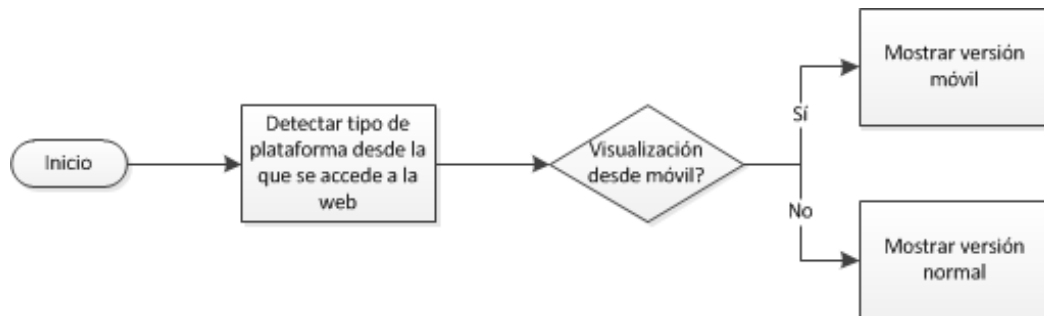


Figura 6.6: Diagrama de flujo básico del módulo de publicación de contenido

Una vez mostrada la visualización adecuada, se ejecutará cada sub-módulo según lo requiera el usuario, siendo los sub-módulos las diferentes visualizaciones de las secciones de las que consta la web: visualización del mapa, listado de eventos, publicación de los calendarios, módulo de búsqueda de eventos, vista de contacto o feedback y publicación de eventos individuales en formato semántico.

6.3.6 Módulo de administración

El módulo de administración del sitio es el encargado de posibilitar la gestión de la información del sitio a administradores del sistema. El módulo deberá mantener un sistema de acceso basado en usuarios y permisos. Por lo tanto, deberá encargarse de verificar el acceso de cada usuario a lo que sus permisos le permiten. Estos permisos los podrán modificar los administradores del sitio desde el propio módulo de administración.

Desde el módulo, se permitirán las operaciones de alta, baja y modificación de elementos de la Base de Datos, además de permitirse el acceso a la zona de traducción, desde la cual podrán traducir los elementos del sitio de forma sencilla.

6.3.7 Módulo de API

El módulo de la API debe ser el que habilite la obtención de datos siguiendo la metodología REST. Así pues, el módulo debe mantener una serie de URIs para acceder a eventos concretos, y, a su vez, ser capaz de realizar búsquedas más completas utilizando los campos de nombre, fecha de inicio y fecha de fin mediante la recepción de atributos de búsqueda pasados vía HTTP GET. Además de ello, deberá mantener 2 atributos de HTTP GET especiales: uno de selección de lenguaje y otro de selección de formato de salida.

La salida del módulo debe proporcionarse en los siguientes formatos: XML, JSON, YAML y RDF+XML. La estructura en los ficheros XML, JSON y YAML debe ser la misma, es decir, deberán tener los mismos nombres para los campos. La estructura del RDF+XML se define a partir de las ontologías antes mencionadas, RDF Calendar y WGS84 Geo.

Así pues, el diagrama de flujo será el siguiente:

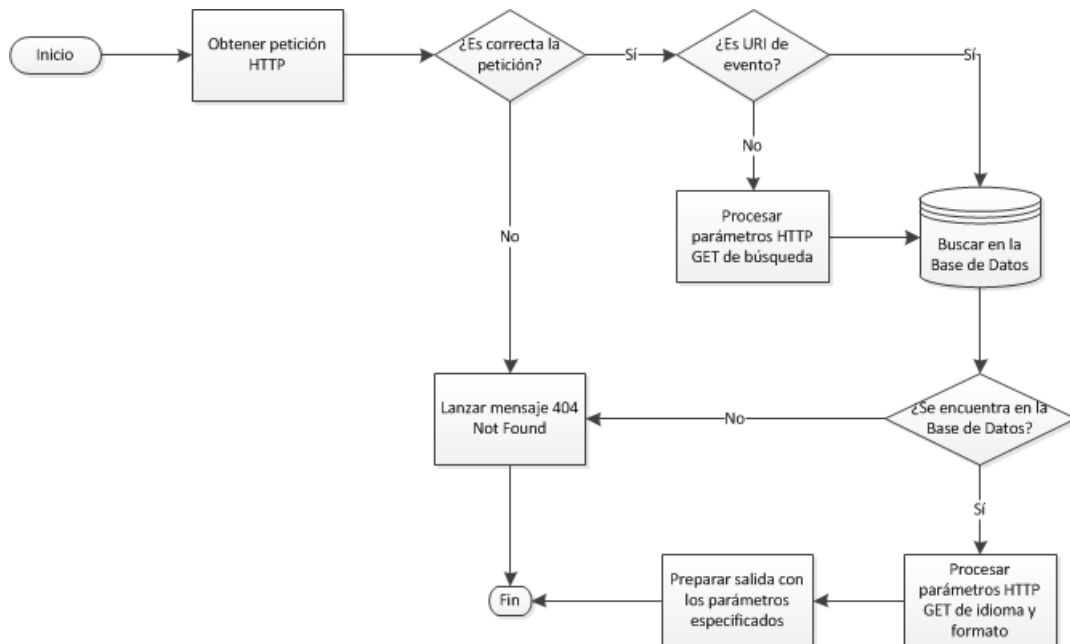


Figura 6.7: Diagrama de flujo del módulo de API

6.4 DISEÑO DE LA INTERACCIÓN CON EL USUARIO

Los módulos que requieren la interacción del usuario, como se ha comentado con anterioridad, son el módulo de publicación, el módulo de administración y el módulo de API. En este punto se describirá la parte de “**Vista**” de los módulos MVC.

La interacción del usuario de los primeros dos módulos es completamente visual, y por tanto se ha diseñado la interacción tratando de conseguir una interfaz de usuario agradable y sencilla de utilizar. Además, la interacción del usuario con el módulo de publicación ha requerido el diseño de una interfaz adaptada a su visualización desde terminales móviles.

La interacción del usuario del módulo de API se basa en peticiones HTTP, por lo que el principal objetivo ha sido el diseño de un sistema de URLs sencillo de entender y manejar, y que siga el paradigma REST.

6.4.1 Interacción con la versión de escritorio del módulo de publicación

El módulo de publicación es el centro de la interfaz del usuario de la web. En ella se tiene acceso a la publicación de todos los contenidos. La versión de escritorio de este módulo está dividida en varios sub-módulos, que son las diferentes partes de la interfaz de InnovAgenda. Cada una de estas partes se podrá acceder desde las demás con un sistema de pestañas visible en todas ellas. Los sub-módulos son los siguientes:

Mapa

La primera interfaz que se verá tras acceder a la aplicación web será la del mapa, en la que se incluye la visualización del mapa de Google Maps. Este mapa aparece personalizado con la inclusión de marcadores de los 20 eventos más próximos. Además, el mapa puede aparecer centrado en la localización del usuario, si así lo requiere, para encontrar de forma más sencilla los eventos más cercanos a su posición.

Además del mapa, al ser la primera interfaz de la web, esta vista incluye el mensaje de bienvenida a la web, en la que se explican los contenidos y las diferentes partes que incluye.

Lista de eventos

Se trata de la vista de la lista de todos los eventos que aún no han finalizado. Se paginarán dichos eventos de siete en siete. De ellos se mostrará el título, la fecha de inicio y el tipo de evento, y serán puntos de acceso a la visualización única de cada evento.

Calendario

Se trata de la vista en la que se mostrará el calendario de Google Calendar, con vistas por semanas, mes o lista de eventos. En ella se explicarán también los pasos a seguir para la exportación de dicho calendario.

Buscador

La vista del buscador se compone de un formulario de búsqueda, y de un área en el que se mostrarán los resultados. El formulario permitirá la búsqueda de eventos basada en diversos campos: identificativo, título, fechas de inicio y fin, precio, idioma de impartición del curso, provincia, finalizado o no, tipo de evento, temática del evento, agencia coordinadora y agencia organizadora.

Los resultados de la búsqueda se mostrarán en el campo de resultados, en el que aparecerán los eventos encontrados paginados de 5 en 5.

Contacto

Es la interfaz para el contacto con los administradores de la web. En ella se encontrará la dirección de email de contacto así como de un pequeño formulario para facilitar el contacto. El resultado del envío del formulario será el envío de un email a los administradores, y la muestra de un mensaje de agradecimiento por el feedback.

Documentación de la API

Se trata de la interfaz en la que se muestra la documentación de la API de acceso a eventos. En ella se mostrará un texto con toda la explicación de cómo utilizar la API, y una serie de ejemplos que muestren su uso y los diferentes resultados que puede ofrecer.

Evento

Se trata de la interfaz más completa de toda la interfaz de usuario, y la que consta de más componentes. Habrá dos visualizaciones posibles de esta vista, una utilizando marcado semántico basado en Microdatos HTML5, y otra basada en RDFa. Esta diferencia de marcado no afectará en la interfaz visible para el usuario. Ambas vistas constarán de dos partes diferenciadas, una superior y otra inferior:

En la superior se mostrará toda la información disponible acerca del evento, marcada internamente de la forma descrita. Dentro de esta información se incluirá también un mapa estático de la localización geográfica. En esta misma parte se incluirá un recuadro con las diferentes opciones de exportación del contenido, tanto a formatos semánticos, como su posibilidad de exportación a Google Calendar.

La parte inferior constará de dos partes. Una de ellas contendrá el sistema de comentarios que permitirá la participación activa de los usuarios en la web. La otra contendrá las opciones de compartición de la asistencia a los eventos, en la que se utilizará la API de Facebook, Twitter y Google+ para la compartición en dichas redes sociales.

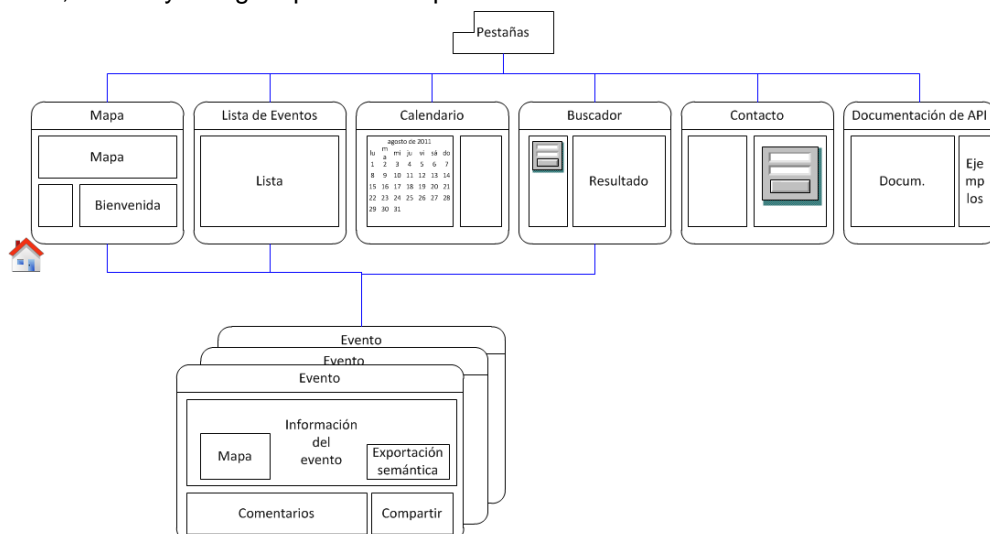


Figura 6.8: Mapa conceptual de la navegación de diálogos de la versión de escritorio del módulo de publicación

6.4.2 Interacción con la versión móvil del módulo de publicación

La versión móvil de este módulo, aunque no en contenido, sí que difiere con su versión de escritorio en cuanto al diseño de la interfaz de usuario. A la hora de diseñar la versión móvil de este módulo, se debe tener en cuenta que los elementos de la interfaz deben ser suficientemente grandes para habilitar la navegación táctil, y que estos elementos deben adaptarse a su visualización desde diferentes resoluciones.

La versión móvil, al igual que la de escritorio, está dividida en varios sub-módulos. Los módulos de lista de eventos y buscador mantendrán su función, aunque se adaptarán su contenido a elementos que permitan su manejo desde una interfaz móvil, mientras que las vistas de la documentación de la API y de contacto no serán incluida. Los sub-módulos cuyo cambio es más significativo son los siguientes:

Bienvenida

La versión móvil no tendrá su pantalla de bienvenida en la vista del mapa, sino que habrá una visualización exclusiva para esta labor. En ésta, además del mensaje de bienvenida, se mostrará un menú formado por botones que darán acceso al resto de partes de la web. Además, se deberá mostrar un link para ver la versión completa de la web.

Mapa

En este caso, la visualización del mapa será a pantalla completa, para facilitar su navegación en pantallas pequeñas. El mapa incluirá, al igual que el mapa de la versión de escritorio, marcadores de los 20 eventos más próximos, y podrá aparecer centrado en la localización del usuario, si así lo requiere.

Evento

En la visualización móvil, la interfaz de eventos contendrá todos los detalles de la versión completa, a excepción del mapa de su localización y de la zona de exportación a contenido semántico, ya que en esta versión los eventos no contendrán marcado semántico.

Todas las vistas de la versión móvil deberán tener una zona de navegación para su retorno a la pantalla de bienvenida.

Así pues, el diseño conceptual de cada una de las vistas de la versión móvil de este módulo será el siguiente:

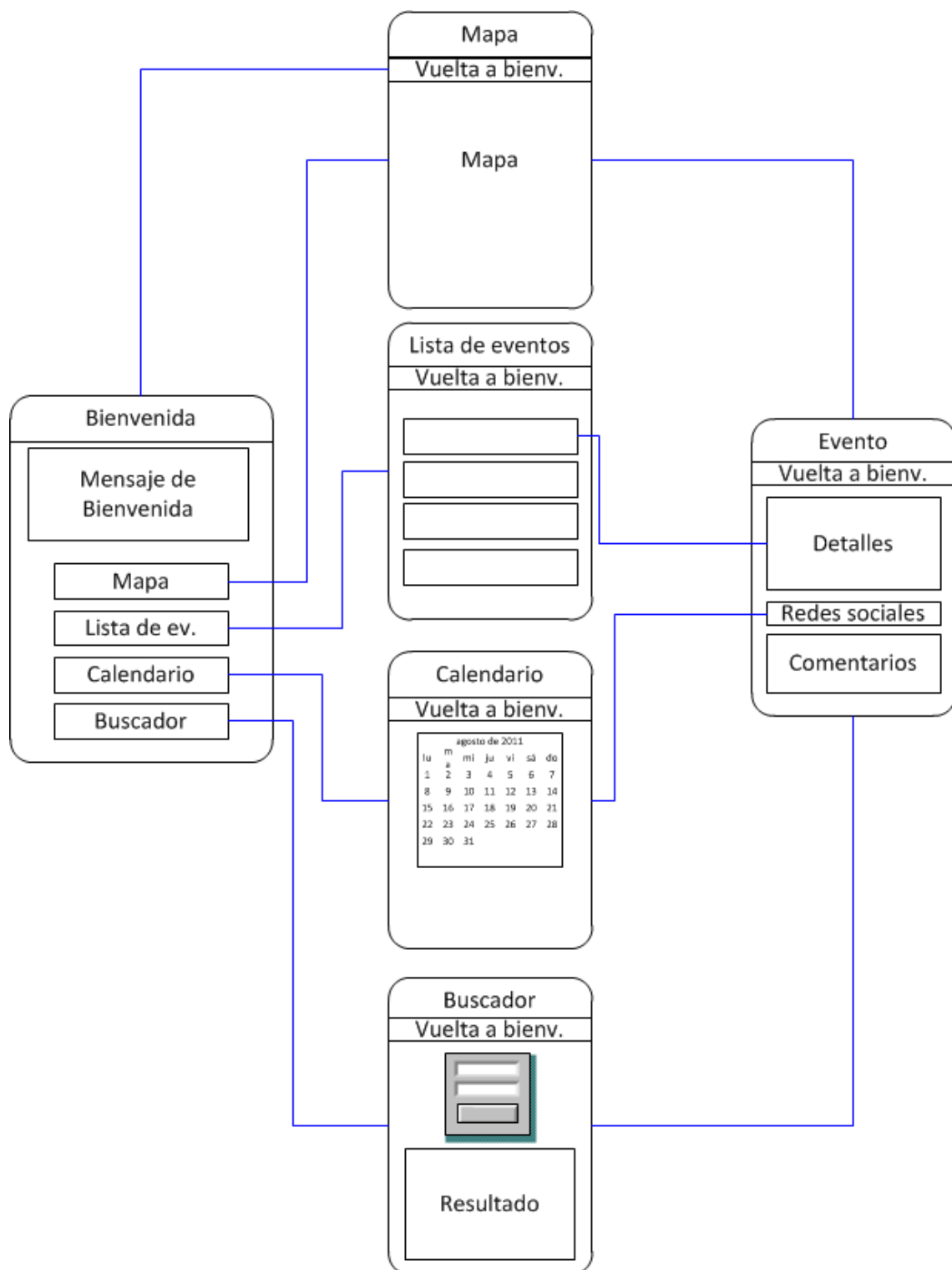


Figura 6.9: Mapa conceptual de la navegación de diálogos de la versión móvil del módulo de publicación

6.4.3 Interacción con módulo de administración

El módulo de administración constará de una vista que requerirá la validación de usuario y contraseña y que da acceso a la parte de administración de la aplicación. En caso de no validar la autenticación de un usuario, se mantendrá esta vista incluyendo un mensaje de error.

En caso de conseguir el acceso, se accederá a la vista principal de la administración del sitio, el cual mostrará todos los elementos modificables. Los elementos administrables serán de dos tipos:

- Gestión de usuarios y permisos: Para el alta, baja o modificación de cuentas que tienen acceso a la zona de administración, así como para la configuración de sus permisos.
- Gestión del contenido de la base de datos: Para el alta, baja o modificación de los contenidos de las diferentes tablas de la base de datos (eventos, lugares, etc.).

Cada uno de los contenidos visibles en esta vista llevará a una nueva plantilla, en la cual se podrán consultar y modificar los contenidos de la tabla seleccionada.

Además, desde la ventana principal de administración del sitio, se podrá abrir el sitio de traducción. Este sitio deberá abrirse en una nueva pestaña del navegador, y desde ella, se podrá seleccionar el idioma al que se quiere traducir. Una vez seleccionado el idioma, se accederá a una vista basada en formularios en la que se verá el valor a traducir y su traducción actual.

El mapa del sitio de administración, mostrando el diseño conceptual de cada una de las vistas, sería el siguiente:

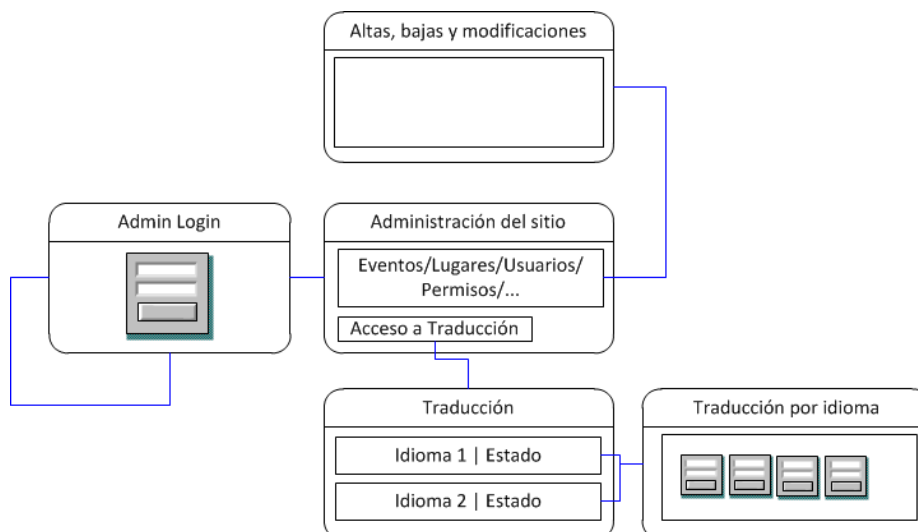


Figura 6.10: Mapa conceptual de la navegación de diálogos del módulo de administración

6.4.4 Interacción con el módulo de API

La interacción con el módulo de API está basada en la introducción de mensajes HTTP que el módulo interpreta para dar la salida requerida. Las URLs de acceso a la API deben estar estructuradas de la siguiente manera:

- URI para cada evento:

<code>http://www.innovagenda.com/api/v1/(ID_EVENTO)/(PARÁMETROS GET SALIDA)</code>
--

Se trata de URLs que identifican unívocamente los eventos y permite acceder a ellos con solo introducir el identificativo del evento, que es el mismo identificativo utilizado en la web de Euskadi+Innova. Además, debe permitir la inclusión de dos parámetros opcionales HTTP GET para personalizar la forma en que se publicarán los resultados. Estos parámetros son:

- Idioma: Para especificar si se desea la información en euskera o en castellano. Si no se indica, se deberá dar la información en ambos idiomas.
- Formato de salida: Para especificar el formato de salida que se desea: XML, JSON, YAML o RDF+XML. El formato por defecto será JSON.

- URL de búsqueda:

<code>http://www.innovagenda.com/api/v1/(PARÁMETROS GET)</code>

Se trata de URLs que recogen información pasada en los parámetros HTTP GET y que devuelven los resultados de la búsqueda de eventos basada en dichos parámetros. Los parámetros sobre los que se permitirá la búsqueda son los siguientes:

- Nombre del evento, tanto en euskera como en castellano.
- Fecha de inicio del evento.
- Fecha fin del evento.

Además, la API deber permitir buscar sobre estos campos mediante diferentes operadores de búsqueda: valor exacto, valor mayor, valor menor, valor mayor o igual, valor menor o igual, valor que empieza con, valor que contiene.

En la URL de búsqueda también pueden incluirse los parámetros HTTP GET opcionales de idioma y formato, con las mismas condiciones especificadas anteriormente.

6.5 DISEÑO DE LA BASE DE DATOS

La base de datos a incluir en el proyecto será una base de datos relacional. Las bases de datos relacionales son las que se basan en el modelo relacional, el modelo más utilizado para el diseño y administración de bases de datos. Este modelo se basa en un conjunto de entidades, que representan objetos del mundo real, que contienen diferentes atributos, que representan las características de dichas entidades, y de las relaciones entre dichas entidades. En la estructura interna de la base de datos las entidades se almacenan como tablas, que se componen de registros (filas) y columnas, que representan los atributos.

El diseño de la base de datos principal para este proyecto se basa en el almacenaje de eventos. Para representar este **modelo** de datos, se utiliza el siguiente diagrama del modelo relacional:

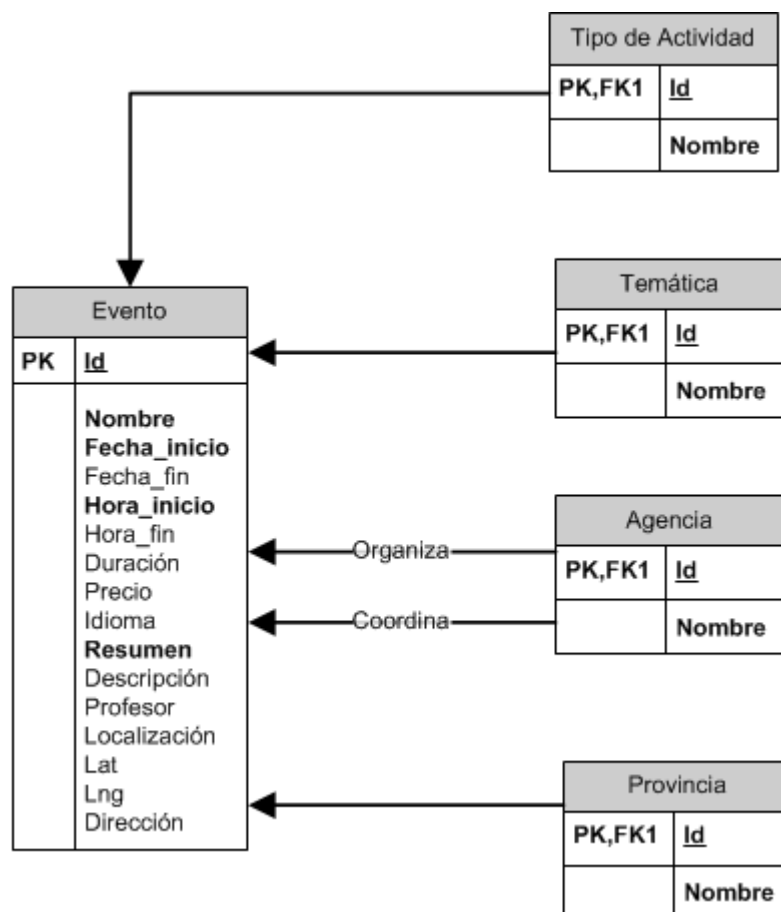


Figura 6.11: Diagrama del modelo relacional de la base de datos

Teniendo en cuenta el diagrama del modelo relacional expuesto, el esquema del modelo relacional será el siguiente:

TIPO_DE_ACTIVIDAD (<u>Id</u> , Nombre)
TEMÁTICA (<u>Id</u> , Nombre)
AGENCIA (<u>Id</u> , Nombre)
PROVINCIA (<u>Id</u> , Nombre)
EVENTO (<u>Id</u> , Nombre, Fecha_Inicio, Fecha_Fin, Hora_Inicio, Hora_Fin, Duración, Precio, Idioma, Resumen, Descripción, Profesor, Localización, Lat, Lng, Finished, Id_Prov, Id_Provincia, Id_TipodeActividad, Id_Tematica, Id_Agencia_Org, Id_Agencia_Coord)

Figura 6.12: Esquema del modelo relacional de la base de datos

Por lo tanto, en la base de datos se guardarán las siguientes entidades:

- **Tipo de Actividad:** Esta entidad guardará los tipos que Euskadi+Innova establece para sus eventos (Reunión, Curso, Jornada, Barnetegi, Taller, Conferencia, Congreso y Feria).
- **Temática:** Esta entidad contendrá las diferentes temáticas que cubren los eventos de Euskadi+Innova (Innovación tecnológica, Innovación social, Transformación empresarial y Empresa Digitala).
- **Agencia:** Esta entidad guardará las diferentes agencias que organizan y coordinan los eventos. Estas agencias no están predefinidas por Euskadi+Innova, por lo que se deberán descubrir y guardar según los datos de los eventos que obtenidos en el parseo.
- **Provincia:** Esta entidad guardará las provincias en las que se realizan los eventos (Bizkaia, Gipuzkoa, Araba y Otras).
- **Evento:** Esta entidad es la entidad principal de la base de datos, ya que contiene todos los datos que se desean guardar acerca de los eventos. Se relaciona con todas las demás entidades con relaciones de 1 a N, y guarda dos relaciones de este tipo con la entidad Agencia, debido a que ésta es la encargada de guardar los datos de las agencias tanto organizadoras como coordinadoras, las cuales no tiene porqué ser las mismas para un evento. El identificativo –y clave primaria- de esta entidad será el identificativo de evento, el cual coincidirá con el identificativo de eventos que utiliza Euskadi+Innova internamente, para que exista mayor compatibilidad entre las dos webs.

Además de estas entidades, la base de datos podrá contener más objetos que se deban crear por las necesidades de los distintos módulos de la aplicación.

7. DESARROLLO

En este capítulo se mostrará la parte más técnica del desarrollo del proyecto. En él se detallarán las herramientas elegidas para el desarrollo del proyecto, y se explicarán tanto la estructura como los detalles de la implementación de los diferentes módulos.

7.1 HERRAMIENTAS UTILIZADAS

7.1.1 Framework de desarrollo web: Django y Python

Para el desarrollo de la web se analizaron diversas opciones, para las cuales se estimó que se debían cumplir las siguientes condiciones:

- Que cumpla el paradigma del Modelo-Vista-Controlador o derivados.
- Que cumpla con una metodología ágil de desarrollo.
- Que sea extensible.
- Que permita diferentes Sistemas de Gestión de Bases de Datos (SGBD) y haga transparente su uso.
- Que permita la creación y uso de contenidos semánticos.

Teniendo en cuenta esas características, se compararon varios frameworks de desarrollo web: Jena, Django y Ruby-on-Rails (RoR). La comparación tuvo como resultado la siguiente tabla comparativa:

	JENA	DJANGO	RoR
Descripción	Framework para el desarrollo de Web Semántica basado en Java Server Pages	Framework para el desarrollo de webs complejas de forma sencilla basado en Python	Framework para el desarrollo de webs complejas de forma sencilla basado en Ruby
Lenguaje de Programación	Java	Python	Ruby
MVC	Adaptando Java Server Pages	Derivación del MVC: MTV (Model, Template, View)	Sí
SGBD	TDB (No-relacional y específica para RDF), SDB (Relacional, soporta MySQL, PostgreSQL, Oracle...)	Bases de datos relacionales (MySQL, PostgreSQL, Oracle...). Transparencia de uso.	Bases de datos relacionales (MySQL, PostgreSQL, Oracle...). Transparencia de uso.
Web Semántica	Framework centrado en web semántica con diversas herramientas integradas	Librerías de Python y módulos para Django para adaptar el framework a la web semántica	Librerías de Ruby y módulos para RoR para adaptar el framework a la web semántica.
Base de Datos RDF	TDB y SDB	Es necesaria la adaptación de los RDF al modelo relacional mediante librerías	Es necesaria la adaptación de los RDF al modelo relacional mediante librerías
Extensibilidad	Librerías para Java	Multitud de módulos reutilizables para Django, y librerías para Python	Multitud de módulos reutilizables para RoR, y librerías para Ruby

Figura 7.1: Tabla de comparación de frameworks web

Teniendo en cuenta esta comparación, y viendo las posibilidades que ofrecen Django y RoR para adaptarse a la web semántica, se decidió desechar la opción de Jena, por ser un sistema más pesado y estricto en cuanto a su utilización. Otra de las razones para desechar esta opción fue la extensibilidad y flexibilidad de los otros frameworks, así como las posibilidades que ofrecen los lenguajes de programación en los que están basados.

Django y Ruby-on-Rails son dos frameworks que comparten muchas similitudes, pero el framework de desarrollo que se terminó utilizando fue Django. Fue decisiva para esta decisión la gran base de documentación y soporte acerca de Django y sus módulos, así como la potencia de Python y el gran número de librerías disponibles para este lenguaje de programación.

Django

Django es un framework de desarrollo web open source liberado bajo licencia BSD. Sus características principales son:

- Basado en Python.
- Se adhiere a la metodología DRY (Don't Repeat Yourself), es decir, se basa en módulos reutilizables con el objetivo de que no se repitan funcionalidades.
- Mapeador objeto-relacional: Permite la definición del modelo de base de datos completo en Python, y ofrece una API sencilla para el acceso a la base de datos de forma sencilla y transparente del SGBD utilizado.
- Paradigma MTV (Model Template View), que es una adaptación del patrón MVC a la filosofía de los creadores de Django. "Model" representa lo mismo que los modelos del MVC, es decir, la representación de la base de datos. "Template" es la representación visual de la web, es decir, lo equivalente a las vistas de MVC. Y "View" representan la lógica previa a la representación de las "Templates", es decir, lo equivalente a los controladores de MVC.
- Un sistema de plantillas con herencia y basado en etiquetas personalizadas.
- Diseño de URLs flexible y elegante.
- Multitud de módulos integrados en el propio framework, como los de administración, autenticación o internacionalización, entre otros.
- Sistema de middleware configurable y que proporciona por defecto opciones para cacheo, compresión de salida, normalización de URLs, protección CSRF y soporte de sesiones.
- Gran documentación y soporte.

Python

Python es un lenguaje de programación de alto nivel multiplataforma y open source con las siguientes características

- Sintaxis limpia y sencilla de leer.
- Lenguaje de programación multi-paradigma, es decir, se adapta a la orientación a objetos, a la programación funcional y programación imperativa.
- Lenguaje interpretado.
- Lenguaje de tipado dinámico y fuertemente tipado.
- Ampliable mediante extensiones.
- Consola de programación dinámica multiplataforma.

7.1.2 Librerías Python utilizadas

Se han utilizado múltiples librerías para Python para el desarrollo de esta aplicación web, además de las que Python trae por defecto. Son las siguientes:

BeautifulSoup

Librería para Python para el parseo de documentos HTML/XML. Ofrece una serie de métodos para el parseo de los documentos, y para su navegación, búsqueda y modificación. Es capaz de funcionar con HTMLs y XMLs con formatos incorrectos.

Ha sido utilizado en el módulo de obtención de datos de Euskadi+Innova para el parseo de los HTML de eventos, y en el módulo de geolocalización de eventos para los XML de respuesta de la API de Geolocalización de Google Maps.

- Versión utilizada: BeautifulSoup 3.2.0
- Licencia: BSD
- URL: <http://www.crummy.com/software/BeautifulSoup/>
- Autor: Leonard Richardson

Universal Feed Parser

Librería para Python para el parseo y manejo de fuentes RSS y Atom. Llamando al método de parseo, FeedParser se conecta al feed y devuelve un diccionario de Python con la información.

Ha sido utilizado en el módulo de obtención de datos de Euskadi+Innova para la conexión al RSS y el parseo de su contenido.

- Versión utilizada: FeedParser 5.0.1
- Licencia: MIT
- URL: <http://www.feedparser.org/>
- Autor: Mark Pilgrim

Google Data Python Client

Librería para Python que ofrece un protocolo simple de acceso a los datos de una cuenta de Google. Entre las APIs que contiene, se encuentra la de Google Calendar; esta API provee de acceso a los calendarios de una cuenta, y permite la lectura y modificación de todos los aspectos de un calendario, desde configuraciones a eventos individuales.

Ha sido utilizada en el módulo de actualización de calendarios de Google Calendar, para la conexión a la cuenta y la creación de nuevos eventos en los dos calendarios que la aplicación maneja.

- Versión utilizada: GData 2.0.14
- Licencia: Apache License 2.0
- URL: <http://code.google.com/p/gdata-python-client/>
- Autor: Google Inc.

RDFLib + RDFExtras

Librería para Python para trabajar con el estándar RDF. Proporciona parseadores y serializadores para diferentes formatos: RDF/XML, N3, N-Triples, Turtle, Trix y RDFa. La librería se usa mediante objetos en forma de grafo, que representan las tripletas RDF. Esta librería incluye también la opción de guardar tripletas RDF en bases de datos convencionales como MySQL, SQLite o SQLAlchemy, entre otras.

La versión 3.1.0 de RDFLib separaba algunas de estas funciones en otro paquete llamado RDFExtras.

Estas librerías han sido utilizadas en el módulo de publicación para la exportación de contenido semántico a formatos RDF/XML, N3, N-Triples y Turtle, y para el parseo del RDFa publicado en la sección de eventos. También han sido utilizadas para crear el serializador RDF de la API de InnovAgenda.

- Versión utilizada: RDFLib 3.1.0 + RDFExtras 0.1

- Licencia: New BSD License
- URL: <http://www.rdfliib.net/>
- Autor: Varios autores (<http://code.google.com/p/rdfliib/>)

RDFLib-MicroData

Pequeño plugin para RDFLib para habilitar el parseo de Microdatos HTML5 con RDFLib.

Este plugin ha sido utilizado en el módulo de publicación para el parseo de los Microdatos HTML5 de la sección de eventos, para su posterior exportación usando RDFLib.

- Versión utilizada: Versión de desarrollo publicada en GitHub
- Licencia: Dominio público
- URL: <https://github.com/edsu/rdfliib-microdata>
- Autor: Ed Summers

7.1.3 Módulos Django utilizadas

Se han utilizado varios módulos reutilizables de Django para ampliar la funcionalidad del framework. Son los siguientes:

Django-Localeurl

Django-localeurl es un pequeño modulo reutilizable para Django para introducir de forma automática códigos de idiomas en las URLs del proyecto, de manera que cuando se acceda en castellano el formato de las URLs sea <http://www.innovagenda.com/es/> y en euskera <http://www.innovagenda.com/eu/>. Siguiendo esta nomenclatura, se ayuda al posicionamiento de la web en buscadores, ya que éstos indexan las webs por idioma siguiendo este formato.

Además lo hace sin que esto conlleve ningún prejuicio sobre los usuarios, ya que incluye métodos de redirección automática al idioma del navegador desde el que se visita la web (en caso de que exista versión para dicho idioma, en caso contrario redirecciona a la versión con el idioma por defecto de la aplicación web).

Se adapta perfectamente a Django ya que lo que hace es instalar un middleware para el framework que no entra en conflicto con ninguno de los demás middlewares. Además provee de etiquetas para los templates de Django para ayudar a construir selectores de idiomas.

Este módulo ha sido utilizado en el proyecto para la adaptación de las URLs a este tipo de formato, y para la adición del selector de lenguajes del módulo de publicación.

- Versión utilizada: django-localesurl 1.4
- Licencia: MIT
- URL: <http://pypi.python.org/pypi/django-localesurl>
- Autor: Joost Cassee, Artiom Diomin y Carl Meyer

Django-Rosetta

Django-rosetta es una aplicación que ofrece una interfaz para la traducción de proyectos basada en la interfaz de usuario del sitio de administración por defecto que Django provee. Además de la interfaz, Rosetta provee de un servicio de sugerencias de traducción basada en Google Translator.

Este módulo ha sido utilizado para proporcionar el sitio de traducción para la administración del sitio, así como para la traducción de las vistas de toda la aplicación web.

- Versión utilizada: django-rosetta 0.6.2
- Licencia: MIT
- URL: <https://github.com/mbi/django-rosetta>
- Autor: divio.shared@gmail.com

Django-Tastypie

Django-Tastypie es un framework para la creación de APIs REST para Django. Se trata de un módulo altamente configurable y potente.

Es capaz de proporcionar una serie de URLs de acceso a los modelos de tu proyecto, seleccionando los atributos a partir de los cuales permitir la búsqueda y las operaciones posibles sobre la búsqueda. Además, no solo provee una API de lectura de datos, sino que también es posible la habilitación de una API de escritura de datos. Para ambos tipos, Django-Tastypie ofrece la posibilidad de gestionar un sistema de autenticación opcional para manejar los permisos en el uso de la API que implementes.

Ofrece múltiples opciones de serialización de los resultados, es decir, es capaz de publicar la información de tus modelos en diferentes formatos: JSON, XML, YAML... Además, ofrece la posibilidad de añadir tus propios serializadores a los formatos que tu aplicación requiera.

Django Tastypie ha sido utilizada en este proyecto para implementar el módulo de la API de InnovAgenda, del cual se han aprovechado sus capacidades de configuración y personalización, y gracias a las cuales se ha habilitado el serializador RDF de esta API.

- Versión utilizada: django-tastypie 1.0.0-beta
- Licencia: BSD
- URL: <https://github.com/toastdriven/django-tastypie>
- Autor: Daniel Lindsley

Django-Transmeta

Se trata de una aplicación reusable que permite traducir los modelos de las aplicaciones Django de forma sencilla. Django-Transmeta añade automáticamente campos extra en los modelos para los atributos que quieren ser traducidos. Además, hace transparente el acceso a los valores ya que funciona de tal manera que devuelve el valor según el idioma actual de la aplicación (el idioma que el usuario ha seleccionado, o bien el idioma por defecto). Cabe comentar también que es totalmente compatible con el módulo de administración de Django.

Se ha utilizado esta aplicación para la habilitación de modelos multilenguaje, que era una de las necesidades de la aplicación al obtener datos de los mismos eventos tanto en euskera como en castellano.

- Versión utilizada: django-transmeta 0.6.2
- Licencia: GNU Lesser GPL
- URL: <http://code.google.com/p/django-transmeta/>
- Autor: Marc García y Manuel Saelices

Django-Mobile

Se trata de una aplicación reusable que añade un middleware a Django para la detección del tipo de dispositivo desde el que se visita la web, y que se compone de multitud de opciones de configuración.

Además, este módulo provee al proyecto de un mecanismo de intercambio de plantillas HTML transparente, es decir, que el propio Django-Mobile se encarga de renderizar la plantilla acorde con el tipo de visitante.

Por último, añade a plantillas y vistas nuevas etiquetas y funciones para el intercambio dinámico de las vistas.

- Versión utilizada: django-mobile 0.2.1
- Licencia: BSD
- URL: <https://github.com/gregmuellegger/django-mobile>

- Autor: Greg Muellegger

Django-Haystack

Se trata de una aplicación reutilizable que añade un módulo de búsqueda en aplicaciones Django. Soporta varios motores de búsqueda: Solr, Whoosh y Xapian. Basándose en el motor elegido, desarrolla formularios de búsqueda personalizables para los modelos Django que se elijan.

Para el proyecto, se ha hecho uso de Haystack en conjunción con **Whoosh 2.2.2** –una librería rápida y potente para indexación y búsqueda en textos, escrita nativamente en Python-, para el desarrollo del sub-módulo de búsqueda de eventos.

- Versión utilizada: django-haystack 1.2.4
- Licencia: BSD
- URL: <http://haystacksearch.org/>
- Autor: Daniel Lindsley

7.1.4 Base de datos: PostgreSQL

Para la base de datos relacional, se decidió utilizar un Sistema de Gestión de Bases de Datos libre. Por su estabilidad, madurez, capacidad de transacción y flexibilidad, se eligió PostgreSQL como SGBD.

Sus características son las siguientes:

- SGBD libre, publicado bajo licencia BSD.
- Sistema relacional y orientado a objetos.
- Amplia variedad de tipos nativos.
- Soporte para alta concurrencia.
- Posibilidad de escribir bloques de código programático que se ejecuten en el servidor de base de datos.
- Sistema ampliable mediante extensiones.
- Multitud de herramientas alrededor del sistema: herramientas de administración, de búsqueda, etc.

7.1.5 Otras herramientas: JQueryMobile

JQuery Mobile es un framework de desarrollo web para móviles. Se trata de una librería JavaScript que provee de una serie de contenidos web adaptados a la navegación desde terminales móviles. Es compatible con la mayoría de los navegadores de terminales móviles y tablets.

Ofrece todo tipo de componentes para la estructuración de componentes dentro de una página web adaptada a móviles, así como barras de navegación, botones, formularios, listas, tablas y demás elementos de páginas compatibles con navegación táctil.

Sus características más importantes son las siguientes:

- Escrito sobre el núcleo de JQuery.
- Compatible con la mayoría de plataformas móviles y de escritorio.
- Ligera y con pocas dependencias.
- Basado en marcado HTML5.
- Accesibilidad al contenido siguiendo el estándar WAI-ARIA.
- Multitud de widgets de interfaz de usuario.
- Customizable con diferentes temas.

7.2 FUNCIONAMIENTO DE DJANGO

Antes de entrar en detalles técnicos acerca de la estructura e implementación del proyecto, es conveniente explicar cómo funciona Django, y cómo se implementa el patrón MTV (la variación de Django del patrón Modelo-Vista-Controlador), para comprender mejor las decisiones tomadas.

El esquema de funcionamiento de Django es el siguiente:

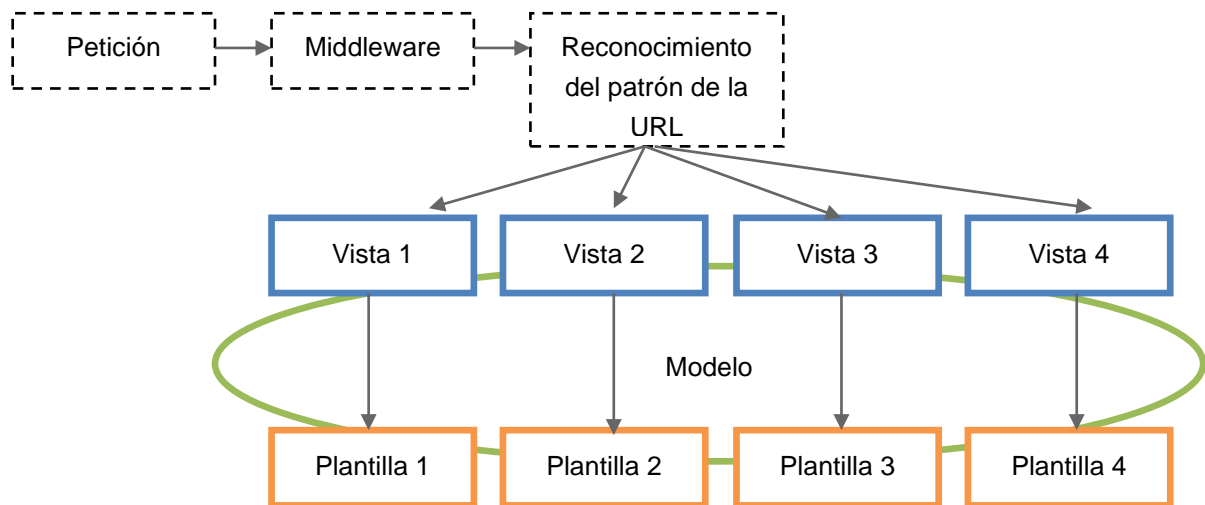


Figura 7.2: Esquema de funcionamiento de Django

Cómo se puede observar, una aplicación Django lo primero que hace tras recibir una petición, es activar los procesos de middleware para el procesamiento de la petición. Una vez ejecutados los métodos middleware, se intenta reconocer la URL de la petición, con una serie de patrones de URLs que especificadas al crear la aplicación. Si el formato de la URL se corresponde con alguno de los patrones, se llama a la vista asociada a dicho patrón. La vista es el método que lleva la lógica de la aplicación, es decir, lo equivalente al “controlador” de MVC.

Desde las vistas se puede acceder al modelo. El modelo es la representación de la base de datos en el proyecto, y es independiente de vistas y plantillas. Una vez ejecutada la lógica de las vistas, se renderizan las plantillas. Las plantillas son las visualizaciones de la web, es decir, lo equivalente a las “vistas” de MVC. Se trata de plantillas HTML a las que Django les añade etiquetas personalizadas para crear contenido dinámico dependiente de las salidas de las vistas.

7.3 ESTRUCTURA DEL PROYECTO

La estructura de carpetas del proyecto es la que se puede apreciar en la Figura 7.3. En ella se pueden apreciar las diferentes carpetas y ficheros que forman parte de la estructura del proyecto.

Antes de entrar en detalle acerca de los diferentes archivos, debe comentarse que los archivos “__init__.py” normalmente son archivos vacíos que sirven para especificar que la carpeta que lo contiene es un paquete Python.

Los archivos “dotcloud.yml” y “wsgi.py” son archivos de configuración que se han requerido para el paso a producción del proyecto (ver Anexo II para más detalles).

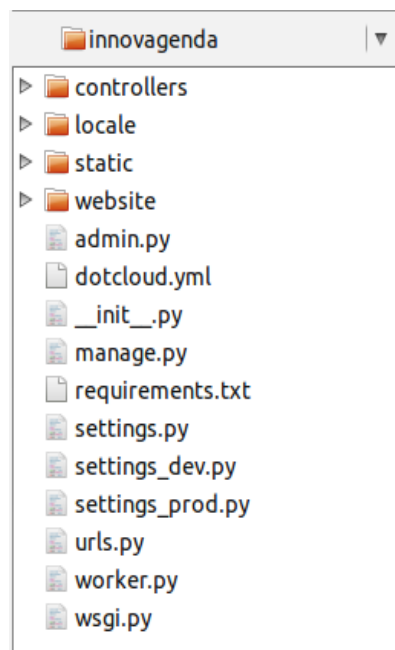


Figura 7.3: Estructura de carpetas del proyecto

El archivo “requirements.txt” es un archivo de requerimientos. Éste explica las dependencias del proyecto; estas se especifican en el formato que PIP indica. PIP es un instalador y administrador de paquetes Python que es capaz de instalar dependencias automáticamente a partir de archivos “requirements.txt”.

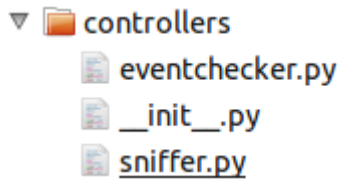
El archivo “worker.py” es un script para la ejecución automática de los módulos del servidor.

El resto de archivos y carpetas de que se pueden observar en la estructura, son las propias de un proyecto Django:

- “manage.py” es el script de administración que se crea al crear un proyecto de Django. Sirve para lanzar tareas administrativas del proyecto como la sincronización de la base de datos o el inicio del servidor de desarrollo.

- Los archivos “settings_X.py” son archivos de configuración del proyecto Django. En ellos se especifican las opciones de lenguaje, las conexiones de bases de datos, los módulos instalados, el lugar de los archivos estáticos y todas las demás opciones de cada proyecto. En este caso “settings.py” es la base de los otros dos archivos de configuración, es decir, el que contiene las opciones en común de los otros dos. “settings_dev.py” es el archivo de configuración para el entorno de desarrollo, que contiene la configuración específica de la conexión de bases de datos y otras opciones de desarrollo y debugueo activadas; mientras que “settings_prod.py” es el archivo de configuración del entorno de producción, en el que se especifica principalmente la conexión a la base de datos de producción.
- “urls.py” es el archivo de configuración de las URLs del proyecto. En él se especifican mediante expresiones regulares los patrones de URLs a los que se deben responder, y qué métodos del “View” del paradigma MTV se deben ejecutar antes de la publicación del “Template”.
- “admin.py” es el archivo de configuración del módulo de administración por defecto de Django.
- La carpeta “locale” incluye los archivos con extensión “.po” para la traducción de la web a otros idiomas a parte del por defecto. Los archivos con extensión “.po” son los que Django utiliza para devolver valores traducidos.
- La carpeta “static” incluye los archivos estáticos del proyecto. Estos archivos son los documentos CSS de estilo de la web, las imágenes y los archivos JavaScript.

Las carpetas “controllers” y “website” contienen contenido más complejo:



La carpeta “controllers” contiene las implementaciones en Python de los módulos de obtención de datos, de geolocalización de eventos, de publicación de eventos y de revisión de eventos finalizados.

Figura 7.4: Estructura del paquete de controladores El archivo “eventchecker.py” es el que contiene el método de detección de eventos finalizados.

El archivo “sniffer.py” contiene los métodos de implementación del resto de módulos. Se han implementado en un mismo archivo porque son métodos dependientes entre sí.

Se trata de métodos cuya ejecución es independiente del resto de la web, por lo que no dependen de ella. En cambio, sí que utilizan los archivos de configuración de la aplicación para acceder a los modelos de la base de datos. Por este motivo es por el que se creó el script “worker.py” antes mencionado, ya que éste fuerza al intérprete de Python a utilizar el archivo de configuración necesario.

La carpeta “website” contiene la aplicación Django llamada “website”, que es el grueso de los módulos que tienen interacción con el usuario (publicación, administración y API). En esta aplicación se configura el patrón MTV de Django:

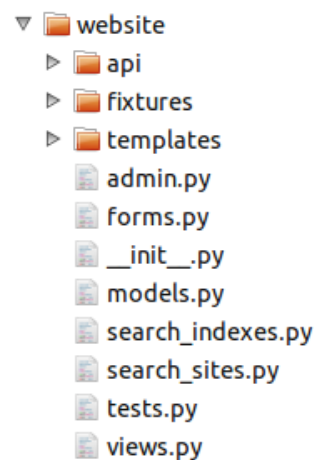


Figura 7.5: Estructura de la aplicación Django

El archivo “views.py” es en el que se encuentra la lógica de la aplicación, es decir, los diferentes métodos que se ejecutan antes o tras la interacción del usuario con el módulo de publicación. Éstos son los métodos que se llaman desde el archivo “urls.py”.

El archivo “models.py” es en el que se especifican los modelos de la aplicación, es decir, la representación de la base de datos.

El archivo “tests.py” es un archivo que Django crea automáticamente para realizar pruebas de la aplicación, pero que en este caso no ha sido utilizado.

Los archivos “search_indexes.py” y “search_sites.py” son los archivos de configuración de Django-Haystack, el módulo de búsqueda de la aplicación.

El archivo “forms.py” es el archivo en el que se especifica el contenido de los diferentes formularios que pueden aparecer en las plantillas HTML. Se trata de una forma de trabajo que Django provee para realizar de forma sencilla la gestión de datos de los formularios.

La carpeta “templates” es en la que se encuentran todas las plantillas HTML del proyecto. Hay 3 tipos de plantillas dentro de esta carpeta:

- Las que están en la raíz: Correspondientes a la versión de escritorio de los diferentes submódulos del módulo de visualización.
- Las de la carpeta “mobile”: Correspondientes a la versión móvil de los diferentes submódulos de la versión móvil del módulo de visualización.
- Las de “admin”: Correspondientes a las modificaciones realizadas en las plantillas del módulo por defecto de administración de Django.

La carpeta “fixtures” incluye un solo fichero JSON que contiene los datos iniciales para los modelos de la aplicación.

La carpeta “API” contiene dos archivos:

- “api.py”: Es el archivo de configuración de Django-Tastypie, en el que se especifican las opciones del módulo de la API.
- “RDFSerializer.py”: Es el archivo en el que se incluye la implementación del serializador RDF desarrollado para Tastypie.

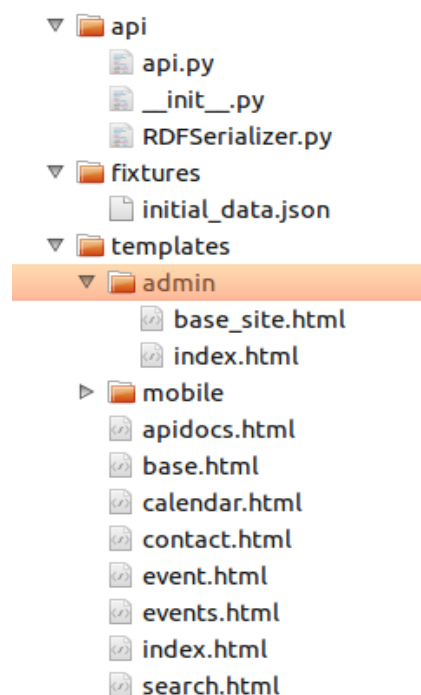


Figura 7.6: Contenido de las carpetas de la aplicación

7.3.1 Resumen de paquetes Python del proyecto

Resumiendo, y teniendo en cuenta esta estructura y los diferentes paquetes Python creados en ella, el proyecto se estructura de la siguiente manera:

- Paquete **innovagenda**: Paquete base que incluye la configuración de los diferentes aspectos de la aplicación web.
- Paquete **innovagenda.controllers**: Paquete que contiene los scripts que se ejecutan en el servidor.
- Paquete **innovagenda.website**: Paquete que contiene la aplicación Django “website”, que es en la que se implementan las plantillas HTML, las vistas con la lógica previa y posterior a la interacción del usuario con las plantillas, y los modelos de la base de datos.
- Paquete **innovagenda.website.api**: Paquete para la configuración de la API y en el que se incluye la implementación de un nuevo serializador para Django-Tastypie.

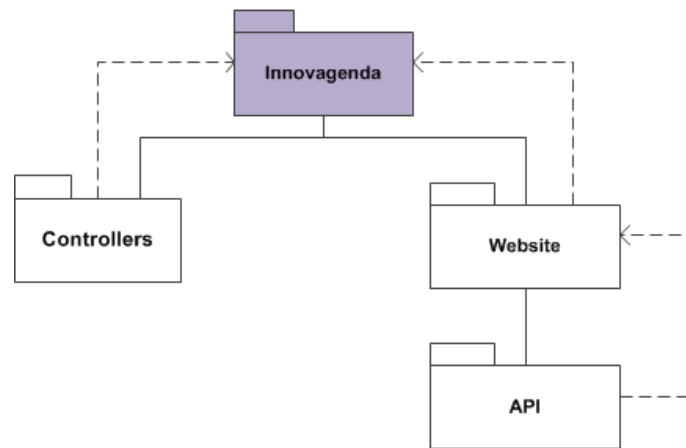


Figura 7.7: Diagrama de estructura de paquetes de la aplicación

7.4 DIAGRAMA DE CLASES

Antes de comenzar con los detalles de implementación de cada módulo del proyecto, es recomendable explicar la lista de clases que se han creado para el manejo de datos de la aplicación, así como las relaciones entre ellas.

Se debe explicar que el modelo de desarrollo web MTV planteado por Django es un modelo que basa sus partes de “Template” (Plantillas) y “Views” (Vistas) en **programación funcional**. La parte del “Model” (Modelo) sí que conlleva una estructura de clases que representan los objetos que dan acceso a la base de datos. La estructura de Templates y Views del proyecto se explicará al explicar la implementación del módulo de publicación, que es el que realmente hace uso de este modelo.

Además de las clases del modelo de datos, el módulo de administración de Django (usado para proporcionar el backend de administración del sitio) y la aplicación Django-Tastypie (usada para crear la API), crean sus propias clases a las que también se accede durante la implementación.

Teniendo en cuenta lo explicado, el diagrama de clases resultante es el siguiente:



7.5 IMPLEMENTACIÓN

7.5.1 Implementación del modelo de datos

Antes de comenzar la explicación de la implementación de cada uno de los módulos, es conveniente explicar la implementación del modelo de datos que utilizan todos ellos. Como se ha explicado anteriormente, los modelos de Django son representaciones de las entidades de la base de datos.

Para su implementación se debe modificar el archivo `models.py` que se crea al crear una aplicación Django. En este caso el `models.py` se corresponde con la aplicación “website” y las clases que se crean en él son las especificadas en el diagrama de clases. La implementación correspondiente al modelo de eventos, es la siguiente:

```
class Event(models.Model):
    __metaclass__ = TransMeta

    event_id = models.IntegerField()
    name = models.CharField(max_length=150)
    inserted_date = models.DateTimeField(auto_now_add=True)
    start_date = models.DateField()
    finish_date = models.DateField(blank=True, null=True)
    start_hour = models.TimeField(blank=True, null=True)
    finish_hour = models.TimeField(blank=True, null=True)
    duration = models.IntegerField(blank=True, null=True)
    price = models.FloatField(blank=True, null=True)
    description = models.CharField(max_length=10000)
    summary = models.CharField(max_length=3000)
    teacher = models.CharField(max_length=50, blank=True, null=True)

    url = models.CharField(max_length=300)
    gcalendar_url = models.CharField(max_length=300)
    finished = models.BooleanField(default=False)

    localization = models.CharField(max_length=300)
    address = models.CharField(max_length=300)
    lat = models.FloatField()
    lng = models.FloatField()

    prov = models.ForeignKey(Province)
    course_lang = models.ForeignKey(Language)
    activity_type = models.ForeignKey(ActivityType)
    topic = models.ForeignKey(Topic)
    coordinator = models.ForeignKey(Agency,
    related_name="event_coordinator")
    organizer = models.ForeignKey(Agency,
    related_name="event_organizer")
```

Ilustración 7.9: Implementación del modelo Event

Cómo se puede observar, la implementación de cada clase del modelo se basa en la declaración de cada uno de sus atributos, y de los tipos de dichos atributos. La implementación del resto de modelos es similar a la de Event.

7.5.2 Implementación del módulo de obtención de datos de Euskadi+Innova

Tras analizar la estructura tanto del RSS como de los HTML de las diferentes páginas de eventos que Euskadi+Innova publica, se determinó de dónde de cuál de las dos fuentes se obtendría cada uno de los campos a llenar en la base de datos de eventos. Así pues se consideró que, debido a la estructura confusa del RSS, se tomaría como fuente principal de datos el HTML de cada evento.

Los RSS a los que conectarse son 4, uno por temática. Se decidió acceder a los 4 por separado y desechar la opción de utilizar únicamente el general porque de esta manera se obtenía la opción ya dividida por temáticas y porque los 4 por separado ofrecían más contenido que el general. Para la conexión y parseo de los RSS se hizo uso de la librería Feedparser, para la conexión a los HTML de eventos se usó la librería nativa de Python urllib2, y BeautifulSoup para su parseo..

A continuación se explicará detalladamente el código programático de este módulo:

1. En la primera parte de código se establece la inicialización del módulo:
 - Se crea un “opener” (un plugin) para la habilitación de Cookies en Urllib2. Se trata de algo necesario para conectarse a la web de Euskadi+Innova.
 - Se crea un diccionario con las URLs de los diferentes RSS especificando en él también la temática de cada uno, para obtener directamente este campo de la base de datos.
 - Por último, se especifican los diferentes patrones de expresiones regulares que se van a utilizar durante la implementación del módulo. Estas expresiones regulares se compilan y evalúan mediante la librería “re” que Python provee por defecto. Se establecen patrones para la identificación del identificador de evento (identificador numérico), fechas, números enteros, números reales, y horas.

```
def sniff():
    opener = urllib2.build_opener(urllib2.HTTPCookieProcessor) #Cookie
    enable url opener
    rss_urls = {

"http://www.euskadinnova.net/modulos/RSSGenerar.aspx?language=es&temat
ica=2&tipo=2": "Innovación Tecnológica",

"http://www.euskadinnova.net/modulos/RSSGenerar.aspx?language=es&temat
ica=3&tipo=2": "Innovación Social",
```

```

"http://www.euskadinnova.net/modulos/RSSGenerar.aspx?language=es&temat
ica=4&tipo=2": "Transformación Empresarial",

"http://www.euskadinnova.net/modulos/RSSGenerar.aspx?language=es&temat
ica=5&tipo=2": "Empresa Digitala",
}

#re pattern for getting event ID
pat_evid = re.compile('.*\/(\d+)\.aspx')
#re pattern for getting start and finish dates
pat_dates = re.compile('\d{2}\/\d{2}\/\d{4}')
#re pattern for integers
pat_digits = re.compile('\d+')
#re pattern for floats
pat_floats = re.compile('\d+,\d+')
#re pattern for getting start and finish hours
pat_hours = re.compile('\d+:\d+')

```

Figura 7.10: Código de inicialización del módulo de obtención de datos

2. En la segunda parte se establecen las conexiones y se comprueba si los datos deben ser parseados o no (es decir, si el evento en cuestión está en la base de datos o no).

- Se parsea el RSS y se recorre cada una de sus entradas, que se corresponde con cada evento que el RSS publica.
- Se obtiene el identificativo del evento evaluando la expresión regular antes especificada, y se comprueba si está en la base de datos o no.
- En caso de que el evento no está en la base de datos, se establecerá la conexión con las URL de los eventos tanto en euskera como en castellano. Debido a la estructura errónea de los RSS de Euskadi+Innova, se accede a URLs “por defecto” para ambos idiomas, en las que se especifica únicamente el identificativo del evento y se espera a la redirección al contenido requerido.

Esto se debe a que, por culpa de la forma en la que se envía el RSS (se crea dinámicamente mediante una petición ASPX), en ocasiones los RSS envían contenido en el idioma equivocado, por lo que las URL que se recogían del RSS no siempre eran correctas para ambos idiomas. Tras investigar el formato de las URLs de Euskadi+Innova, se descubrió que con ciertos patrones de URL, con solo indicar el identificativo del evento, la propia página te redirigía a la página correcta, por lo tanto, se optó por esta opción para la conexión a los HTML.

- Una vez recogidos ambos HTMLs, se parsean ambos por separado, obteniendo dos objetos navegables de BeautifulSoup.

```

for rss_url in rss_urls:
    rss = feedparser.parse(rss_url)

    for entry in rss.entries:
        #event_id
        evid = pat_evid.split(entry.link)[1]
        print "...EVENT " + str(evid) + " parsing..."
        #If the event is in the DB, no parsing
        try:
            event = Event.objects.get(event_id = evid)
            print "EVENT " + str(evid) + " already in DB" + "\n"

        except Event.DoesNotExist:
            url_es = "http://www.euskadinnova.net/es/enpresa-
digitala/agenda/x/" + str(evid) + ".aspx"
            url_eu = "http://www.euskadinnova.net/eu/enpresa-
digitala-eu/agenda/x/" + str(evid) + ".aspx"

            #Parse HTML
            response = opener.open(url_es)
            data = response.read()
            soup_es = BeautifulSoup(data)
            soup_es.prettify()

            response = opener.open(url_eu)
            data = response.read()
            soup_eu = BeautifulSoup(data)
            soup_eu.prettify()

```

Figura 7.11: Código de conexión del módulo de obtención de datos

3. La tercera parte es en la que se accede al contenido de los HTML. Para encontrar los contenidos requeridos en dicha estructura, fue necesaria la investigación minuciosa del marcado de los documentos. En esta parte se buscan los diferentes patrones que se siguen en las publicaciones de eventos de Euskadi+Innova, inspeccionando los diferentes títulos, links, tablas y demás contenidos HTML de las páginas. A continuación se detallan algunas de las obtenciones de datos mediante la búsqueda en elementos "span", "li" o "meta" de los citados documentos:

```

#name
name_es = soup_es.find('span',
id="ctl09_ltrTitulo").contents[0]
name_eu = soup_eu.find('span',
id="ctl09_ltrTitulo").contents[0]

#hours
try:
    hours = pat_hours.findall(soup_es.find('li',
id="ctl09_div_horario").next.next.next)
    hourparts = pat_digits.findall(hours[0])

```

```

        start_hour = time(int(hourparts[0]),
int(hourparts[1]))
        hourparts = pat_digits.findall(hours[1])
        finish_hour = time(int(hourparts[0]),
int(hourparts[1]))
        except AttributeError:
            start_hour = None
            finish_hour = None
        except IndexError:
            finish_hour = start_hour

        #summary
        summary_es = soup_es.find("meta",
id="metaDcDescription")["content"]
        summary_eu = soup_eu.find("meta",
id="metaDcDescription")["content"]

        #...

        #Get Address
        place_es = soup_es.find('li',
id="ctl09_div_localizacion").find('strong').contents[0]
        place_eu = soup_eu.find('li',
id="ctl09_div_localizacion").find('strong').contents[0]
        place = geolocalize(place_es, place_eu)

        #organizador
        org_str = soup_es.find('li',
id="ctl09_div_organizador").find('strong').contents[0]

```

Figura 7.12: Código de parseo HTML del módulo de obtención de datos

4. La cuarta y última parte de código de este módulo realiza las llamadas a los módulos de Geolocalización y Publicación en Google Calendar, para recoger los datos que estos devuelven para su introducción en la base de datos.
 - Se le llama al módulo de Geolocalización pasándole la localización parseada tanto en euskera como en castellano. Éste devuelve un objeto indicando la dirección real y la latitud y longitud exacta de la localización.
 - Se llama al módulo de Google Calendar pasándole todos los datos del evento. Éste módulo devuelve las URLs del evento dentro de ambos calendarios, para su posterior exportación, las cuales deben guardarse también en la base de datos.
 - Se guarda toda la información en un objeto de tipo Event, que es el que se guardará en la base de datos.

```

        #Geolocalization
        place = geolocalize(place_es, place_eu)

        #Insert in Google Calendar
        gcalurl_es, gcalurl_eu =
insertInGCalendar(event=event)
        event.gcalendar_url_es = gcalurl_es
        event.gcalendar_url_eu = gcalurl_eu

        event = Event()
        event.event_id = evid
        event.name_es = str(name_es)
        event.name_eu = str(name_eu)
        event.start_date = start_date

        #...

        event.address = place.address
        event.lat = place.lat
        event.lng = place.lng
        event.prov = place.prov

        #Set data from foreign keys
        event.set_foreign_data(str(act_str), str(topic_str),
str(coord_str), str(org_str), str(lang_str))

        #Save it in database
        event.save()

        print "EVENT " + str(event.event_id) + " OK!" + "\n" +
event.url + "\n"

```

Figura 7.13: Código de guardado en base de datos del módulo de obtención de datos

7.5.3 Implementación del módulo de geolocalización de eventos

En la fase de diseño de este módulo se definió que el funcionamiento se basaría en la identificación de ciertas cadenas frecuentes en las direcciones recogidas, y en caso de no coincidencia, se utilizaría la API de Geolocalización de Google Maps.

La **API de Geolocalización de Google Maps** es un servicio web, al que se accede realizando una petición HTTP a una URL determinada incluyendo ciertos parámetros de búsqueda. La respuesta se devuelve en un documento con una estructura determinada, que puede estar en formato XML o bien en formato JSON, a elección del programador.

El módulo de geolocalización tiene como parámetros de entrada las localizaciones tanto en euskera como en castellano. A continuación se explicará detalladamente el código programático de este módulo:

1. En la primera parte de código, además de definir la URL de acceso al web service y de la definición de una expresión regular para la búsqueda de códigos postales en las localizaciones, se busca en las cadenas de localización lugares habituales en los que se celebran los eventos:

```
def geolocalize(address_es, address_eu):
    #re pattern for getting postal code
    pat_cp = re.compile('\d{5}')
    gmaps_url = "http://maps.google.com/maps/api/geocode/xml?address="
    a = str(address_es).lower()
    place = None

    if (a.find("parque tecnol") != -1):
        if (a.find("zamudio") != -1) or (a.find("bizkaia") != -1) or
(a.find("vizcaya") != -1):
            place = Place.objects.get(name_es = "Parque Tecnológico de
Zamudio")
        elif (a.find("guipuzcoa") != -1) or (a.find("gipuzkoa") != -1)
or (a.find("guipúzcoa") != -1) or (a.find("donosti") != -1) or
(a.find("san seb") != -1):
            place = Place.objects.get(name_es = "Parque Tecnológico de
San Sebastián")
        elif (a.find("álava") != -1) or (a.find("alava") != -1) or
(a.find("araba") != -1) or (a.find("miñano") != -1):
            place = Place.objects.get(name_es = "Parque Tecnológico de
Álava")
        elif (a.find("spri") != -1) or (a.find("promoción y
reconversión") != -1):
            place = Place.objects.get(name_es = "SPRI")
        elif (a.find("empresa digitala") != -1):
            if (a.find("araba") != -1):
                place = Place.objects.get(name_es = "Araba Empresa
Digitala")
            elif (a.find("zamudio") != -1) or (a.find("bizkaia") != -1):
                place = Place.objects.get(name_es = "Bizkaia Empresa
Digitala")
            elif (a.find("garaia") != -1) or (a.find("arrasate") != -1) or
(a.find("mondragón") != -1):
                place = Place.objects.get(name_es = "Garaia Empresa
Digitala")
            elif (a.find("donosti") != -1) or (a.find("miramón") != -1) or
(a.find("miramon") != -1) or (a.find("san sebast") != -1):
                place = Place.objects.get(name_es = "Miramón Empresa
Digitala")
            elif (a.find("polo") != -1) and (a.find("innovación") != -1):
                place = Place.objects.get(name_es = "Polo de Innovación
Garaia")
```

Figura 7.14: Código de búsqueda de localizaciones frecuentes en el módulo de geolocalización

- En la segunda parte, y si no se ha encontrado la localización entre los lugares frecuentes, se accede a la API de Geolocalización de GMaps primero introduciendo como valor la dirección completa, y en caso de error, el código postal. En caso de que no se consiga geolocalizar la dirección, se devuelve una geolocalización errónea y se manda un email de aviso a los administradores del sistema.

```

else:
    #Look directly at GMaps WS with address
    try:
        #Connect to the API
        resp = urllib2.urlopen(gmaps_url + str(address).replace("
", "+") + "+españa&sensor=false")

        #Parse the response with BeautifulSoup
        xml = resp.read()
        soup = BeautifulSoup(xml)

        #If it finds anything for it, OK, else, look by C.P.
        if (soup.status.string == 'OK'):
            place = Place( address =
str(soup.formatted_address.string),
                           lat =
float(soup.geometry.location.lat.string),
                           lng =
float(soup.geometry.location.lng.string),
                           prov =
get_province(str(soup.findAll("address_component")[2].long_name.string
)))
        except:
            pass

        if not place:
            #Look at GMaps WS by Postal Code
            cp = pat_cp.findall(a)
            if cp:
                resp = urllib2.urlopen(gmaps_url + str(cp)
+"+españa&sensor=false")

                #...

            #If place isn't set yet, fill it with default "Not Found" place
            if not place:
                place = Place( address = "Not Found",
                               lat = 0,
                               lng = 0,
                               prov = get_province("Otras"))
            #Send email to the administrators informing of the error
            send_mail_to_admins(address_es, address_eu)

            #Put as names given addresses
            place.name_es = str(address_es)
            place.name_eu = str(address_eu)
            return place

```

Figura 7.15: Código de búsqueda de direcciones en GMaps Geolocalization API

7.5.4 Implementación del módulo de actualización de Google Calendar

Para la implementación de este módulo se hace uso del cliente Python para la API de Google Data. El código del módulo se basa en una serie de llamadas de conexión a la API y de adición de eventos:

Para la conexión y autenticación en Google Data:

```
cal_client =
gdata.calendar.client.CalendarClient(source='InnovAgenda.com')

cal_client.ClientLogin("info@innovagenda.com",
"lacontraseñanoosimporta;)", cal_client.source)
```

Figura 7.16: Código de autenticación en GData API

Para la creación del evento:

```
event_es = gdata.calendar.data.CalendarEventEntry()
event_es.title = atom.data.Title(text=force_unicode(event.name_es))
event_es.content =
    atom.data.Content(text=force_unicode(event.summary_es))
event_es.where.append(gdata.data.Where(value=force_unicode(event.local
ization_es)))
event_es.when.append(gdata.data.When(start=start_time, end=end_time))
```

Figura 7.17: Código de preparación del evento de GCalendar

Para la inserción del evento en el calendario que se especifique mediante la URI. Esta URI la provee Google Calendar en su zona de configuración:

```
ev_es = cal_client.InsertEvent(event_es,
    insert_uri="http://www.google.com/calendar/feeds/innovagenda.com
_d3g76urbacsgv2q8qducml3og@group.calendar.google.com/private/fu
11")
```

Figura 7.18: Código de creación del evento de GCalendar

El objeto devuelto por la llamada a "InsertEvent" es el que contiene la URL del evento que se el módulo de actualización de Google Calendar devuelve.

7.5.5 Implementación del módulo de comprobación de eventos finalizados

Este módulo es un simple script que haciendo uso de las librerías de Django para el acceso a la base de datos comprueba que eventos han finalizado ya mediante la comparación con la fecha y hora en la que se ejecuta.

7.5.6 Implementación del módulo de publicación

El módulo de publicación se implementa mediante Vistas (Views) y Plantillas (Templates). De las plantillas se implementarán dos versiones, una para móviles y otra para la versión de escritorio, mientras que las vistas serán comunes para ambas versiones. La renderización de las versiones de las plantillas corre a cargo de la aplicación de Django-Mobile, que será quien detecte la plataforma y cargue la plantilla de la versión correspondiente, siempre que las plantillas de ambas versiones se llamen igual.

Para determinar qué método de vista se tiene que ejecutar, se crea el archivo “urls.py”, que contiene las relaciones entre patrones de URLs (basados en expresiones regulares) y vistas. La relación entre patrones y vistas para InnovAgenda es la siguiente:

```
urlpatterns = patterns('',
    # Innovagenda URLs
    (r'^$', 'website.views.index'),
    (r'^map/$', 'website.views.mapa'),
    (r'^events/$', 'website.views.all_events'),
    (r'^calendar/$', 'website.views.calendar'),
    (r'^search/', 'website.views.customsearch'),
    (r'^event/(?P<eventid>\d+)/$', 'website.views.single_event'),
    (r'^contact/$', 'website.views.contact'),
    (r'^thanks/$', 'website.views.thanks'),
    (r'^apidocs/$', 'website.views.apidocs'),
)
```

Figura 7.19: Relación entre patrones de URLs y Vistas (urls.py)

A continuación se explicará la forma en que se han implementado las vistas y las plantillas HTML:

Views

Las vistas se implementan en el archivo “views.py”. La implementación de la vista correspondiente al inicio de la aplicación es la siguiente:

```
def index(request):
    if django_mobile.get_flavour(request) == 'mobile':
        return render_to_response('index.html',
            context_instance=RequestContext(request))
    else:
        coming_events =
Event.objects.filter(finished=False).order_by('start_date')[:20]
        repeated_points = {}
        for evx in coming_events:
            evp = (evx.lat, evx.lng)
            if (evp in repeated_points):
                repeated_points[evp] += 1
            else:
                repeated_points[evp] = 1
```

```

        return render_to_response('map.html', {'comming_events':
comming_events, 'repeated_points': repeated_points},
context_instance=RequestContext(request))

```

Figura 7.20: Implementación de la vista de inicio

En ella se devuelve la plantilla de inicio en caso de que se visite la versión móvil, y la plantilla del mapa en caso de la versión de escritorio, tal y como se había diseñado. Para la visualización del mapa es necesario enviar a la plantilla los datos de los próximos eventos, por lo que se accede a la base de datos antes de mandar renderizar la plantilla HTML requerida.

El resto de vistas siguen el mismo funcionamiento que la de inicio:

- La de visualización de mapas no tiene comportamiento especial para móviles, y realiza las mismas acciones que la vista de inicio de la versión de escritorio.
- La de visualización del calendario, al igual que la de la documentación de la API, renderiza la plantilla directamente, ya que no es necesario pasar ninguna información de los modelos.
- La de la lista de eventos recoge todos los eventos no finalizados y los manda a la plantilla ordenados por fecha de inicio del evento.

A excepción de la visualización de un evento único, de contacto y de búsqueda, ya que estos métodos incluyen implementación para antes de la interacción y para después de la interacción:

- En la vista de evento, el comportamiento previo a la interacción recoge toda la información del evento seleccionado y la manda. El comportamiento posterior depende de los parámetros HTTP GET enviados a la misma URL. El comportamiento posterior es el referente a la exportación del contenido del evento a los formatos semánticos disponibles. Para la exportación del contenido se hace uso de RDFLib, que parsea la propia página del evento para devolver el archivo en el formato que se especifique. El código es el siguiente:

```

def single_event(request, eventid):
    format = request.GET.get('markFormat')
    export = request.GET.get('exportAs')

    #Check if the exportation is requested
    if export:
        g = rdflib.Graph()
        response = HttpResponse()
        #Parse depending of the version of the event markup, RDFa or
        MicroData
        if format and format == 'rdfa':
            g.parse('http://' + str(request.META['HTTP_HOST']) + "/" +
request.LANGUAGE_CODE.lower() + "/event/" + str(eventid) +
"/?markFormat=rdfa", format="rdfa")

```

```

        else:
            g.parse('http://' + str(request.META['HTTP_HOST']) + "/" +
request.LANGUAGE_CODE.lower() + "/event/" + str(eventid),
format="microdata")

            #Serialize parsed graph depending on wanted format
            if export == "rdf":
                response = HttpResponse(g.serialize(format='xml'),
mimetype='application/rdf+xml')
                response['Content-Disposition'] = 'filename=event' +
str(eventid)
                if format and format == 'rdfa':
                    response['Content-Disposition'] += '_rdfa.rdf'
                else:
                    response['Content-Disposition'] += '_schemaorg.rdf'
            elif export == "ntriples":

                #...

            return response
        else:
            #If no exportation requested return event information
            event = get_object_or_404(Event, event_id=eventid)
            if format and format == 'rdfa':
                return render_to_response('event.html', {'event': event,
'format': format}, context_instance=RequestContext(request))
            else:
                return render_to_response('event.html', {'event': event,
'format': ''}, context_instance=RequestContext(request))

```

Figura 7.21: Implementación de la vista de evento

- La vista de visualización de la plantilla de contacto es diferente porque es dependiente de un formulario.
- La vista de visualización de la plantilla de búsqueda depende de formularios específicos de la aplicación Django-Haystack.

Templates

Las plantillas HTML son documentos HTML normales, pero que contienen algunas etiquetas especiales proporcionadas por Django para dinamizar su contenido según lo que las vistas les transfieran. Un ejemplo de su uso es el siguiente:

```

{% if event.teacher %}
    <b>{% trans "Impartido por" %}</b>
    <span>{{ event.teacher }}</span><br />
{% endif %}

```

Figura 7.22: Ejemplo de manejo de datos en plantillas

En este ejemplo se puede observar la utilización de la etiqueta “if” de control de flujo, así como el manejo de datos de objetos pasados desde las vistas (“{{ event.teacher }}"). Estas etiquetas son interpretadas por Django antes de servir el HTML al navegador.

Todas las plantillas de este proyecto están escritas utilizando HTML5 y JavaScript, en conjunto con estas etiquetas especiales para mostrar el contenido según las vistas.

La mayoría de las plantillas basan su estructura en su propio contenido, y no merecen mención especial, a excepción de la de visualización del mapa, y la de información de cada evento.

- La plantilla de visualización del mapa tiene dos contenidos remarcables:
 - La utilización de HTML5+JavaScript para conseguir la localización actual del usuario.
 - El uso de la API v3 de Google Maps para JavaScript de forma dependiente del contenido de lo pasado por la vista.

La localización del usuario se consigue con el método “navigator.geolocation” que ejecuta una función Javascript en función de si consigue o no recoger la localización del usuario.

El resto de funciones son funciones de manejo de la API de JavaScript, como la creación del mapa y sus marcadores. En la función “setMarkers” se aprecia la utilización de etiquetas de Django en medio del código JavaScript.

```
<script type="text/javascript">
  var map;

  function success(position) {
    var latlng = new google.maps.LatLng(position.coords.latitude,
position.coords.longitude);
    createMap(latlng);
    setMarkers();
    createMarker(latlng, "You're here!")
  }

  function error(msg) {
    var latlng = new google.maps.LatLng(42.924252, -2.384033);
    createMap(latlng);
    setMarkers();
  }

  function createMap(latlng) {
    var myOptions = {
      zoom: 9,
      center: latlng,
      mapTypeId: google.maps.MapTypeId.ROADMAP
    };
    map = new
google.maps.Map(document.getElementById("map_canvas_big"), myOptions);
```

```

}

function createMarker(latlng, text) {
    var marker = new google.maps.Marker({
        position: latlng,
        title: text
    });
    marker.setMap(map);
}

function createMarkerWithContent(lat, lng, text, image,
contentString) {
    var latlng = new google.maps.LatLng(lat, lng);

    var marker = new google.maps.Marker({
        position: latlng,
        title: text,
        icon: image,
        map: map
    });

    var infowindow = new google.maps.InfoWindow({
        content: contentString
    });

    google.maps.event.addListener(marker, 'click', function() {
        infowindow.open(map, marker);
    });
}

function setMarkers() {
    var a = []
    a = {{ repeated_points.values }};
    {% for point in repeated_points %}
    var contentString =
        '<section id="content">' +
        '<article>' +
        {% for event in comming_events %}
        {% if event.lat == point.0 and event.lng == point.1 %}
        '<h3>{{ event.name }}</h3>' +
        '<p>{{ event.start_date.isoformat }} |'
        {{ event.activity_type }}<p>' +
        '<a'
        href="/{{ LANGUAGE_CODE }}/event/{{ event.event_id }}">Más info</a>' +
        {% endif %}
        {% endfor %}
        '</article>' +
        '</section>';
    var i = {{ forloop.counter0 }};
    var image = '{{ MEDIA_URL }}/images/gmaps/IA_' + a[i] + '.png';
    createMarkerWithContent({{ point.0 }}, {{ point.1 }}, "", image,
contentString);
    {% endfor %}
}

if (navigator.geolocation) {
    navigator.geolocation.getCurrentPosition(success, error);
} else {

```

```

    error('');
}
</script>

```

Figura 7.23: Script JS de geolocalización del usuario y uso de la API JS de GMaps

La visualización de la plantilla resultante para el mapa es la siguiente:



Figura 7.24: Interfaz de usuario del sub-módulo de visualización del mapa (versión escritorio y móvil)

- La plantilla de visualización de eventos también merece mención especial porque está implementada de tal forma que introduce un marcado semántico u otro (HTML5 MicroData o RDFa), según el parámetro GET que especifica su formato. Un pequeño ejemplo del marcado de esta plantilla:

```

{% ifequal format "rdfa" %}
    <html version="HTML+RDFa 1.1" lang="{{ LANGUAGE_CODE }}"
    xmlns="http://www.w3.org/1999/xhtml"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:cal="http://www.w3.org/2002/12/cal/icaltzd#"
    xmlns:geo="http://www.w3.org/2003/01/geo/wgs84_pos#">

{% else %}
    <html lang="{{ LANGUAGE_CODE }}">
{% endifequal %}

...

{% ifequal format "rdfa" %}
    <div
about="{{ request.META.HTTP_HOST }}{{ request.get_full_path }}"
instanceof="cal:Vevent">
{% else %}
    <div itemscope itemtype="http://schema.org/Event"
itemid="{{ request.META.HTTP_HOST }}{{ request.get_full_path }}">
{% endifequal%}

```

```
...
<span {% ifequal format "rdfa" %}property="cal:description"{% else
%}itemprop="description"{% endifequal %}>{{ event.summary }}</span>
...
```

Figura 7.25: Ejemplo de marcado semántico múltiple en plantilla de evento

Además de por el marcado semántico, esta vista se caracteriza por la inclusión de elementos de la llamada Web 2.0, como la inclusión de comentarios (herramienta proporcionada por Disqus, plataforma de comentarios para la creación de comunidades en torno a webs, o la compartición de la asistencia al evento en las redes sociales Twitter, Facebook y Google+:

```
<!-- Facebook -->
<iframe
src="http://www.facebook.com/plugins/like.php?href=http://{ request.M
ETA.HTTP_HOST }}/es/event/{{ event.event_id }}&send=false&layo
ut=box_count&width=350&show_faces=true&action=recommend&am
p;colorscheme=light&font=verdana&height=90" scrolling="no"
frameborder="0" style="border:none; overflow:hidden; width:100px;
height:110px;" allowTransparency="true"></iframe>

<!-- Twitter -->
<a href="http://twitter.com/share" class="twitter-share-button" data-
count="vertical" data-lang="es">Tweet</a><script
type="text/javascript"
src="http://platform.twitter.com/widgets.js"></script>
<!-- Google+ -->
&nbsp;<g:plusone size="tall"></g:plusone>
```

Figura 7.26: Código de los botones sociales de Twitter, Facebook y G+

```
<script type="text/javascript">
  var disqus_shortcode = 'innovagenda';
  var disqus_identifier = 'disqus_ev_{{ event.event_id }}';
  var disqus_url =
'http://{ request.META.HTTP_HOST }}/es/event/{{ event.event_id }}';

  (function() {
    var dsq = document.createElement('script'); dsq.type =
'text/javascript'; dsq.async = true;
    dsq.src = 'http://' + disqus_shortcode +
'.disqus.com/embed.js';
    (document.getElementsByTagName('head')[0] ||
document.getElementsByTagName('body')[0]).appendChild(dsq);
  })();
</script>
<noscript>Please enable JavaScript to view the <a
href="http://disqus.com/?ref_noscript">comments powered by
Disqus.</a></noscript>
```



```
<a href="http://disqus.com" class="dsq-brlink">blog comments
powered by <span class="logo-disqus">Disqus</span></a>
```

Figura 7.27: Código de la inserción del sistema de comentarios Disqus

La visualización de la plantilla resultante para la visualización de un evento, incluyendo las implementaciones descritas es la siguiente:

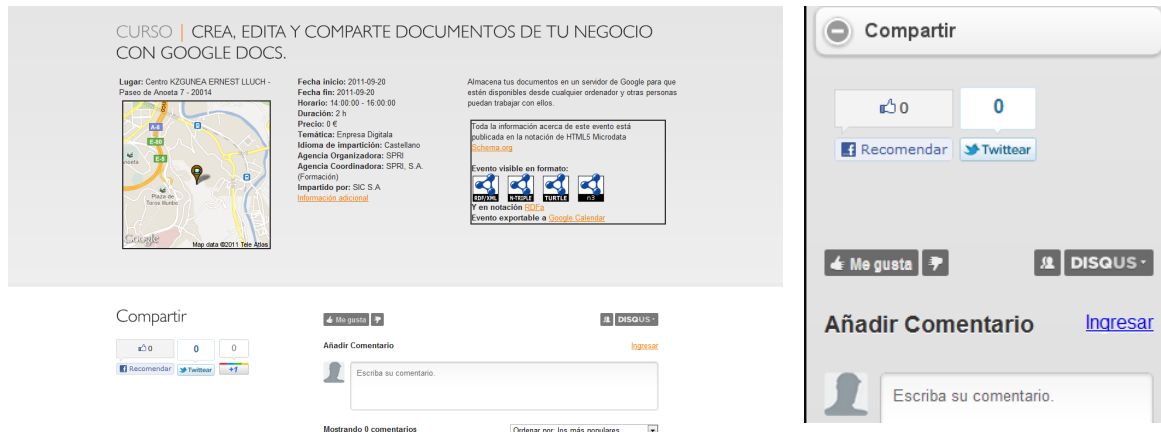


Figura 7.28: Interfaz de usuario del sub-módulo de visualización de un evento (versión escritorio y móvil)

7.5.7 Implementación del módulo de administración

La implementación del módulo de administración se ha realizado gracias a la herramienta Django Admin incluida en Django por defecto. Para un uso simple de ella, tan solo hay que decir a Django que va a ser utilizada, y crear un archivo “admin.py” especificando los modelos que van a poder modificarse desde él.

Para el desarrollo de esta aplicación, se han realizado una serie de modificaciones en el sitio de administración por defecto:

- Personalización de la plantilla inicial para la inclusión del acceso a la zona de traducción. La zona de traducción la proporcionará Django-Rosetta. Estas personalizaciones se hacen sobrescribiendo las plantillas por defecto de Django Admin.
- Modificación de la forma en que se visualizan los campos de la base de datos para incluir títulos más descriptivos para los campos. Estas modificaciones se realizan incluyendo modificaciones en la especificación de los modelos en models.py.
- Inclusión de opciones de filtrado, búsqueda y ordenación. Estas modificaciones se realizan incluyendo modificaciones en la especificación de los modelos en models.py.

7.5.8 Implementación del módulo de API

El módulo de API se ha implementado utilizando la aplicación Django-Tastypie descrita anteriormente. Para la habilitar la utilización de esta aplicación en el proyecto tan solo es necesaria incluirla en la lista de aplicaciones instaladas del archivo de configuración.

Para configurar la API es necesario crear un archivo de configuración llamado “api.py”. En este archivo se especifican los modelos a los que se va a poder acceder, los métodos de acceso, los campos a devolver, etc. Para crear la API de InnovAgenda de acuerdo con el diseño especificado, se configuró Tastypie de la siguiente manera:

- Se especificó que Event era el modelo al que se quería acceder desde la API.
- Se habilitó la búsqueda de eventos por nombre, fecha de inicio y fecha fin; y se especificaron las opciones de filtrado (mayor que, menor que, contiene, etc.).
- Se configuró el método de acceso para posibilitar únicamente las peticiones de tipo GET (es decir, únicamente las de consulta) y para que éste estuviera disponible sin requerir ningún tipo de autenticación (es decir, se deshabilitó el sistema de autenticación para la API).
- Se reescribió el método “dehydrate” para añadir nuevas opciones a la API. “Dehydrate” es el método al que llama Tastypie tras la búsqueda y filtrado de la información a la que se accede en cada petición, pero antes de la serialización de la salida. Este método recibe un objeto con el contenido filtrado, y devuelve dicho objeto con los contenidos a serializar, por lo que mediante su redefinición se pueden añadir o excluir contenidos justo antes de crear el documento de salida. Así pues, se redefinió este método para que la API de InnovAgenda filtrara la información de salida, además de por contenido, por idioma, es decir, que admitiera un nuevo parámetro “lang” para definir el idioma de salida del documento a recibir. Para ello, se añadieron a las opciones de filtrado la exclusión de los campos que contuvieran contenido localizado, para añadirlos más tarde en el método reescrito:

```
def dehydrate(self, bundle):
    lang = bundle.request.GET.get('lang', 'both')
    if lang == 'es':
        bundle.data['summary_es'] = bundle.obj.summary_es
        bundle.data['localization_es'] =
bundle.obj.localization_es
        bundle.data['url_es'] = bundle.obj.url_es
    elif lang == 'eu':
        bundle.data['summary_eu'] = bundle.obj.summary_eu
        bundle.data['localization_eu'] =
bundle.obj.localization_eu
        bundle.data['url_eu'] = bundle.obj.url_eu
    else:
        bundle.data['summary_es'] = bundle.obj.summary_es
        bundle.data['summary_eu'] = bundle.obj.summary_eu
```

```

        bundle.data['localization_es'] =
bundle.obj.localization_es
        bundle.data['localization_eu'] =
bundle.obj.localization_eu
        bundle.data['url_es'] = bundle.obj.url_es
        bundle.data['url_eu'] = bundle.obj.url_eu
    return bundle

```

Figura 7.29: Implementación del método de filtrado redefinido para el módulo de API

- Se creó un nuevo serializador para Tastypie para dotar a la API de InnovAgenda de serialización en formato semántico. Tastypie ofrece la posibilidad de crear nuevos serializadores a formatos diferentes de los que trae por defecto, mediante la extensión a la clase “tastypie.serializers.Serializer”. Se decidió diseñar el algoritmo de serialización de la siguiente manera:

1. Obtener el Identificativo de cada evento a serializar.
2. A partir de ese identificador, parsear mediante RDFLib la versión RDFa del HTML de dicho evento dentro de InnovAgenda.
3. Juntar los parseos de los diferentes RDF/XML-s generados para cada evento en un solo grafo a devolver.
4. Serializar dicho grafo en versión RDF/XML.

Así pues, el código resultante para dicho algoritmo fue el siguiente:

```

class RDFSerializer(Serializer):
    formats = ['json', 'jsonp', 'xml', 'yaml', 'html', 'plist', 'rdf']
    content_types = {
        'json': 'application/json',
        'jsonp': 'text/javascript',
        'xml': 'application/xml',
        'yaml': 'text/yaml',
        'html': 'text/html',
        'plist': 'application/x-plist',
        'rdf': 'application/rdf+xml',
    }

    def to_rdf(self, data, options=None):
        g = rdflib.Graph()
        options = options or {}
        simple_data = self.to_simple(data, options)
        # Parse Dictionary for getting list of Objects
        for key in simple_data.keys():
            # Parse list for getting every dictionary
            if (get_type_string(simple_data[key]) == 'list'):
                for obj in simple_data[key]:
                    # Parse each object's dictionary / string
                    if (get_type_string(obj) == 'hash'):
                        # Parse RDFa version of event's page in Innovagenda

```

```

        g.parse("http://www.innovagenda.com/es/event/" +
str(obj["event_id"]) + "?markFormat=rdfa", format="rdfa")

    return StringIO.StringIO(g.serialize())

```

Figura 7.30: Código del serializador RDF para Tastypie

7.5.9 Implementación de la internacionalización y localización del sitio

La traducción del sitio se ha realizado siguiendo dos procesos, el de internacionalización (adaptación a I18N) y el de localización (L10N):

Internacionalización (I18N)

La internacionalización es el proceso de adaptación de la aplicación para su posterior localización a diferentes idiomas. En resumen, durante este proceso se especifica que partes del proyecto van a ser localizadas.

Para las tareas de internacionalización de plantillas, las partes de código que van a ser traducidas se marcan con las etiquetas personalizadas {% trans %} y {% blocktrans %} que provee el módulo I18N de Django.

Para la internacionalización de los modelos se ha utilizado la aplicación reutilizable Django-Transmeta. Gracias a esta aplicación se pueden especificar los campos de cada modelo que van a recibir traducción, y Transmeta se encarga de crear nuevos campos personalizados para cada lenguaje al que se va a traducir.

Localización (L10N)

Las tareas de localización se centran en especificar los idiomas a los que se va a traducir la aplicación, y en la traducción de los campos marcados para internacionalización.

Los idiomas a los que va a ser traducida la aplicación se especifican en el archivo de configuración de Django. Las propias herramientas de traducción de Django se encargan de crear los catálogos de traducción para las etiquetas marcadas en la etapa de internacionalización. Para la traducción de estos elementos se usa la aplicación Django-Rosetta, que detecta los catálogos de traducción y proporciona una interfaz parecida a la de administración para facilitar el proceso de traducción a los diferentes idiomas. De esta manera se ha realizado la localización de las plantillas.

La localización de los modelos se ha realizado automáticamente en la inclusión de valores en la base de datos, ya los campos que Django-Transmeta crea para la traducción se completan a la vez que todo el resto de campos al ejecutar el módulo de obtención de datos.

La localización de las URLs, es decir, la inclusión de los códigos de idiomas en las URLs del proyecto, se ha realizado mediante Django-Localeurl.

8. PLANIFICACIÓN

En este capítulo se detalla la forma en la que se ha organizado la realización de las diferentes fases del proyecto que se han ido desgranando en los capítulos anteriores del presente documento. Se estudiará la planificación del trabajo desde un punto de vista temporal y logístico, especificando las tareas realizadas, los roles del equipo de trabajo, el plan de trabajo mantenido, las cargas de trabajo separadas por tareas y por recursos, y los diagramas de Gantt y de precedencia de tareas.

8.1 TAREAS

A.1 - Organización y control

A.1.1 - Organización

Actividad mediante la que se define la planificación, asignación y lanzamiento de las diferentes fases del proyecto.

A.1.2 - Pilotaje

Seguimiento y control del desarrollo del proyecto, que permita la detección y solución lo más rápida posible de problemas que puedan surgir.

A.2 - Análisis

A.2.1 – Reuniones

Reuniones entre el director del proyecto y el analista para la definición del objetivo del proyecto y sus requisitos.

A.2.2 – Especificación de requisitos del sistema

Redacción del documento que especifica qué funcionalidades deben ser implementadas.

A.3 – Investigación del Estado del Arte

A.3.1 – Investigación acerca de la Web Semántica

Trabajo de investigación acerca de la evolución de la web, y del funcionamiento, los estándares y las herramientas que rodean a la web semántica.

A.3.2 – Estudio de los Datos Abiertos en la Comunidad Autónoma Vasca

Trabajo de estudio acerca de las fuentes de datos abiertos proporcionadas por las instituciones vascas y de sus métodos de publicación.

A.3.3 – Investigación acerca de web adaptada a móviles

Trabajo de investigación acerca de las alternativas para publicar páginas web de forma que estén adaptadas a su visibilidad correcta desde sistemas en movilidad.

A.3.4 – Estudio acerca de mashups y posibles fuentes de datos

Trabajo de estudio acerca del concepto de mashup, así como de las fuentes de datos que pueden aprovecharse para el desarrollo de uno, poniendo especial hincapié acerca de la publicación de mapas y calendarios.

A.4 – Diseño del sistema

A.4.1 – Selección de fuentes de datos y formatos de publicación para la aplicación

Establecimiento de las fuentes de datos definitivas de las que va a obtener información el mashup, así como de los formatos de publicación que va a utilizar.

A.4.2 – Diseño de la arquitectura de la aplicación

Definición de la estructura modular sobre la que se va a construir la aplicación, así como de la estructura de la base de datos.

A.4.3 – Diseño de la interacción con el usuario

Definición de los modelos de interacción con el usuario, así como de los formatos de las interfaces gráficas.

A.5 – Selección e instalación de la plataforma de desarrollo

A.5.1 – Selección de la plataforma de desarrollo

Selección de software y hardware que se va a utilizar para el desarrollo de la aplicación.

A.5.2 - Instalación de los puestos de trabajo

Instalación y configuración del equipo en el que se desarrollará la aplicación web, incluyendo el servidor y la base de datos de desarrollo.

A.6 - Desarrollo del módulo de obtención de datos de Euskadi+Innova

A.6.1 – Investigación de la fuente de datos

Estudio del formato en el que se publican los datos en Euskadi+Innova, así como de la estructura de su web, tanto interna como externa.

A.6.2 - Definición del módulo

Diseño del algoritmo de recogida de datos.

A.6.3 - Implementación del módulo

Desarrollo del sistema en lenguaje Python y diversas librerías para este lenguaje. La implementación incluirá los métodos de conexión y descarga de datos desde el RSS y los HTML de eventos.

A.7 - Desarrollo del módulo de geolocalización de eventos

A.7.1 – Investigación de la API de Google Maps

Estudio la forma en la que se puede acceder a la información de la API de Geolocalización de Google Maps.

A.7.2 – Investigación del formato de las direcciones en Euskadi+Innova

Estudio la forma en la que se publican las direcciones de las localizaciones en la web de Euskadi+Innova.

A.7.3 - Definición del módulo

Diseño del algoritmo de geolocalización de eventos.

A.7.4 - Implementación del módulo

Desarrollo del sistema en lenguaje Python. La implementación incluirá los métodos de conexión con la API y de búsqueda de la localización en ella.

A.8 - Desarrollo del módulo de actualización de Google Calendar

A.8.1 – Investigación de la API de Google Calendar

Estudio la forma en la que se pueden publicar eventos en los calendarios de Google Calendar.

A.8.2 - Definición del módulo

Diseño del algoritmo de publicación de eventos en Google Calendar.

A.8.3 - Implementación del módulo

Desarrollo del sistema en lenguaje Python. La implementación incluirá los métodos de conexión a la API y publicación de eventos.

A.9 - Desarrollo del módulo de comprobación de eventos finalizados

A.9.1 - Definición del módulo

Diseño del algoritmo de comprobación de fechas de eventos.

A.9.2 - Implementación del módulo

Desarrollo del sistema en lenguaje Python. La implementación incluirá el método de comprobación de eventos finalizados.

A.10 - Desarrollo del módulo de publicación

A.10.1 – Investigación de la API de los servicios de publicación a utilizar

Estudio la forma en la que se puede publicar información de Google Maps, Google Calendar, Disqus, Twitter, Facebook, etc.

A.10.2 – Investigación acerca de la publicación en formato semántico

Investigación acerca de la forma de conseguir publicar la información en los formatos de la web semántica requeridos.

A.10.3 - Definición del módulo

Diseño de los algoritmos de publicación de eventos.

A.10.4 - Implementación del módulo

Desarrollo del sistema en lenguaje Python, gracias al framework de desarrollo web Django. La implementación incluirá el desarrollo de la interfaz web para la

publicación de los datos, así como de los distintos métodos para acceder a los datos requeridos en cada momento de la base de datos.

A.11 - Desarrollo del módulo de administración

A.11.1 - Implementación del módulo

Desarrollo del módulo gracias a la utilidad de sitios de administración proporcionada por el framework de desarrollo web Django, añadiéndole las modificaciones pertinentes para la personalización del sitio y la adición del sitio de traducción.

A.12 - Desarrollo del módulo de API

A.12.1 – Diseño de las URLs del módulo

Diseño del formato de las diferentes URLs que van a servir para acceder a los datos de la base de datos a través de este módulo.

A.12.1 – Diseño del serializador semántico

Diseño del algoritmo para serializar información en formato RDF.

A.12.1 - Implementación del módulo

Desarrollo del módulo en lenguaje Python y gracias a las utilidades que proporciona la librería Tastypie para el framework de desarrollo web Django, añadiéndole las modificaciones pertinentes para la personalización de URLs y la serialización semántica.

A.13 - Pruebas de la Aplicación Web

A.13.1 - Testeo de la Aplicación

Prueba de las funcionalidades de la aplicación web y de su correcto funcionamiento.

A.13.2 - Depuración de errores encontrados

Corrección de los diferentes problemas encontrados durante el testeo de la aplicación.

A.14 - Documentación del proyecto

A.14.1 - Documentación de la Aplicación Web

Redacción de la documentación del mashup semántico, indicando sus funcionalidades y su forma de uso, así como su proceso de creación.

A.15 – Paso a producción y publicación de la Aplicación Web

A.15.1 – Selección del servicio de hosting

Estudio de las diferentes alternativas para el hosting de aplicaciones web basadas en Django.

A.15.2 – Adaptación del proyecto a las características del servicio de hosting

Adaptación de la estructura y el contenido del proyecto a las características especificadas por el servicio de hosting.

A.15.3 - Subida al Servidor Web

Subida de la aplicación web al servidor web de hosting web proporcionado por DotCloud. para su publicación en Internet para el público general.

8.2 EQUIPO REAL

Teniendo en cuenta las necesidades de las tareas previamente especificadas, ha sido necesario que el equipo de trabajo haya estado formado por los siguientes perfiles:

- **Jefe del Proyecto**
 - Número de Personas: 1.
 - Función: Encargado de coordinar, aconsejar y seguir la evolución del proyecto.
 - Perfil: Ingeniero Informático con experiencia en dirección de proyectos.
- **Director del Proyecto**
 - Número de Personas: 1.
 - Función: Encargado de la organización y planificación del proyecto, así como de mantener un canal de información fluido con el Jefe del Proyecto.
 - Perfil: Ingeniero Informático.
- **Analista**
 - Número de Personas: 1.
 - Función: Encargado de las funciones de análisis del sistema.

- Perfil: Ingeniero Informático.
- **Programador**
 - Número de Personas: 1.
 - Función: Encargado de las funciones de diseño, desarrollo y puesta en marcha del proyecto.
 - Perfil: Ingeniero Informático.
- **Técnico informático**
 - Número de Personas: 1.
 - Función: Encargado de la instalación, configuración, administración y mantenimiento de las estaciones de trabajo.
 - Perfil: Técnico de sistemas.

Estos perfiles han sido cubiertos por 2 personas:

- Diego López-de-Ipiña González-de-Artaza, como Jefe de Proyecto.
- Jon Lázar Aduna, como Director del Proyecto, Analista, Programador y Técnico informático.

El organigrama mediante el cual los miembros del equipo se han organizado ha sido el siguiente:

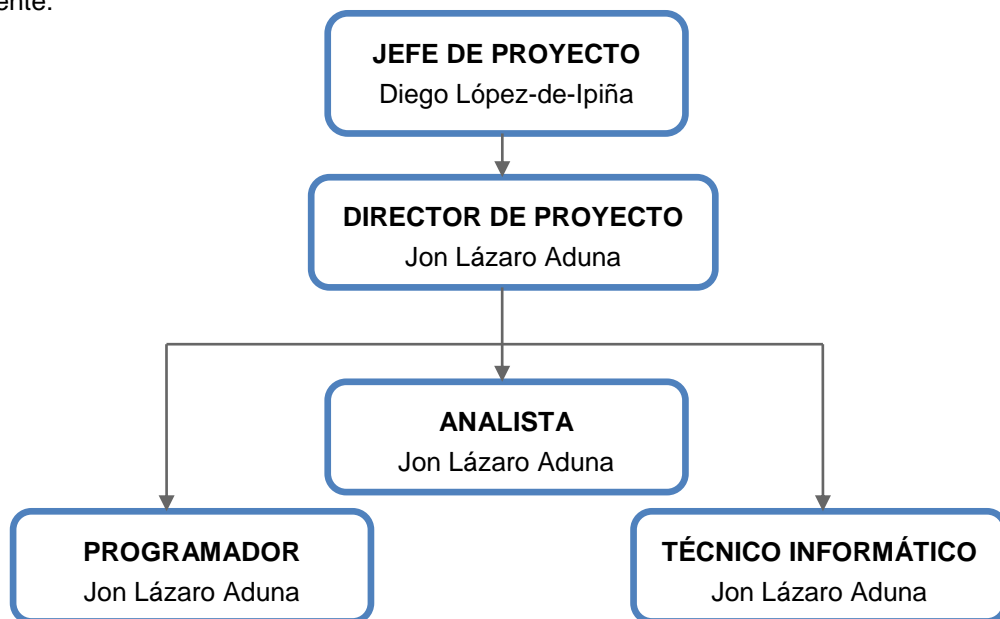


Figura 8.1: Organigrama de la organización de roles y recursos

En los siguientes puntos se mostrarán diferentes gráficos, para comprenderlos, se deben tener en cuenta las siguientes restricciones:

- La jornada laboral se considera de 4 horas entre las fechas 04/04/2011-15/05/2011, y de 8 horas del 06/06/2011 en adelante. Esto se debe a que los días de abril y mayo han coincidido con el calendario lectivo de la Universidad de Deusto.
- Los días de trabajo son de lunes a viernes.
- Además de todos los sábados y domingos, se debe tener en cuenta que los días festivos de Semana Santa (del 21 al 25 de abril) y las fechas de exámenes (del 16 de mayo al 5 de junio) se han considerado como no-laborables.
- El campo “Duración” está expresado en días laborables de 8 horas. Las horas reales de trabajo (teniendo en cuenta las diferentes jornadas laborales) son las que aparecen en el campo “Trabajo”.
- Los recursos humanos están separados por perfiles, pero como una sola persona se hace cargo de 4 de los perfiles, la carga de trabajo de cada perfil cuando trabajan en una misma tarea se divide en porcentajes estimados de trabajo de cada perfil.

8.3 PLAN DE TRABAJO

Las tablas del plan de trabajo muestran las horas de trabajo que requiere cada una de las tareas, así como las fechas en las que se han llevado a cabo:

	Identificador	Nombre de tarea	Duración	Trabajo	Comienzo	Fin	Predecesoras	Nombres de los recursos
1	A1	Planificación de las tareas del proyecto	2 días	16 horas	lun 04/04/11	jue 07/04/11		Director del Proyecto - Jon
2	A2.1	Reuniones	0,5 días	6 horas	vie 08/04/11	vie 08/04/11	1	Analista - Jon[50%]; Director del Proyecto - Jon[50%]
3	A2.2	Especificación de requisitos del sistema	1 día	8 horas	lun 11/04/11	mar 12/04/11	2	Analista - Jon
4	A3.1	Investigación acerca de la Web Semántica	3 días	24 horas	mié 13/04/11	mié 20/04/11	3	Programador - Jon
5	A3.2	Estudio de los Datos Abiertos en la CAV	2 días	16 horas	mar 26/04/11	vie 29/04/11	3	Programador - Jon
6	A3.3	Investigación acerca de la web móvil	2 días	16 horas	lun 02/05/11	jue 05/05/11	3	Programador - Jon
7	A3.4	Estudio acerca de mashups y fuentes de datos	2 días	16 horas	vie 06/05/11	mié 11/05/11	3	Programador - Jon
8	A4.1	Selección de fuentes de datos y formatos de publicación	0,5 días	4 horas	jue 12/05/11	jue 12/05/11	4;5;6;7	Director del Proyecto - Jon[25%]; Programador - Jon[75%]
9	A4.2	Diseño de la arquitectura	2 días	16 horas	vie 13/05/11	mar 07/06/11	8	Analista - Jon[30%]; Director del Proyecto - Jon[20%]; Programador - Jon[50%]
10	A4.3	Diseño de la interacción con el usuario	2 días	16 horas	mar 07/06/11	jue 09/06/11	9	Analista - Jon[30%]; Director del Proyecto - Jon[20%]; Programador - Jon[50%]
11	A5.1	Selección de la plataforma de desarrollo	1 día	8 horas	jue 09/06/11	vie 10/06/11	10	Director del Proyecto - Jon[10%]; Programador - Jon[90%]
12	A5.2	Instalación de los puestos de trabajo	0,5 días	4 horas	vie 10/06/11	vie 10/06/11	11	Técnico de Sistemas - Jon
13	A6.1	Investigación de la fuente de datos de E+I	2 días	16 horas	mar 21/06/11	mié 22/06/11	12	Programador - Jon
14	A6.2	Definición del módulo de E+I	1 día	8 horas	jue 23/06/11	jue 23/06/11	13	Programador - Jon[90%]; Analista - Jon[10%]
15	A6.3	Implementación del módulo de E+I	4 días	32 horas	vie 24/06/11	mié 29/06/11	19;22;14	Programador - Jon
16	A7.1	Investigación de la API de Google Maps	1 día	8 horas	lun 13/06/11	lun 13/06/11	12	Programador - Jon
17	A7.2	Investigación del formato de las direcciones	0,5 días	4 horas	mar 14/06/11	mar 14/06/11	16	Programador - Jon
18	A7.3	Definición del módulo de geolocalización	1 día	8 horas	mar 14/06/11	mié 15/06/11	17	Analista - Jon[10%]; Programador - Jon[90%]
19	A7.4	Implementación del módulo de geolocalización	2,5 días	20 horas	mié 15/06/11	vie 17/06/11	18	Programador - Jon
20	A8.1	Investigación de la API de Google Calendar	0,5 días	4 horas	lun 20/06/11	lun 20/06/11	12	Programador - Jon
21	A8.2	Definición del módulo de Gcalendar	0,25 días	2 horas	lun 20/06/11	lun 20/06/11	20	Programador - Jon[90%]; Analista - Jon[10%]
22	A8.3	Implementación del módulo de Gcalendar	0,25 días	2 horas	lun 20/06/11	lun 20/06/11	21	Programador - Jon
23	A9.1	Definición del módulo de eventos finalizados	0,13 días	1 hora	jue 30/06/11	jue 30/06/11	15	Programador - Jon
24	A9.2	Implementación del módulo de eventos finalizados	0,13 días	1 hora	jue 30/06/11	jue 30/06/11	23	Programador - Jon
25	A10.1	Investigación de las APIs de los servicios	1 día	8 horas	vie 08/07/11	vie 08/07/11	15	Programador - Jon
26	A10.2	Investigación de la publicación semántica	1 día	8 horas	lun 11/07/11	lun 11/07/11	15	Programador - Jon
27	A10.3	Definición del módulo de publicación	3 días	24 horas	mar 12/07/11	jue 14/07/11	25;26	Analista - Jon[10%]; Programador - Jon[90%]
28	A10.4	Implementación del módulo de publicación	10 días	80 horas	vie 15/07/11	jue 28/07/11	27;32;24;29	Programador - Jon
29	A11.1	Implementación del módulo de administración	1 día	8 horas	jue 07/07/11	jue 07/07/11	15	Analista - Jon[5%]; Programador - Jon[95%]
30	A12.1	Diseño de las URLs del módulo de API	0,75 días	6 horas	jue 30/06/11	jue 30/06/11	15	Programador - Jon
31	A12.2	Diseño del serializador semántico	0,5 días	4 horas	vie 01/07/11	vie 01/07/11	30	Programador - Jon
32	A12.3	Implementación del módulo de API	3,5 días	28 horas	vie 01/07/11	mié 06/07/11	31	Programador - Jon
33	A13.1	Testeo de la aplicación	1 día	12 horas	vie 29/07/11	vie 29/07/11	28	Analista - Jon[30%]; Director del Proyecto - Jon[70%]
34	A13.2	Depuración de errores encontrados	2 días	16 horas	lun 01/08/11	mar 02/08/11	33	Analista - Jon[10%]; Director del Proyecto - Jon[10%]; Programador - Jon[80%]
35	A14.1	Documentación	9 días	72 horas	jue 04/08/11	mar 16/08/11	38	Analista - Jon[15%]; Director del Proyecto - Jon[10%]
36	A15.1	Selección del servicio de hosting	0,25 días	2 horas	mié 03/08/11	mié 03/08/11	34	Director del Proyecto - Jon[10%]; Programador - Jon[10%]
37	A15.2	Adaptación del proyecto al hosting	0,5 días	4 horas	mié 03/08/11	mié 03/08/11	36	Programador - Jon
38	A15.3	Subida al hosting	0,25 días	2 horas	mié 03/08/11	mié 03/08/11	37	Programador - Jon

Figura 8.1: Diagrama de plan de trabajo

8.4 CARGAS REALES DE TRABAJO POR TAREA

Nombre de tarea	Trabajo	Duración	Detalles															
			D	L	M	X	J	V	S	D	L	M	X	J	V	S	D	
1	Planificación de las tareas del proyecto	16 horas 2 días	Trab.															
2	Reuniones	6 horas 0,5 días	Trab.															
3	Especificación de requisitos del sistema	8 horas 1 día	Trab.															
4	Investigación acerca de la Web Semántica	24 horas 3 días	Trab.															
5	Estudio de los Datos Abiertos en la CAV	16 horas 2 días	Trab.															
6	Investigación acerca de la web móvil	16 horas 2 días	Trab.															
7	Estudio acerca de mashups y fuentes de datos	16 horas 2 días	Trab.															
8	Selección de fuentes de datos y formatos de publicación	4 horas 0,5 días	Trab.															
9	Diseño de la arquitectura	16 horas 2 días	Trab.															
10	Diseño de la interacción con el usuario	16 horas 2 días	Trab.															
11	Selección de la plataforma de desarrollo	8 horas 1 día	Trab.															
12	Instalación de los puestos de trabajo	4 horas 0,5 días	Trab.															
13	Investigación de la fuente de datos de EH	16 horas 2 días	Trab.															
14	Definición del módulo de EH	8 horas 1 día	Trab.															
15	Implementación del módulo de EH	32 horas 4 días	Trab.															
16	Investigación de la API de Google Maps	8 horas 1 día	Trab.															
17	Investigación del formato de las direcciones	4 horas 0,5 días	Trab.															
18	Definición del módulo de geolocalización	8 horas 1 día	Trab.															
19	Implementación del módulo de geolocalización	20 horas 2,5 días	Trab.															
20	Investigación de la API de Google Calendar	4 horas 0,5 días	Trab.															
21	Definición del módulo de Gcalendar	2 horas 0,25 días	Trab.															
22	Implementación del módulo de Gcalendar	2 horas 0,25 días	Trab.															
23	Definición del módulo de eventos finalizados	1 hora 0,13 días	Trab.															
24	Implementación del módulo de eventos finalizados	1 hora 0,13 días	Trab.															
25	Investigación de las APIs de los servicios	8 horas 1 día	Trab.															
26	Investigación de la publicación semántica	8 horas 1 día	Trab.															
27	Definición del módulo de publicación	24 horas 3 días	Trab.															
28	Implementación del módulo de publicación	80 horas 10 días	Trab.															
29	Implementación del módulo de administración	8 horas 1 día	Trab.															
30	Diseño de las URLs del módulo de API	6 horas 0,75 días	Trab.															
31	Diseño del serializador semántico	4 horas 0,5 días	Trab.															
32	Implementación del módulo de API	28 horas 3,5 días	Trab.															
33	Testeo de la aplicación	12 horas 1 día	Trab.															
34	Depuración de errores encontrados	16 horas 2 días	Trab.															
35	Documentación	72 horas 9 días	Trab.															
36	Selección del servicio de hosting	2 horas 0,25 días	Trab.															
37	Adaptación del proyecto al hosting	4 horas 0,5 días	Trab.															
38	Subida al hosting	2 horas 0,25 días	Trab.															

Figura 8.2: Diagrama de cargas reales por tarea (04/04/2011-17/04/2011)

	Nombre de tarea	Trabajo	Duración	Detalles	18 abr '11							25 abr '11						
					L	M	X	J	V	S	D	L	M	X	J	V	S	D
1	+ Planificación de las tareas del proyecto	16 horas 2 días	Trab.															
2	+ Reuniones	6 horas 0,5 días	Trab.															
3	+ Especificación de requisitos del sistema	8 horas 1 día	Trab.															
4	+ Investigación acerca de la Web Semántica	24 horas 3 días	Trab.		4h	4h	4h											
5	+ Estudio de los Datos Abiertos en la CAV	16 horas 2 días	Trab.											4h	4h	4h	4h	
6	+ Investigación acerca de la web móvil	16 horas 2 días	Trab.															
7	+ Estudio acerca de mashups y Fuentes de datos	16 horas 2 días	Trab.															
8	+ Selección de fuentes de datos y formatos de publicación	4 horas 0,5 días	Trab.															
9	+ Diseño de la arquitectura	16 horas 2 días	Trab.															
10	+ Diseño de la interacción con el usuario	16 horas 2 días	Trab.															
11	+ Selección de la plataforma de desarrollo	8 horas 1 día	Trab.															
12	+ Instalación de los puestos de trabajo	4 horas 0,5 días	Trab.															
13	+ Investigación de la fuente de datos de EH	16 horas 2 días	Trab.															
14	+ Definición del módulo de EH	8 horas 1 día	Trab.															
15	+ Implementación del módulo de EH	32 horas 4 días	Trab.															
16	+ Investigación de la API de Google Maps	8 horas 1 día	Trab.															
17	+ Investigación del formato de las direcciones	4 horas 0,5 días	Trab.															
18	+ Definición del módulo de geolocalización	8 horas 1 día	Trab.															
19	+ Implementación del módulo de geolocalización	20 horas 2,5 días	Trab.															
20	+ Investigación de la API de Google Calendar	4 horas 0,5 días	Trab.															
21	+ Definición del módulo de Gcalendar	2 horas 0,25 días	Trab.															
22	+ Implementación del módulo de Gcalendar	2 horas 0,25 días	Trab.															
23	+ Definición del módulo de eventos finalizados	1 hora 0,13 días	Trab.															
24	+ Implementación del módulo de eventos finalizados	1 hora 0,13 días	Trab.															
25	+ Investigación de las APIs de los servicios	8 horas 1 día	Trab.															
26	+ Investigación de la publicación semántica	8 horas 1 día	Trab.															
27	+ Definición del módulo de publicación	24 horas 3 días	Trab.															
28	+ Implementación del módulo de publicación	80 horas 10 días	Trab.															
29	+ Implementación del módulo de administración	8 horas 1 día	Trab.															
30	+ Diseño de las URLs del módulo de API	6 horas 0,75 días	Trab.															
31	+ Diseño del serializador semántico	4 horas 0,5 días	Trab.															
32	+ Implementación del módulo de API	28 horas 3,5 días	Trab.															
33	+ Testeo de la aplicación	12 horas 1 día	Trab.															
34	+ Depuración de errores encontrados	16 horas 2 días	Trab.															
35	+ Documentación	72 horas 9 días	Trab.															
36	+ Selección del servicio de hosting	2 horas 0,25 días	Trab.															
37	+ Adaptación del proyecto al hosting	4 horas 0,5 días	Trab.															
38	+ Subida al hosting	2 horas 0,25 días	Trab.															

Figura 8.3: Diagrama de cargas reales por tarea (18/04/2011-01/05/2011)

9. PLANIFICACIÓN

	Nombre de tarea	Trabajo	Duración	Detalles													
				02 may '11							09 may '11						
				L	M	X	J	V	S	D	L	M	X	J	V	S	D
1	+ Planificación de las tareas del proyecto	16 horas 2 días															
2	+ Reuniones	6 horas 0,5 días	Trab.														
3	+ Especificación de requisitos del sistema	8 horas 1 día	Trab.														
4	+ Investigación acerca de la Web Semántica	24 horas 3 días	Trab.														
5	+ Estudio de los Datos Abiertos en la CAV	16 horas 2 días	Trab.														
6	+ Investigación acerca de la web móvil	16 horas 2 días	Trab.		4h	4h	4h	4h									
7	+ Estudio acerca de mashups y fuentes de datos	16 horas 2 días	Trab.					4h			4h	4h	4h				
8	+ Selección de fuentes de datos y formatos de publicación	4 horas 0,5 días	Trab.											4h			
9	+ Diseño de la arquitectura	16 horas 2 días	Trab.												4h		
10	+ Diseño de la interacción con el usuario	16 horas 2 días	Trab.														
11	+ Selección de la plataforma de desarrollo	8 horas 1 día	Trab.														
12	+ Instalación de los puestos de trabajo	4 horas 0,5 días	Trab.														
13	+ Investigación de la fuente de datos de EH	16 horas 2 días	Trab.														
14	+ Definición del módulo de EH	8 horas 1 día	Trab.														
15	+ Implementación del módulo de EH	32 horas 4 días	Trab.														
16	+ Investigación de la API de Google Maps	8 horas 1 día	Trab.														
17	+ Investigación del formato de las direcciones	4 horas 0,5 días	Trab.														
18	+ Definición del módulo de geolocalización	8 horas 1 día	Trab.														
19	+ Implementación del módulo de geolocalización	20 horas 2,5 días	Trab.														
20	+ Investigación de la API de Google Calendar	4 horas 0,5 días	Trab.														
21	+ Definición del módulo de Gcalendar	2 horas 0,25 días	Trab.														
22	+ Implementación del módulo de Gcalendar	2 horas 0,25 días	Trab.														
23	+ Definición del módulo de eventos finalizados	1 hora 0,13 días	Trab.														
24	+ Implementación del módulo de eventos finalizados	1 hora 0,13 días	Trab.														
25	+ Investigación de las APIs de los servicios	8 horas 1 día	Trab.														
26	+ Investigación de la publicación semántica	8 horas 1 día	Trab.														
27	+ Definición del módulo de publicación	24 horas 3 días	Trab.														
28	+ Implementación del módulo de publicación	80 horas 10 días	Trab.														
29	+ Implementación del módulo de administración	8 horas 1 día	Trab.														
30	+ Diseño de las URLs del módulo de API	6 horas 0,75 días	Trab.														
31	+ Diseño del serializador semántico	4 horas 0,5 días	Trab.														
32	+ Implementación del módulo de API	28 horas 3,5 días	Trab.														
33	+ Testeo de la aplicación	12 horas 1 día	Trab.														
34	+ Depuración de errores encontrados	16 horas 2 días	Trab.														
35	+ Documentación	72 horas 9 días	Trab.														
36	+ Selección del servicio de hosting	2 horas 0,25 días	Trab.														
37	+ Adaptación del proyecto al hosting	4 horas 0,5 días	Trab.														
38	+ Subida al hosting	2 horas 0,25 días	Trab.														

Figura 8.4: Diagrama de cargas reales por tarea (02/05/2011-15/05/2011)

	Nombre de tarea	Trabajo	Duración	Detalles	06 jun '11							13 jun '11						
					L	M	X	J	V	S	D	L	M	X	J	V	S	D
1	+ Planificación de las tareas del proyecto	16 horas 2 días	Trab.															
2	+ Reuniones	6 horas 0,5 días	Trab.															
3	+ Especificación de requisitos del sistema	8 horas 1 día	Trab.															
4	+ Investigación acerca de la Web Semántica	24 horas 3 días	Trab.															
5	+ Estudio de los Datos Abiertos en la CAV	16 horas 2 días	Trab.															
6	+ Investigación acerca de la web móvil	16 horas 2 días	Trab.															
7	+ Estudio acerca de mashups y fuentes de datos	16 horas 2 días	Trab.															
8	+ Selección de fuentes de datos y formatos de publicación	4 horas 0,5 días	Trab.															
9	+ Diseño de la arquitectura	16 horas 2 días	Trab.		8h	4h												
10	+ Diseño de la interacción con el usuario	16 horas 2 días	Trab.			4h	8h		4h									
11	+ Selección de la plataforma de desarrollo	8 horas 1 día	Trab.						4h									
12	+ Instalación de los puestos de trabajo	4 horas 0,5 días	Trab.						4h									
13	+ Investigación de la fuente de datos de E+I	16 horas 2 días	Trab.						4h									
14	+ Definición del módulo de E+I	8 horas 1 día	Trab.															
15	+ Implementación del módulo de E+I	32 horas 4 días	Trab.															
16	+ Investigación de la API de Google Maps	8 horas 1 día	Trab.									8h						
17	+ Investigación del formato de las direcciones	4 horas 0,5 días	Trab.										4h					
18	+ Definición del módulo de geolocalización	8 horas 1 día	Trab.										4h					
19	+ Implementación del módulo de geolocalización	20 horas 2,5 días	Trab.											4h	4h			
20	+ Investigación de la API de Google Calendar	4 horas 0,5 días	Trab.											4h	8h	8h		
21	+ Definición del módulo de Gcalendar	2 horas 0,25 días	Trab.															
22	+ Implementación del módulo de Gcalendar	2 horas 0,25 días	Trab.															
23	+ Definición del módulo de eventos finalizados	1 hora 0,13 días	Trab.															
24	+ Implementación del módulo de eventos finalizados	1 hora 0,13 días	Trab.															
25	+ Investigación de las APIs de los servicios	8 horas 1 día	Trab.															
26	+ Investigación de la publicación semántica	8 horas 1 día	Trab.															
27	+ Definición del módulo de publicación	24 horas 3 días	Trab.															
28	+ Implementación del módulo de publicación	80 horas 10 días	Trab.															
29	+ Implementación del módulo de administración	8 horas 1 día	Trab.															
30	+ Diseño de las URLs del módulo de API	6 horas 0,75 días	Trab.															
31	+ Diseño del serializador semántico	4 horas 0,5 días	Trab.															
32	+ Implementación del módulo de API	28 horas 3,5 días	Trab.															
33	+ Testeo de la aplicación	12 horas 1 día	Trab.															
34	+ Depuración de errores encontrados	16 horas 2 días	Trab.															
35	+ Documentación	72 horas 9 días	Trab.															
36	+ Selección del servicio de hosting	2 horas 0,25 días	Trab.															
37	+ Adaptación del proyecto al hosting	4 horas 0,5 días	Trab.															
38	+ Subida al hosting	2 horas 0,25 días	Trab.															

Figura 8.5: Diagrama de cargas reales por tarea (06/06/2011-19/06/2011)

9. PLANIFICACIÓN

	Nombre de tarea	Trabajo	Duración	Detalles	20 jun '11							27 jun '11						
					L	M	X	J	V	S	D	L	M	X	J	V	S	D
1	+ Planificación de las tareas del proyecto	16 horas 2 días	Trab.															
2	+ Reuniones	6 horas 0,5 días	Trab.															
3	+ Especificación de requisitos del sistema	8 horas 1 día	Trab.															
4	+ Investigación acerca de la Web semántica	24 horas 3 días	Trab.															
5	+ Estudio de los Datos Abiertos en la CAV	16 horas 2 días	Trab.															
6	+ Investigación acerca de la web móvil	16 horas 2 días	Trab.															
7	+ Estudio acerca de mashups y fuentes de datos	16 horas 2 días	Trab.															
8	+ Selección de fuentes de datos y formatos de publicación	4 horas 0,5 días	Trab.															
9	+ Diseño de la arquitectura	16 horas 2 días	Trab.															
10	+ Diseño de la interacción con el usuario	16 horas 2 días	Trab.															
11	+ Selección de la plataforma de desarrollo	8 horas 1 día	Trab.															
12	+ Instalación de los puestos de trabajo	4 horas 0,5 días	Trab.															
13	+ Investigación de la fuente de datos de EHI	16 horas 2 días	Trab.			8h	8h											
14	+ Definición del módulo de EHI	8 horas 1 día	Trab.						8h									
15	+ Implementación del módulo de EHI	32 horas 4 días	Trab.						8h			8h	8h	8h				
16	+ Investigación de la API de Google Maps	8 horas 1 día	Trab.															
17	+ Investigación del formato de las direcciones	4 horas 0,5 días	Trab.															
18	+ Definición del módulo de geolocalización	8 horas 1 día	Trab.															
19	+ Implementación del módulo de geolocalización	20 horas 2,5 días	Trab.															
20	+ Investigación de la API de Google Calendar	4 horas 0,5 días	Trab.		4h													
21	+ Definición del módulo de Gcalendar	2 horas 0,25 días	Trab.		2h													
22	+ Implementación del módulo de Gcalendar	2 horas 0,25 días	Trab.		2h													
23	+ Definición del módulo de eventos finalizados	1 hora 0,13 días	Trab.												1h			
24	+ Implementación del módulo de eventos finalizados	1 hora 0,13 días	Trab.												1h			
25	+ Investigación de las APIs de los servicios	8 horas 1 día	Trab.															
26	+ Investigación de la publicación semántica	8 horas 1 día	Trab.															
27	+ Definición del módulo de publicación	24 horas 3 días	Trab.															
28	+ Implementación del módulo de publicación	80 horas 10 días	Trab.															
29	+ Implementación del módulo de administración	8 horas 1 día	Trab.															
30	+ Diseño de las URLs del módulo de API	6 horas 0,75 días	Trab.												6h			
31	+ Diseño del serializador semántico	4 horas 0,5 días	Trab.													4h		
32	+ Implementación del módulo de API	28 horas 3,5 días	Trab.													4h		
33	+ Testeo de la aplicación	12 horas 1 día	Trab.															
34	+ Depuración de errores encontrados	16 horas 2 días	Trab.															
35	+ Documentación	72 horas 9 días	Trab.															
36	+ Selección del servicio de hosting	2 horas 0,25 días	Trab.															
37	+ Adaptación del proyecto al hosting	4 horas 0,5 días	Trab.															
38	+ Subida al hosting	2 horas 0,25 días	Trab.															

Figura 8.6: Diagrama de cargas reales por tarea (20/06/2011-03/07/2011)

Nombre de tarea	Trabajo	Duración	Detalles	04 jul '11							11 jul '11						
				L	M	X	J	V	S	D	L	M	X	J	V	S	D
1 + Planificación de las tareas del proyecto	16 horas 2 días	Trab.															
2 + Reuniones	6 horas 0,5 días	Trab.															
3 + Especificación de requisitos del sistema	8 horas 1 día	Trab.															
4 + Investigación acerca de la Web semántica	24 horas 3 días	Trab.															
5 + Estudio de los Datos Abiertos en la CAV	16 horas 2 días	Trab.															
6 + Investigación acerca de la web móvil	16 horas 2 días	Trab.															
7 + Estudio acerca de mashups y fuentes de datos	16 horas 2 días	Trab.															
8 + Selección de fuentes de datos y formatos de publicación	4 horas 0,5 días	Trab.															
9 + Diseño de la arquitectura	16 horas 2 días	Trab.															
10 + Selección de la plataforma de desarrollo	16 horas 2 días	Trab.															
11 + Selección de los puestos de trabajo	8 horas 1 día	Trab.															
12 + Instalación de la fuente de datos de EH	4 horas 0,5 días	Trab.															
13 + Investigación de la fuente de datos de EH	16 horas 2 días	Trab.															
14 + Definición del módulo de EH	8 horas 1 día	Trab.															
15 + Implementación del módulo de EH	32 horas 4 días	Trab.															
16 + Investigación de la API de Google Maps	8 horas 1 día	Trab.															
17 + Investigación del formato de las direcciones	4 horas 0,5 días	Trab.															
18 + Definición del módulo de geolocalización	8 horas 1 día	Trab.															
19 + Implementación del módulo de geolocalización	20 horas 2,5 días	Trab.															
20 + Investigación de la API de Google Calendar	4 horas 0,5 días	Trab.															
21 + Definición del módulo de Gcalendar	2 horas 0,25 días	Trab.															
22 + Implementación del módulo de Gcalendar	2 horas 0,25 días	Trab.															
23 + Definición del módulo de eventos finalizados	1 hora 0,13 días	Trab.															
24 + Implementación del módulo de eventos finalizados	1 hora 0,13 días	Trab.															
25 + Investigación de las APIs de los servicios	8 horas 1 día	Trab.						8h									
26 + Investigación de la publicación semántica	8 horas 1 día	Trab.									8h						
27 + Definición del módulo de publicación	24 horas 3 días	Trab.										8h					
28 + Implementación del módulo de publicación	80 horas 10 días	Trab.											8h				
29 + Implementación del módulo de administración	8 horas 1 día	Trab.						8h									
30 + Diseño de las URLs del módulo de API	6 horas 0,75 días	Trab.															
31 + Diseño del serializador semántico	4 horas 0,5 días	Trab.															
32 + Implementación del módulo de API	28 horas 3,5 días	Trab.		8h													
33 + Testeo de la aplicación	12 horas 1 día	Trab.															
34 + Depuración de errores encontrados	16 horas 2 días	Trab.															
35 + Documentación	72 horas 9 días	Trab.															
36 + Selección del servicio de hosting	2 horas 0,25 días	Trab.															
37 + Adaptación del proyecto al hosting	4 horas 0,5 días	Trab.															
38 + Subida al hosting	2 horas 0,25 días	Trab.															

Figura 8.7: Diagrama de cargas reales por tarea (04/07/2011-17/07/2011)

Nombre de tarea				Trabajo		Duración		Detalles																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																		
								18 jul '11							25 jul '11																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																											
								L							M																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																											
								M							X																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																											
								J							V																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																											
								S							D																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																											
								D							L																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																											
								M							X																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																											
								J							V																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																											
								S							D																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																											
1	+	Planificación de las tareas del proyecto		16 horas 2 días	Trab.																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																					

Figura 8.8: Diagrama de cargas reales por tarea (18/07/2011-31/07/2011)

Nombre de tarea			Trabajo	Duración	Detalles															
					01 ago '11				08 ago '11				15 ago '11							
					L	M	X	J	V	S	D	L	M	X	J	V	S	D	L	M
1	+ Planificación de las tareas del proyecto	16 horas 2 días	Trab.																	
2	+ Reuniones	6 horas 0,5 días	Trab.																	
3	+ Especificación de requisitos del sistema	8 horas 1 día	Trab.																	
4	+ Investigación acerca de la Web Semántica	24 horas 3 días	Trab.																	
5	+ Estudio de los Datos Abiertos en la CAV	16 horas 2 días	Trab.																	
6	+ Investigación acerca de la web móvil	16 horas 2 días	Trab.																	
7	+ Estudio acerca de mashups y fuentes de datos	16 horas 2 días	Trab.																	
8	+ Selección de fuentes de datos y formatos de publicación	4 horas 0,5 días	Trab.																	
9	+ Diseño de la arquitectura	16 horas 2 días	Trab.																	
10	+ Diseño de la interacción con el usuario	16 horas 2 días	Trab.																	
11	+ Selección de la plataforma de desarrollo	8 horas 1 día	Trab.																	
12	+ Instalación de los puestos de trabajo	4 horas 0,5 días	Trab.																	
13	+ Investigación de la fuente de datos de EH	16 horas 2 días	Trab.																	
14	+ Definición del módulo de EH	8 horas 1 día	Trab.																	
15	+ Implementación del módulo de EH	32 horas 4 días	Trab.																	
16	+ Investigación de la API de Google Maps	8 horas 1 día	Trab.																	
17	+ Investigación del formato de las direcciones	4 horas 0,5 días	Trab.																	
18	+ Definición del módulo de geolocalización	8 horas 1 día	Trab.																	
19	+ Implementación del módulo de geolocalización	20 horas 2,5 días	Trab.																	
20	+ Investigación de la API de Google Calendar	4 horas 0,5 días	Trab.																	
21	+ Definición del módulo de Galendar	2 horas 0,25 días	Trab.																	
22	+ Implementación del módulo de Galendar	2 horas 0,25 días	Trab.																	
23	+ Definición del módulo de eventos finalizados	1 hora 0,13 días	Trab.																	
24	+ Implementación del módulo de eventos finalizados	1 hora 0,13 días	Trab.																	
25	+ Investigación de las APIs de los servicios	8 horas 1 día	Trab.																	
26	+ Investigación de la publicación semántica	8 horas 1 día	Trab.																	
27	+ Definición del módulo de publicación	24 horas 3 días	Trab.																	
28	+ Implementación del módulo de publicación	80 horas 10 días	Trab.																	
29	+ Implementación del módulo de administración	8 horas 1 día	Trab.																	
30	+ Diseño de las URIs del módulo de API	6 horas 0,75 días	Trab.																	
31	+ Diseño del serializador semántico	4 horas 0,5 días	Trab.																	
32	+ Implementación del módulo de API	28 horas 3,5 días	Trab.																	
33	+ Testeo de la aplicación	12 horas 1 día	Trab.																	
34	+ Depuración de errores encontrados	16 horas 2 días	Trab.		8h	8h														
35	+ Documentación	72 horas 9 días	Trab.				8h													
36	+ Selección del servicio de hosting	2 horas 0,25 días	Trab.				2h						8h	8h	8h	8h				
37	+ Adaptación del proyecto al hosting	4 horas 0,5 días	Trab.				4h													
38	+ Subida al hosting	2 horas 0,25 días	Trab.				2h													

Figura 8.9: Diagrama de cargas reales por tarea (01/08/2011-16/08/2011)

8.5 CARGAS REALES DE TRABAJO POR RECURSO

	Nombre del recurso	Trabajo	Detalles	04 abr '11							11 abr '11						
				L	M	X	J	V	S	D	L	M	X	J	V	S	D
1	<input type="checkbox"/> Jefe del Proyecto - Diego	6 horas	Trabajo					2h									
	Reuniones	2 horas	Trabajo					2h									
	Testeo de la aplicación	4 horas	Trabajo														
2	<input type="checkbox"/> Director del Proyecto - Jon	40,8 horas	Trabajo	4h	4h	4h	4h	2h									
	Planificación de las tareas del proyecto	16 horas	Trabajo	4h	4h	4h	4h										
	Selección de fuentes de datos y formatos de publicación	1 hora	Trabajo														
	Selección de la plataforma de desarrollo	0,8 horas	Trabajo														
	Selección del servicio de hosting	0,2 horas	Trabajo														
	Reuniones	2 horas	Trabajo					2h									
	Diseño de la arquitectura	3,2 horas	Trabajo														
	Diseño de la interacción con el usuario	3,2 horas	Trabajo														
	Testeo de la aplicación	5,6 horas	Trabajo														
	Depuración de errores encontrados	1,6 horas	Trabajo														
	Documentación	7,2 horas	Trabajo														
3	<input type="checkbox"/> Analista - Jon	39 horas	Trabajo					2h			4h	4h					
	Reuniones	2 horas	Trabajo					2h									
	Especificación de requisitos del sistema	8 horas	Trabajo								4h	4h					
	Diseño de la arquitectura	4,8 horas	Trabajo														
	Diseño de la interacción con el usuario	4,8 horas	Trabajo														
	Definición del módulo de geolocalización	0,8 horas	Trabajo														
	Definición del módulo de publicación	2,4 horas	Trabajo														
	Implementación del módulo de administración	0,4 horas	Trabajo														
	Testeo de la aplicación	2,4 horas	Trabajo														
	Depuración de errores encontrados	1,6 horas	Trabajo														
	Documentación	10,8 horas	Trabajo														
	Definición del módulo de E/I	0,8 horas	Trabajo														
	Definición del módulo de Scheduler	0,2 horas	Trabajo														
4	<input type="checkbox"/> Técnico de Sistemas - Jon	5 horas	Trabajo														
	Instalación de los puestos de trabajo	4 horas	Trabajo														
	Selección del servicio de hosting	1 hora	Trabajo														

Figura 8.10: Diagrama de cargas reales para Jefe de Proyecto, Director del Proyecto, Analista y Técnico (04/04/2011-17/04/2011)

Nombre del recurso		Trabajo	Detalles													
			04 abr '11							11 abr '11						
			L	M	X	J	V	S	D	L	M	X	J	V	S	D
5	Programador - Jon	439,2 horas	Trabajo									4h	4h	4h		
	Investigación acerca de la Web Semántica	24 horas	Trabajo									4h	4h	4h		
	Estudio de los Datos Abiertos en la CAV	16 horas	Trabajo													
	Investigación acerca de la web móvil	16 horas	Trabajo													
	Estudio acerca de mashups y fuentes de datos	16 horas	Trabajo													
	Investigación de la fuente de datos de EH	16 horas	Trabajo													
	Definición del módulo de EH	7,2 horas	Trabajo													
	Implementación del módulo de EH	32 horas	Trabajo													
	Investigación de la API de Google Maps	8 horas	Trabajo													
	Investigación del formato de las direcciones	4 horas	Trabajo													
	Implementación del módulo de geolocalización	20 horas	Trabajo													
	Investigación de la API de Google Calendar	4 horas	Trabajo													
	Definición del módulo de Gcalendar	1,8 horas	Trabajo													
	Implementación del módulo de Gcalendar	2 horas	Trabajo													
	Definición del módulo de eventos finalizados	1 hora	Trabajo													
	Implementación del módulo de eventos finalizados	1 hora	Trabajo													
	Investigación de las APIs de los servicios	8 horas	Trabajo													
	Investigación de la publicación semántica	8 horas	Trabajo													
	Implementación del módulo de publicación	80 horas	Trabajo													
	Diseño de las URLs del módulo de API	6 horas	Trabajo													
	Diseño del serializador semántico	4 horas	Trabajo													
	Implementación del módulo de API	28 horas	Trabajo													
	Adaptación del proyecto al hosting	4 horas	Trabajo													
	Subida al hosting	2 horas	Trabajo													
	Selección de fuentes de datos y formatos de publicación	3 horas	Trabajo													
	Diseño de la arquitectura	8 horas	Trabajo													
	Diseño de la interacción con el usuario	8 horas	Trabajo													
	Selección de la plataforma de desarrollo	7,2 horas	Trabajo													
	Definición del módulo de geolocalización	7,2 horas	Trabajo													
	Definición del módulo de publicación	21,6 horas	Trabajo													
	Implementación del módulo de administración	7,6 horas	Trabajo													
	Depuración de errores encontrados	12,8 horas	Trabajo													
	Documentación	54 horas	Trabajo													
	Selección del servicio de hosting	0,8 horas	Trabajo													

Figura 8.11: Diagrama de cargas reales para Programador (04/04/2011-17/04/2011)

9. PLANIFICACIÓN

	Nombre del recurso	Trabajo	Detalles	18 abr '11							25 abr '11						
				L	M	X	J	V	S	D	L	M	X	J	V	S	D
1	▢ Jefe del Proyecto - Diego	6 horas	Trabajo														
	Reuniones	2 horas	Trabajo														
	Testeo de la aplicación	4 horas	Trabajo														
2	▢ Director del Proyecto - Jon	40,8 horas	Trabajo														
	Planificación de las tareas del proyecto	16 horas	Trabajo														
	Selección de fuentes de datos y formatos de publicación	1 hora	Trabajo														
	Selección de la plataforma de desarrollo	0,8 horas	Trabajo														
	Selección del servicio de hosting	0,2 horas	Trabajo														
	Reuniones	2 horas	Trabajo														
	Diseño de la arquitectura	3,2 horas	Trabajo														
	Diseño de la interacción con el usuario	3,2 horas	Trabajo														
	Testeo de la aplicación	5,6 horas	Trabajo														
	Depuración de errores encontrados	1,6 horas	Trabajo														
	Documentación	7,2 horas	Trabajo														
3	▢ Analista - Jon	39 horas	Trabajo														
	Reuniones	2 horas	Trabajo														
	Especificación de requisitos del sistema	8 horas	Trabajo														
	Diseño de la arquitectura	4,8 horas	Trabajo														
	Diseño de la interacción con el usuario	4,8 horas	Trabajo														
	Definición del módulo de geolocalización	0,8 horas	Trabajo														
	Definición del módulo de publicación	2,4 horas	Trabajo														
	Implementación del módulo de administración	0,4 horas	Trabajo														
	Testeo de la aplicación	2,4 horas	Trabajo														
	Depuración de errores encontrados	1,6 horas	Trabajo														
	Documentación	10,8 horas	Trabajo														
	Definición del módulo de E+I	0,8 horas	Trabajo														
	Definición del módulo de Gcalendar	0,2 horas	Trabajo														
4	▢ Técnico de Sistemas - Jon	5 horas	Trabajo														
	Instalación de los puestos de trabajo	4 horas	Trabajo														
	Selección del servicio de hosting	1 hora	Trabajo														

Figura 8.12: Diagrama de cargas reales para Jefe de Proyecto, Director del Proyecto, Analista y Técnico (18/04/2011-01/05/2011)

Nombre del recurso	Trabajo	Detalles	18 abr '11							25 abr '11						
			L	M	X	J	V	S	D	L	M	X	J	V	S	D
5	Programador - Ion	439,2 horas	Trabajo	4h	4h	4h					4h	4h	4h	4h		
	Investigación acerca de la Web Semántica	24 horas	Trabajo	4h	4h											
	Estudio de los Datos Abiertos en la CAV	16 horas	Trabajo								4h		4h			
	Investigación acerca de la web móvil	16 horas	Trabajo									4h		4h		
	Estudio acerca de mashups y fuentes de datos	16 horas	Trabajo													
	Investigación de la fuente de datos de EH	16 horas	Trabajo													
	Definición del módulo de EH	7,2 horas	Trabajo													
	Implementación del módulo de EH	32 horas	Trabajo													
	Investigación de la API de Google Maps	8 horas	Trabajo													
	Investigación del formato de las direcciones	4 horas	Trabajo													
	Implementación del módulo de geolocalización	20 horas	Trabajo													
	Investigación de la API de Google Calendar	4 horas	Trabajo													
	Definición del módulo de Gcalendar	1,8 horas	Trabajo													
	Implementación del módulo de Gcalendar	2 horas	Trabajo													
	Definición del módulo de eventos finalizados	1 hora	Trabajo													
	Implementación del módulo de eventos finalizados	1 hora	Trabajo													
	Investigación de las APIs de los servicios	8 horas	Trabajo													
	Investigación de la publicación semántica	8 horas	Trabajo													
	Implementación del módulo de publicación	80 horas	Trabajo													
	Diseño de las URLs del módulo de API	6 horas	Trabajo													
	Diseño del serializador semántico	4 horas	Trabajo													
	Implementación del módulo de API	28 horas	Trabajo													
	Adaptación del proyecto al hosting	4 horas	Trabajo													
	Subida al hosting	2 horas	Trabajo													
	Selección de fuentes de datos y formatos de publicación	3 horas	Trabajo													
	Diseño de la arquitectura	8 horas	Trabajo													
	Diseño de la interacción con el usuario	8 horas	Trabajo													
	Selección de la plataforma de desarrollo	7,2 horas	Trabajo													
	Definición del módulo de geolocalización	7,2 horas	Trabajo													
	Definición del módulo de publicación	21,6 horas	Trabajo													
	Implementación del módulo de administración	7,6 horas	Trabajo													
	Depuración de errores encontrados	12,8 horas	Trabajo													
	Documentación	54 horas	Trabajo													
	Selección del servicio de hosting	0,8 horas	Trabajo													

Figura 8.13: Diagrama de cargas reales para Programador (18/04/2011-01/05/2011)

9. PLANIFICACIÓN

	Nombre del recurso	Trabajo	Detalles													
			02 may '11	03 may '11	04 may '11	05 may '11	06 may '11	07 may '11	08 may '11	09 may '11	10 may '11	11 may '11	12 may '11	13 may '11	14 may '11	15 may '11
1	<input type="checkbox"/> Jefe del Proyecto - Diego	6 horas	Trabajo													
	Reuniones	2 horas	Trabajo													
	Testeo de la aplicación	4 horas	Trabajo													
2	<input type="checkbox"/> Director del Proyecto - Jon	40,8 horas	Trabajo										1h	0,8h		
	Planificación de los tareas del proyecto	16 horas	Trabajo													
	Selección de fuentes de datos y formatos de publicación	1 hora	Trabajo										1h			
	Selección de la plataforma de desarrollo	0,8 horas	Trabajo													
	Selección del servicio de hosting	0,2 horas	Trabajo													
	Reuniones	2 horas	Trabajo													
	Diseño de la arquitectura	3,2 horas	Trabajo											0,8h		
	Diseño de la interacción con el usuario	3,2 horas	Trabajo													
	Testeo de la aplicación	5,6 horas	Trabajo													
	Depuración de errores encontrados	1,6 horas	Trabajo													
	Documentación	7,2 horas	Trabajo													
3	<input type="checkbox"/> Analista - Jon	39 horas	Trabajo											1,2h		
	Reuniones	2 horas	Trabajo													
	Especificación de requisitos del sistema	8 horas	Trabajo													
	Diseño de la arquitectura	4,8 horas	Trabajo											1,2h		
	Diseño de la interacción con el usuario	4,8 horas	Trabajo													
	Definición del módulo de geolocalización	0,8 horas	Trabajo													
	Definición del módulo de publicación	2,4 horas	Trabajo													
	Implementación del módulo de administración	0,4 horas	Trabajo													
	Testeo de la aplicación	2,4 horas	Trabajo													
	Depuración de errores encontrados	1,6 horas	Trabajo													
	Documentación	10,8 horas	Trabajo													
	Definición del módulo de EH	0,8 horas	Trabajo													
	Definición del módulo de Calendar	0,2 horas	Trabajo													
4	<input type="checkbox"/> Técnico de Sistemas - Jon	5 horas	Trabajo													
	Instalación de los puestos de trabajo	4 horas	Trabajo													
	Selección del servicio de hosting	1 hora	Trabajo													

Figura 8.14: Diagrama de cargas reales para Jefe de Proyecto, Director del Proyecto, Analista y Técnico (02/05/2011-15/05/2011)

	Nombre del recurso	Trabajo	Detalles	02 may '11							09 may '11						
				L	M	X	J	V	S	D	L	M	X	J	V	S	D
5	Programador - Jon	439,2 horas	Trabajo	4h	4h	4h	4h	4h	4h		4h	4h	4h	3h	2h		
	Investigación acerca de la Web Semántica	24 horas	Trabajo														
	Estudio de los Datos Abiertos en la CAV	16 horas	Trabajo														
	Investigación acerca de la web móvil	16 horas	Trabajo	4h	4h	4h	4h										
	Estudio acerca de mashups y fuentes de datos	16 horas	Trabajo					4h			4h	4h	4h				
	Investigación de la fuente de datos de EH	16 horas	Trabajo														
	Definición del módulo de EH	7,2 horas	Trabajo														
	Implementación del módulo de EH	32 horas	Trabajo														
	Investigación de la API de Google Maps	8 horas	Trabajo														
	Investigación del formato de los direcciones	4 horas	Trabajo														
	Implementación del módulo de geolocalización	20 horas	Trabajo														
	Investigación de la API de Google Calendar	4 horas	Trabajo														
	Definición del módulo de Gcalendar	1,8 horas	Trabajo														
	Implementación del módulo de Gcalendar	2 horas	Trabajo														
	Definición del módulo de eventos finalizados	1 hora	Trabajo														
	Implementación del módulo de eventos finalizados	1 hora	Trabajo														
	Investigación de las APIs de los servicios	8 horas	Trabajo														
	Investigación de la publicación semántica	8 horas	Trabajo														
	Implementación del módulo de publicación	80 horas	Trabajo														
	Diseño de las URLs del módulo de API	6 horas	Trabajo														
	Diseño del serializador semántico	4 horas	Trabajo														
	Implementación del módulo de API	28 horas	Trabajo														
	Adaptación del proyecto al hosting	4 horas	Trabajo														
	Subida al hosting	2 horas	Trabajo														
	Selección de fuentes de datos y formatos de publicación	3 horas	Trabajo											3h			
	Diseño de la arquitectura	8 horas	Trabajo												2h		
	Diseño de la interacción con el usuario	8 horas	Trabajo														
	Selección de la plataforma de desarrollo	7,2 horas	Trabajo														
	Definición del módulo de geolocalización	7,2 horas	Trabajo														
	Definición del módulo de publicación	21,6 horas	Trabajo														
	Implementación del módulo de administración	7,6 horas	Trabajo														
	Depuración de errores encontrados	12,8 horas	Trabajo														
	Documentación	54 horas	Trabajo														
	Selección del servicio de hosting	0,8 horas	Trabajo														

Figura 8.15: Diagrama de cargas reales para Programador (02/05/2011-15/05/2011)

9. PLANIFICACIÓN

	Nombre del recurso	Trabajo	Detalles	06 jun '11							13 jun '11						
				L	M	X	J	V	S	D	L	M	X	J	V	S	D
1	<input type="checkbox"/> Jefe del Proyecto - Diego	6 horas	Trabajo														
	Reuniones	2 horas	Trabajo														
	Testeo de la aplicación	4 horas	Trabajo														
2	<input type="checkbox"/> Director del Proyecto - Jon	40,8 horas	Trabajo	1,6h	1,6h	1,6h	1,2h	0,4h									
	Planificación de las tareas del proyecto	16 horas	Trabajo														
	Selección de fuentes de datos y formatos de publicación	1 hora	Trabajo														
	Selección de la plataforma de desarrollo	0,8 horas	Trabajo				0,4h	0,4h									
	Selección del servicio de hosting	0,2 horas	Trabajo														
	Reuniones	2 horas	Trabajo														
	Diseño de la arquitectura	3,2 horas	Trabajo	1,6h	0,8h												
	Diseño de la interacción con el usuario	3,2 horas	Trabajo		0,8h	1,6h	0,8h										
	Testeo de la aplicación	5,6 horas	Trabajo														
	Depuración de errores encontrados	1,6 horas	Trabajo														
	Documentación	7,2 horas	Trabajo														
3	<input type="checkbox"/> Analista - Jon	39 horas	Trabajo	2,4h	2,4h	2,4h	1,2h					0,4h	0,4h				
	Reuniones	2 horas	Trabajo														
	Especificación de requisitos del sistema	8 horas	Trabajo														
	Diseño de la arquitectura	4,8 horas	Trabajo	2,4h	1,2h												
	Diseño de la interacción con el usuario	4,8 horas	Trabajo		1,2h	2,4h	1,2h										
	Definición del módulo de geolocalización	0,8 horas	Trabajo									0,4h	0,4h				
	Definición del módulo de publicación	2,4 horas	Trabajo														
	Implementación del módulo de administración	0,4 horas	Trabajo														
	Testeo de la aplicación	2,4 horas	Trabajo														
	Depuración de errores encontrados	1,6 horas	Trabajo														
	Documentación	10,8 horas	Trabajo														
	Definición del módulo de EH	0,8 horas	Trabajo														
	Definición del módulo de Calendar	0,2 horas	Trabajo														
4	<input type="checkbox"/> Técnico de Sistemas - Jon	5 horas	Trabajo					4h									
	Instalación de los puestos de trabajo	4 horas	Trabajo					4h									
	Selección del servicio de hosting	1 hora	Trabajo														

Figura 8.16: Diagrama de cargas reales para Jefe de Proyecto, Director del Proyecto, Analista y Técnico (06/06/2011-19/06/2011)

	Nombre del recurso	Trabajo	Detalles													
			06 jun '11							13 jun '11						
			L	M	X	J	V	S	D	L	M	X	J	V	S	D
5	Programador - Jon	439,2 horas	Trabajo	4h	4h	4h	5,6h	3,6h		8h	7,6h	7,6h	8h	8h		
	Investigación acerca de la Web Semántica	24 horas	Trabajo													
	Estudio de los Datos Abiertos en la CAV	16 horas	Trabajo													
	Investigación acerca de la web móvil	16 horas	Trabajo													
	Estudio acerca de moshups y fuentes de datos	16 horas	Trabajo													
	Investigación de la fuente de datos de FH	16 horas	Trabajo													
	Definición del módulo de FH	7,2 horas	Trabajo													
	Implementación del módulo de FH	32 horas	Trabajo													
	Investigación de la API de Google Maps	8 horas	Trabajo							8h						
	Investigación del formato de las direcciones	4 horas	Trabajo								4h					
	Implementación del módulo de geolocalización	20 horas	Trabajo									4h	8h	8h		
	Investigación de la API de Google Calendar	4 horas	Trabajo													
	Definición del módulo de Scalendar	1,8 horas	Trabajo													
	Implementación del módulo de Scalendar	2 horas	Trabajo													
	Definición del módulo de eventos finalizados	1 hora	Trabajo													
	Implementación del módulo de eventos finalizados	1 hora	Trabajo													
	Investigación de las APIs de los servicios	8 horas	Trabajo													
	Investigación de la publicación semántica	8 horas	Trabajo													
	Implementación del módulo de publicación	80 horas	Trabajo													
	Diseño de las URLs del módulo de API	6 horas	Trabajo													
	Diseño del serializador semántico	4 horas	Trabajo													
	Implementación del módulo de API	28 horas	Trabajo													
	Adaptación del proyecto al hosting	4 horas	Trabajo													
	Subida al hosting	2 horas	Trabajo													
	Selección de fuentes de datos y formatos de publicación	3 horas	Trabajo													
	Diseño de la arquitectura	8 horas	Trabajo	4h	2h		2h									
	Diseño de la interacción con el usuario	8 horas	Trabajo		2h	4h	2h									
	Selección de la plataforma de desarrollo	7,2 horas	Trabajo				3,6h	3,6h								
	Definición del módulo de geolocalización	7,2 horas	Trabajo								3,6h	3,6h				
	Definición del módulo de publicación	21,6 horas	Trabajo													
	Implementación del módulo de administración	7,6 horas	Trabajo													
	Depuración de errores encontrados	12,8 horas	Trabajo													
	Documentación	54 horas	Trabajo													
	Selección del servicio de hosting	0,8 horas	Trabajo													

Figura 8.17: Diagrama de cargas reales para Programador (06/06/2011-19/06/2011)

9. PLANIFICACIÓN

Nombre del recurso			Trabajo		Detalles													
					20 jun '11							27 jun '11						
					L	M	X	J	V	S	D	L	M	X	J	V	S	D
1	[-] Jefe del Proyecto - Diego	6 horas	Trabajo															
	Reuniones	2 horas	Trabajo															
	Testeo de la aplicación	4 horas	Trabajo															
2	[-] Director del Proyecto - Jon	40,8 horas	Trabajo															
	Planificación de las tareas del proyecto	16 horas	Trabajo															
	Selección de fuentes de datos y formatos de publicación	1 hora	Trabajo															
	Selección de la plataforma de desarrollo	0,8 horas	Trabajo															
	Selección del servicio de hosting	0,2 horas	Trabajo															
	Reuniones	2 horas	Trabajo															
	Diseño de la arquitectura	3,2 horas	Trabajo															
	Diseño de la interacción con el usuario	3,2 horas	Trabajo															
	Testeo de la aplicación	5,6 horas	Trabajo															
	Depuración de errores encontrados	1,6 horas	Trabajo															
	Documentación	7,2 horas	Trabajo															
3	[-] Analista - Jon	39 horas	Trabajo		0,2h				0,8h									
	Reuniones	2 horas	Trabajo															
	Especificación de requisitos del sistema	8 horas	Trabajo															
	Diseño de la arquitectura	4,8 horas	Trabajo															
	Diseño de la interacción con el usuario	4,8 horas	Trabajo															
	Definición del módulo de geolocalización	0,8 horas	Trabajo															
	Definición del módulo de publicación	2,4 horas	Trabajo															
	Implementación del módulo de administración	0,4 horas	Trabajo															
	Testeo de la aplicación	2,4 horas	Trabajo															
	Depuración de errores encontrados	1,6 horas	Trabajo															
	Documentación	10,8 horas	Trabajo															
	Definición del módulo de EH	0,8 horas	Trabajo					0,8h										
	Definición del módulo de Goleendar	0,2 horas	Trabajo		0,2h													
4	[-] Técnico de Sistemas - Jon	5 horas	Trabajo															
	Instalación de los puestos de trabajo	4 horas	Trabajo															
	Selección del servicio de hosting	1 hora	Trabajo															

Figura 8.18: Diagrama de cargas reales para Jefe de Proyecto, Director del Proyecto, Analista y Técnico (20/06/2011-03/07/2011)

	Nombre del recurso	Trabajo	Detalles	20 jun '11							27 jun '11						
				L	M	X	J	V	S	D	L	M	X	J	V	S	D
5	Programador - Ion	439,2 horas	Trabajo	7,8h	8h	8h	7,2h	8h			8h	8h	8h	1h	8h		
	Investigación acerca de la Web Semántica	24 horas	Trabajo														
	Estudio de los Datos Abiertos en la CAV	16 horas	Trabajo														
	Investigación acerca de la web móvil	16 horas	Trabajo														
	Estudio acerca de mashups y fuentes de datos	16 horas	Trabajo		8h	8h											
	Investigación de la fuente de datos de EH	16 horas	Trabajo				7,2h										
	Definición del módulo de EH	7,2 horas	Trabajo					8h			8h	8h	8h				
	Implementación del módulo de EH	32 horas	Trabajo														
	Investigación de la API de Google Maps	8 horas	Trabajo														
	Investigación del formato de las direcciones	4 horas	Trabajo														
	Implementación del módulo de geolocalización	20 horas	Trabajo														
	Investigación de la API de Google Calendar	4 horas	Trabajo	4h													
	Definición del módulo de Gcalendar	1,8 horas	Trabajo	1,8h													
	Implementación del módulo de Gcalendar	2 horas	Trabajo	2h										1h			
	Definición del módulo de eventos finalizados	1 hora	Trabajo											1h			
	Implementación del módulo de eventos finalizados	1 hora	Trabajo														
	Investigación de las APIs de los servicios	8 horas	Trabajo														
	Investigación de la publicación semántica	8 horas	Trabajo														
	Implementación del módulo de publicación	80 horas	Trabajo														
	Diseño de las URLs del módulo de API	6 horas	Trabajo											6h			
	Diseño de la serializador semántico	4 horas	Trabajo												4h		
	Implementación del módulo de API	28 horas	Trabajo													4h	
	Adaptación del proyecto al hosting	4 horas	Trabajo														
	Subida al hosting	2 horas	Trabajo														
	Selección de fuentes de datos y formatos de publicación	3 horas	Trabajo														
	Diseño de la arquitectura	8 horas	Trabajo														
	Diseño de la interacción con el usuario	8 horas	Trabajo														
	Selección de la plataforma de desarrollo	7,2 horas	Trabajo														
	Definición del módulo de geolocalización	7,2 horas	Trabajo														
	Definición del módulo de publicación	21,6 horas	Trabajo														
	Implementación del módulo de administración	7,6 horas	Trabajo														
	Depuración de errores encontrados	12,8 horas	Trabajo														
	Documentación	54 horas	Trabajo														
	Selección del servicio de hosting	0,8 horas	Trabajo														

Figura 8.19: Diagrama de cargas reales para Programador (20/06/2011-03/07/2011)

	Nombre del recurso	Trabajo	Detalles	04 jul '11							11 jul '11						
				L	M	X	J	V	S	D	L	M	X	J	V	S	D
1	▢ Jefe del Proyecto - Diego	6 horas	Trabajo														
	Reuniones	2 horas	Trabajo														
	Testeo de la aplicación	4 horas	Trabajo														
2	▢ Director del Proyecto - Jon	40,8 horas	Trabajo														
	Planificación de las tareas del proyecto	16 horas	Trabajo														
	Selección de fuentes de datos y formatos de publicación	1 hora	Trabajo														
	Selección de la plataforma de desarrollo	0,8 horas	Trabajo														
	Selección del servicio de hosting	0,2 horas	Trabajo														
	Reuniones	2 horas	Trabajo														
	Diseño de la arquitectura	3,2 horas	Trabajo														
	Diseño de la interacción con el usuario	3,2 horas	Trabajo														
	Testeo de la aplicación	5,6 horas	Trabajo														
	Depuración de errores encontrados	1,6 horas	Trabajo														
	Documentación	7,2 horas	Trabajo														
3	▢ Analista - Jon	39 horas	Trabajo				0,4h					0,8h	0,8h	0,8h			
	Reuniones	2 horas	Trabajo														
	Especificación de requisitos del sistema	8 horas	Trabajo														
	Diseño de la arquitectura	4,8 horas	Trabajo														
	Diseño de la interacción con el usuario	4,8 horas	Trabajo														
	Definición del módulo de geolocalización	0,8 horas	Trabajo														
	Definición del módulo de publicación	2,4 horas	Trabajo									0,8h	0,8h	0,8h			
	Implementación del módulo de administración	0,4 horas	Trabajo				0,4h										
	Testeo de la aplicación	2,4 horas	Trabajo														
	Depuración de errores encontrados	1,6 horas	Trabajo														
	Documentación	10,8 horas	Trabajo														
	Definición del módulo de E+I	0,8 horas	Trabajo														
	Definición del módulo de Gcalendar	0,2 horas	Trabajo														
4	▢ Técnico de Sistemas - Jon	5 horas	Trabajo														
	Instalación de los puestos de trabajo	4 horas	Trabajo														
	Selección del servicio de hosting	1 hora	Trabajo														

Figura 8.20: Diagrama de cargas reales para Jefe de Proyecto, Director del Proyecto, Analista y Técnico (04/07/2011-17/07/2011)

	Nombre del recurso	Trabajo	Detalles	04 jul '11							11 jul '11						
				L	M	X	J	V	S	D	L	M	X	J	V	S	D
5	Programador - Jon	439,2 horas	Trabajo	8h							8h	7,2h	7,2h	7,2h	8h		
	Investigación acerca de la Web Semántica	24 horas	Trabajo				7,6h	8h									
	Estudio de los Datos Abiertos en la CAV	16 horas	Trabajo														
	Investigación acerca de la web móvil	16 horas	Trabajo														
	Estudio acerca de mashups y fuentes de datos	16 horas	Trabajo														
	Investigación de la fuente de datos de EH	16 horas	Trabajo														
	Definición del módulo de EH	7,2 horas	Trabajo														
	Implementación del módulo de EH	32 horas	Trabajo														
	Investigación de la API de Google Maps	8 horas	Trabajo														
	Investigación del formato de las direcciones	4 horas	Trabajo														
	Implementación del módulo de geolocalización	20 horas	Trabajo														
	Investigación de la API de Google Calendar	4 horas	Trabajo														
	Definición del módulo de Gcalendar	1,8 horas	Trabajo														
	Implementación del módulo de Gcalendar	2 horas	Trabajo														
	Definición del módulo de eventos finalizados	1 hora	Trabajo														
	Implementación del módulo de eventos finalizados	1 hora	Trabajo														
	Investigación de las APIs de los servicios	8 horas	Trabajo					8h			8h						
	Investigación de la publicación semántica	8 horas	Trabajo								8h						
	Implementación del módulo de publicación	80 horas	Trabajo												8h		
	Diseño de las URLs del módulo de API	6 horas	Trabajo														
	Diseño del serializador semántico	4 horas	Trabajo														
	Implementación del módulo de API	28 horas	Trabajo		8h	8h	8h										
	Adaptación del proyecto al hosting	4 horas	Trabajo														
	Subida al hosting	2 horas	Trabajo														
	Selección de fuentes de datos y formatos de publicación	3 horas	Trabajo														
	Diseño de la arquitectura	8 horas	Trabajo														
	Diseño de la interacción con el usuario	8 horas	Trabajo														
	Selección de la plataforma de desarrollo	7,2 horas	Trabajo														
	Definición del módulo de geolocalización	7,2 horas	Trabajo														
	Definición del módulo de publicación	21,6 horas	Trabajo									7,2h	7,2h	7,2h			
	Implementación del módulo de administración	7,6 horas	Trabajo				7,6h										
	Depuración de errores encontrados	12,8 horas	Trabajo														
	Documentación	54 horas	Trabajo														
	Selección del servicio de hosting	0,8 horas	Trabajo														

Figura 8.21: Diagrama de cargas reales para Programador (04/07/2011-17/07/2011)

Nombre del recurso			Trabajo	Detalles													
				18 jul '11							25 jul '11						
				L	M	X	J	V	S	D	L	M	X	J	V	S	D
1	<div><div></div>Jefe del Proyecto - Diego</div>	6 horas	Trabajo												4h		
	Reuniones	2 horas	Trabajo														
	Testeo de la aplicación	4 horas	Trabajo												4h		
2	<div><div></div>Director del Proyecto - Jon</div>	40,8 horas	Trabajo												5,6h		
	Planificación de las tareas del proyecto	16 horas	Trabajo														
	Selección de fuentes de datos y formatos de publicación	1 hora	Trabajo														
	Selección de la plataforma de desarrollo	0,8 horas	Trabajo														
	Selección del servicio de hosting	0,2 horas	Trabajo														
	Reuniones	2 horas	Trabajo														
	Diseño de la arquitectura	3,2 horas	Trabajo														
	Diseño de la interacción con el usuario	3,2 horas	Trabajo														
	Testeo de la aplicación	5,6 horas	Trabajo												5,6h		
	Depuración de errores encontrados	1,6 horas	Trabajo														
	Documentación	7,2 horas	Trabajo														
3	<div><div></div>Analista - Jon</div>	39 horas	Trabajo												2,4h		
	Reuniones	2 horas	Trabajo														
	Especificación de requisitos del sistema	8 horas	Trabajo														
	Diseño de la arquitectura	4,8 horas	Trabajo														
	Diseño de la interacción con el usuario	4,8 horas	Trabajo														
	Definición del módulo de geolocalización	0,8 horas	Trabajo														
	Definición del módulo de publicación	2,4 horas	Trabajo														
	Implementación del módulo de administración	0,4 horas	Trabajo														
	Testeo de la aplicación	2,4 horas	Trabajo												2,4h		
	Depuración de errores encontrados	1,6 horas	Trabajo														
	Documentación	10,8 horas	Trabajo														
	Definición del módulo de E+I	0,8 horas	Trabajo														
	Definición del módulo de Gcalendar	0,2 horas	Trabajo														
4	<div><div></div>Técnico de Sistemas - Jon</div>	5 horas	Trabajo														
	Instalación de los puestos de trabajo	4 horas	Trabajo														
	Selección del servicio de hosting	1 hora	Trabajo														

Figura 8.22: Diagrama de cargas reales para Jefe de Proyecto, Director del Proyecto, Analista y Técnico (18/07/2011-31/07/2011)

	Nombre del recurso	Trabajo	Detalles													
			18 jul '11							25 jul '11						
			L	M	X	J	V	S	D	L	M	X	J	V	S	D
5	Programador - Jon	439,2 horas	Trabajo	8h	8h	8h	8h	8h		8h	8h	8h	8h			
	Investigación acerca de la Web Semántica	24 horas	Trabajo													
	Estudio de los Datos Abiertos en la CAV	16 horas	Trabajo													
	Investigación acerca de la web móvil	16 horas	Trabajo													
	Estudio acerca de mashups y fuentes de datos	16 horas	Trabajo													
	Investigación de la fuente de datos de EH	16 horas	Trabajo													
	Definición del módulo de EH	7,2 horas	Trabajo													
	Implementación del módulo de EH	32 horas	Trabajo													
	Investigación de la API de Google Maps	8 horas	Trabajo													
	Investigación del formato de los direcciones	4 horas	Trabajo													
	Implementación del módulo de geolocalización	20 horas	Trabajo													
	Investigación de la API de Google Calendar	4 horas	Trabajo													
	Definición del módulo de Gcalendar	1,8 horas	Trabajo													
	Implementación del módulo de Gcalendar	2 horas	Trabajo													
	Definición del módulo de eventos finalizados	1 hora	Trabajo													
	Implementación del módulo de eventos finalizados	1 hora	Trabajo													
	Investigación de las APIs de los servicios	8 horas	Trabajo													
	Investigación de la publicación semántica	8 horas	Trabajo													
	Implementación del módulo de publicación	80 horas	Trabajo	8h	8h	8h	8h	8h		8h	8h	8h	8h			
	Diseño de las URLs del módulo de API	6 horas	Trabajo													
	Diseño del serializador semántico	4 horas	Trabajo													
	Implementación del módulo de API	28 horas	Trabajo													
	Adaptación del proyecto al hosting	4 horas	Trabajo													
	Subida al hosting	2 horas	Trabajo													
	Selección de fuentes de datos y formatos de publicación	3 horas	Trabajo													
	Diseño de la arquitectura	8 horas	Trabajo													
	Diseño de la interacción con el usuario	8 horas	Trabajo													
	Selección de la plataforma de desarrollo	7,2 horas	Trabajo													
	Definición del módulo de geolocalización	7,2 horas	Trabajo													
	Definición del módulo de publicación	21,6 horas	Trabajo													
	Implementación del módulo de administración	7,6 horas	Trabajo													
	Depuración de errores encontrados	12,8 horas	Trabajo													
	Documentación	54 horas	Trabajo													
	Selección del servicio de hosting	0,8 horas	Trabajo													

Figura 8.23: Diagrama de cargas reales para Programador (18/07/2011-31/07/2011)

	Nombre del recurso	Trabajo	Detalles	01 ago '11							08 ago '11							15 ago '11	
				L	M	X	J	V	S	D	L	M	X	J	V	S	D	L	M
1	▢ Jefe del Proyecto - Diego	6 horas	Trabajo																
	Reuniones	2 horas	Trabajo																
	Testeo de la aplicación	4 horas	Trabajo																
2	▢ Director del Proyecto - Jon	40,8 horas	Trabajo	0,8h	0,8h	0,2h	0,8h	0,8h			0,8h	0,8h	0,8h	0,8h	0,8h			0,8h	0,8h
	Planificación de las tareas del proyecto	16 horas	Trabajo																
	Selección de fuentes de datos y formatos de publicación	1 hora	Trabajo																
	Selección de la plataforma de desarrollo	0,8 horas	Trabajo																
	Selección del servicio de hosting	0,2 horas	Trabajo			0,2h													
	Reuniones	2 horas	Trabajo																
	Diseño de la arquitectura	3,2 horas	Trabajo																
	Diseño de la interacción con el usuario	3,2 horas	Trabajo																
	Testeo de la aplicación	5,6 horas	Trabajo																
	Depuración de errores encontrados	1,6 horas	Trabajo	0,8h	0,8h						0,8h	0,8h	0,8h	0,8h	0,8h			0,8h	0,8h
	Documentación	7,2 horas	Trabajo								0,8h	0,8h	0,8h	0,8h	0,8h			0,8h	0,8h
3	▢ Analista - Jon	39 horas	Trabajo	0,8h	0,8h		1,2h	1,2h			1,2h	1,2h	1,2h	1,2h	1,2h			1,2h	1,2h
	Reuniones	2 horas	Trabajo																
	Especificación de requisitos del sistema	8 horas	Trabajo																
	Diseño de la arquitectura	4,8 horas	Trabajo																
	Diseño de la interacción con el usuario	4,8 horas	Trabajo																
	Definición del módulo de geolocalización	0,8 horas	Trabajo																
	Definición del módulo de publicación	2,4 horas	Trabajo																
	Implementación del módulo de administración	0,4 horas	Trabajo																
	Testeo de la aplicación	2,4 horas	Trabajo																
	Depuración de errores encontrados	1,6 horas	Trabajo	0,8h	0,8h						1,2h	1,2h	1,2h	1,2h	1,2h			1,2h	1,2h
	Documentación	10,8 horas	Trabajo																
	Definición del módulo de EH	0,8 horas	Trabajo																
	Definición del módulo de Gcalendar	0,2 horas	Trabajo																
4	▢ Técnico de Sistemas - Jon	5 horas	Trabajo			1h													
	Instalación de los puestos de trabajo	4 horas	Trabajo																
	Selección del servicio de hosting	1 hora	Trabajo			1h													

Figura 8.24: Diagrama de cargas reales para Jefe de Proyecto, Director del Proyecto, Analista y Técnico (01/08/2011-16/08/2011)

	Nombre del recurso	Trabajo	Detalles													
			01 ago '11							08 ago '11						
			L	M	X	J	V	S	D	L	M	X	J	V	S	D
5	Programador - Jon	439,2 horas	Trabajo	6,4h	6,4h	6,8h	6h	6h		6h	6h	6h	6h	6h		
	Investigación acerca de la Web Semántica	24 horas	Trabajo													
	Estudio de los Datos Abiertos en la CAV	16 horas	Trabajo													
	Investigación acerca de la web móvil	16 horas	Trabajo													
	Estudio acerca de mashups y fuentes de datos	16 horas	Trabajo													
	Investigación de la fuente de datos de EH	16 horas	Trabajo													
	Definición del módulo de EH	7,2 horas	Trabajo													
	Implementación del módulo de EH	32 horas	Trabajo													
	Investigación de la API de Google Maps	8 horas	Trabajo													
	Investigación del formato de las direcciones	4 horas	Trabajo													
	Implementación del módulo de geolocalización	20 horas	Trabajo													
	Investigación de la API de Google Calendar	4 horas	Trabajo													
	Definición del módulo de Gcalendar	1,8 horas	Trabajo													
	Implementación del módulo de Gcalendar	2 horas	Trabajo													
	Definición del módulo de eventos finalizados	1 hora	Trabajo													
	Implementación del módulo de eventos finalizados	1 hora	Trabajo													
	Investigación de las APIs de los servicios	8 horas	Trabajo													
	Investigación de la publicación semántica	8 horas	Trabajo													
	Implementación del módulo de publicación	80 horas	Trabajo													
	Diseño de las URLs del módulo de API	6 horas	Trabajo													
	Diseño del serializador semántico	4 horas	Trabajo													
	Implementación del módulo de API	28 horas	Trabajo													
	Adaptación del proyecto al hosting	4 horas	Trabajo			4h										
	Subida al hosting	2 horas	Trabajo			2h										
	Selección de fuentes de datos y formatos de publicación	3 horas	Trabajo													
	Diseño de la arquitectura	8 horas	Trabajo													
	Diseño de la interacción con el usuario	8 horas	Trabajo													
	Selección de la plataforma de desarrollo	7,2 horas	Trabajo													
	Definición del módulo de geolocalización	7,2 horas	Trabajo													
	Definición del módulo de publicación	21,6 horas	Trabajo													
	Implementación del módulo de administración	7,6 horas	Trabajo													
	Depuración de errores encontrados	12,8 horas	Trabajo	6,4h	6,4h					6h	6h	6h	6h	6h		
	Documentación	54 horas	Trabajo				6h	6h			6h	6h	6h	6h		
	Selección del servicio de hosting	0,8 horas	Trabajo			0,8h									6h	6h

Figura 8.25: Diagrama de cargas reales para Programador (01/08/2011-16/08/2011)

8.6 DIAGRAMA DE GANTT

El Diagrama de Gantt es una herramienta gráfica cuyo objetivo es mostrar el tiempo de dedicación para diferentes tareas a lo largo del tiempo total de un proyecto.

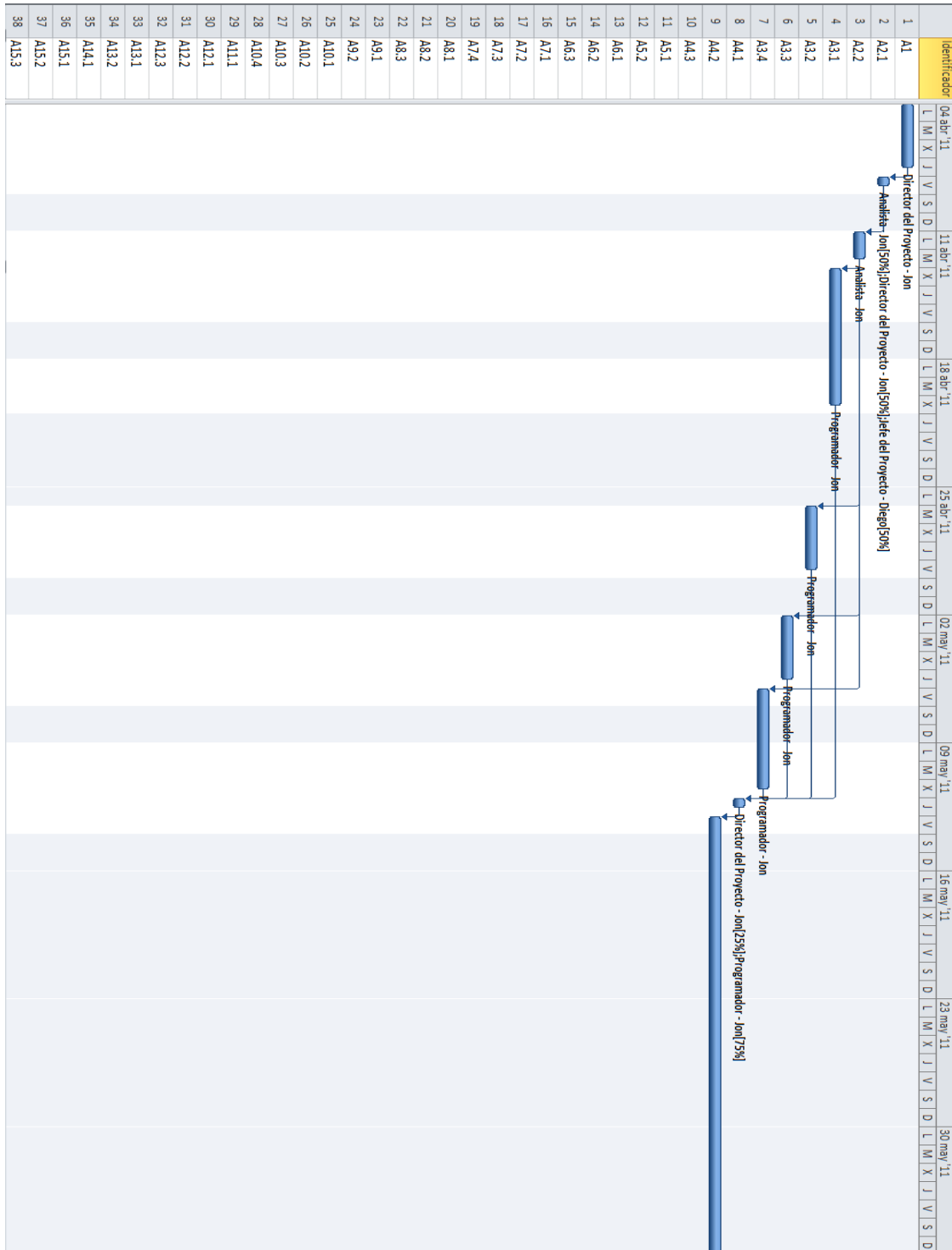


Figura 8.26: Diagrama de Gantt (04/04/2011-05/06/2011)

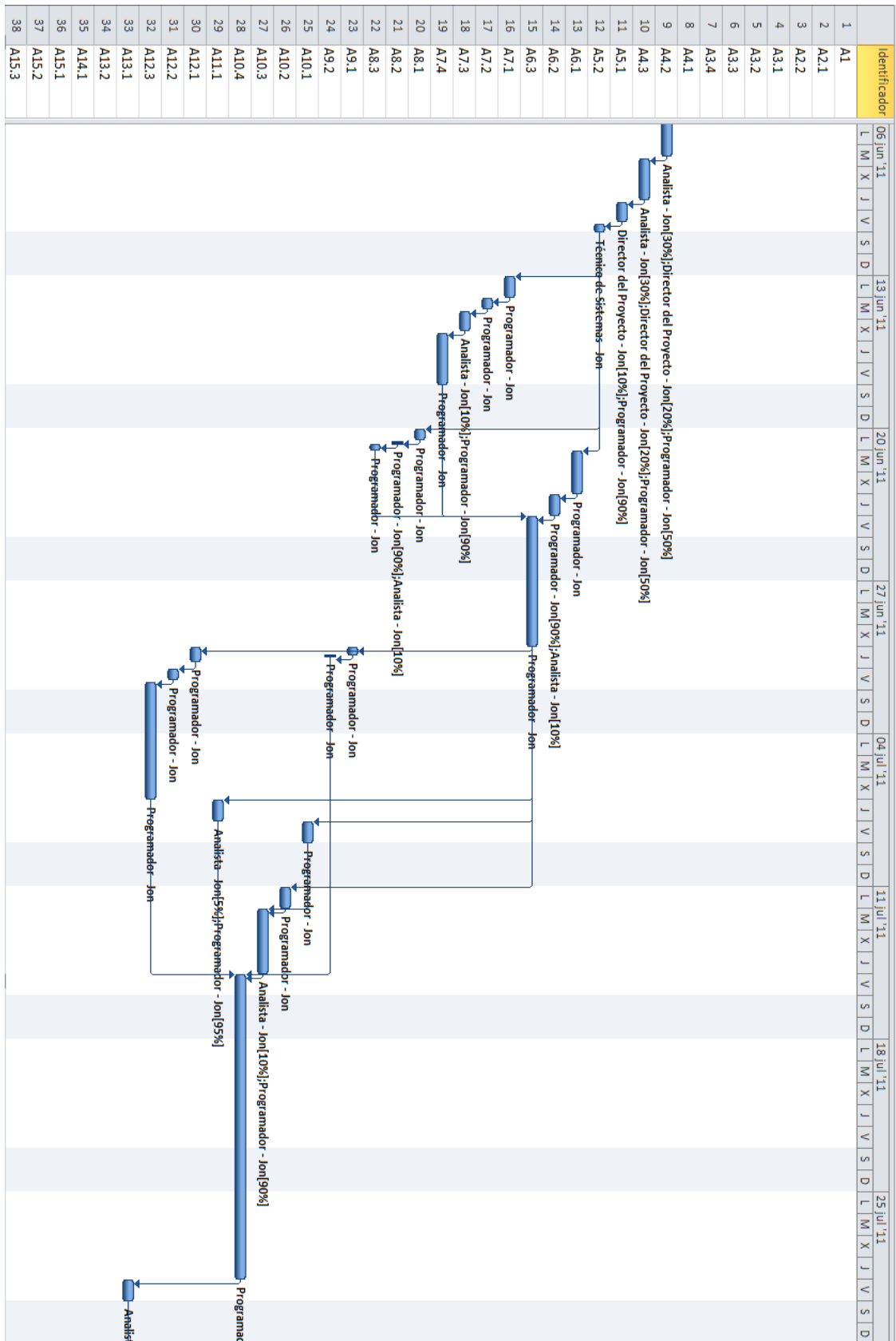


Figura 8.27: Diagrama de Gantt (06/06/2011-31/07/2011)

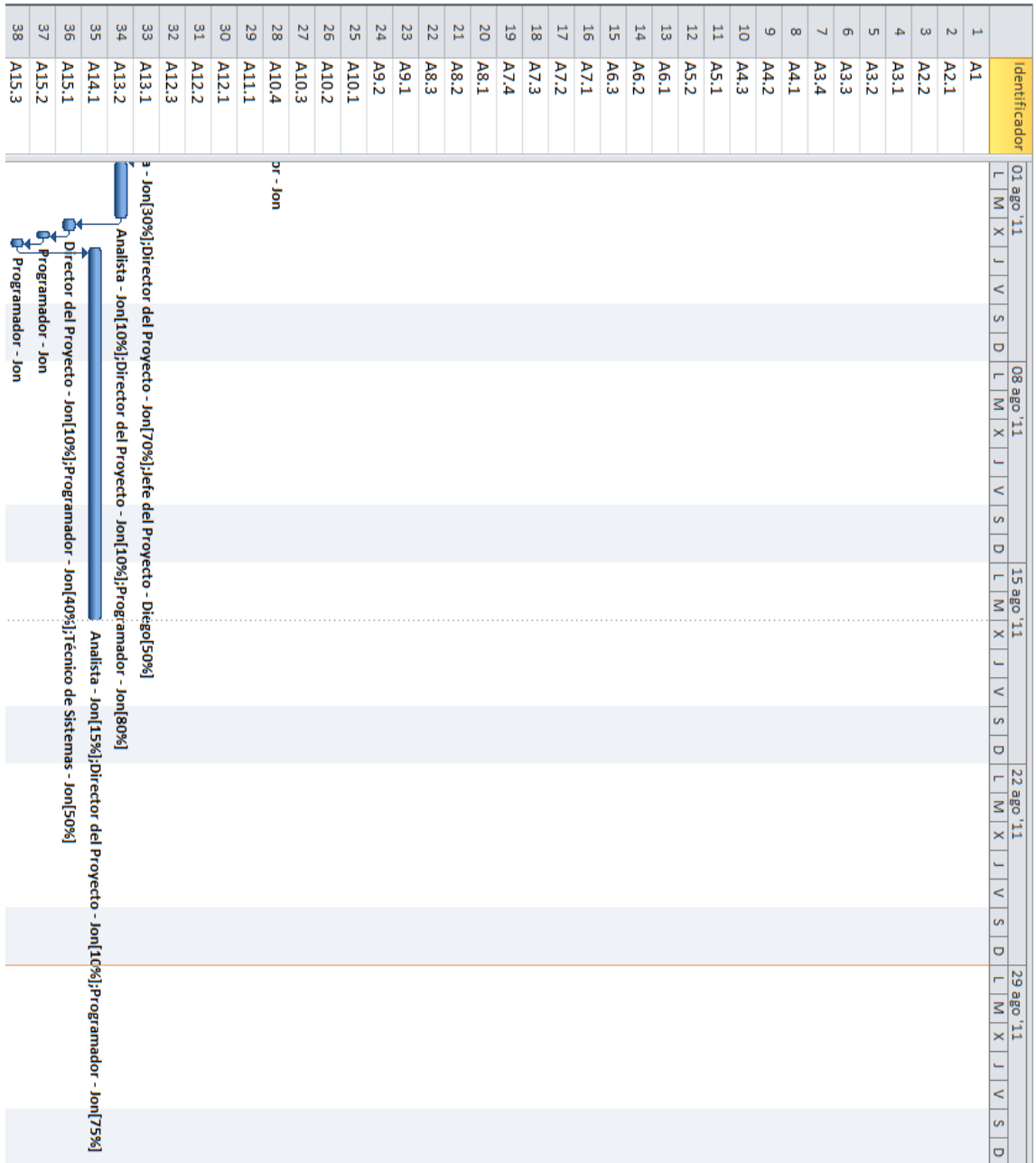


Figura 8.28: Diagrama de Gantt (01/08/2011-16/08/2011)

8.7 DIAGRAMA DE PRECEDENCIAS

El diagrama de precedencias o diagrama de red muestra de forma detallada la estructura de precedencias de las tareas del proyecto e identifica las tareas críticas. Las tareas críticas son aquellas que no pueden retrasarse sin retrasar el tiempo total del proyecto.

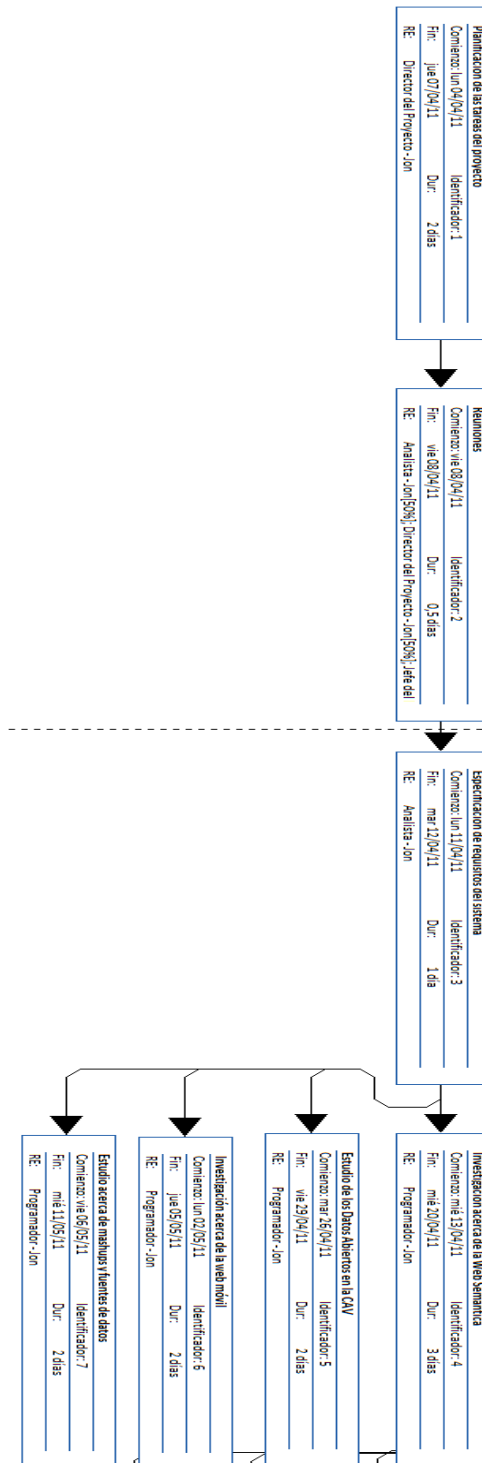


Figura 8.29: Diagrama de precedencias I

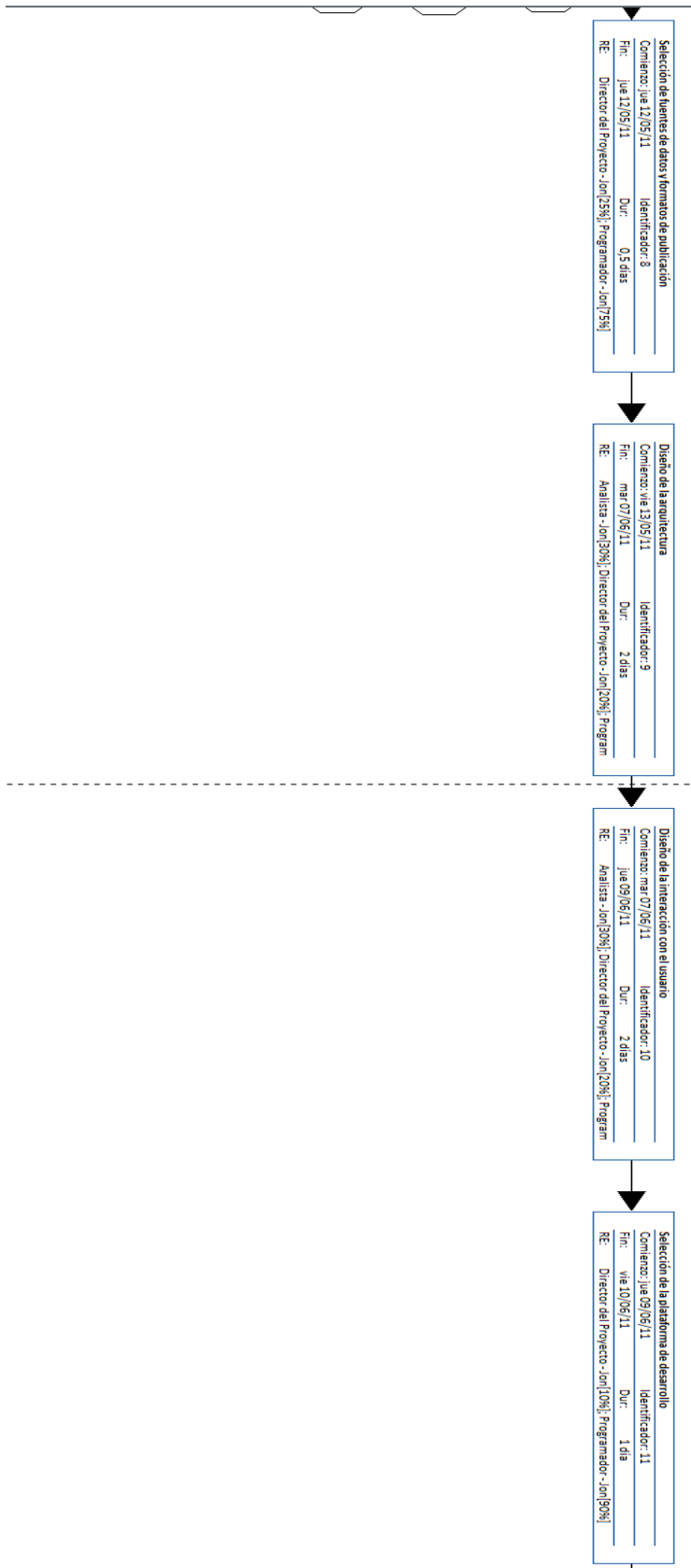


Figura 8.30: Diagrama de precedencias II

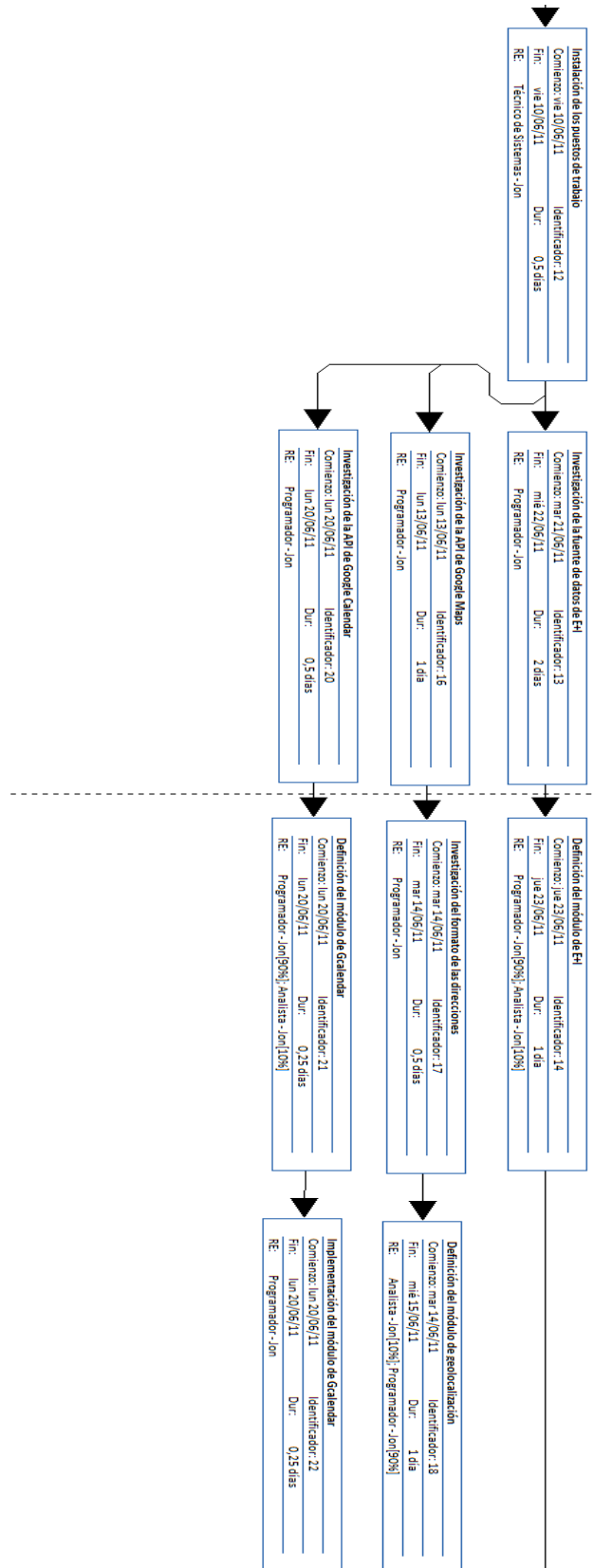


Figura 8.31: Diagrama de precedencias III

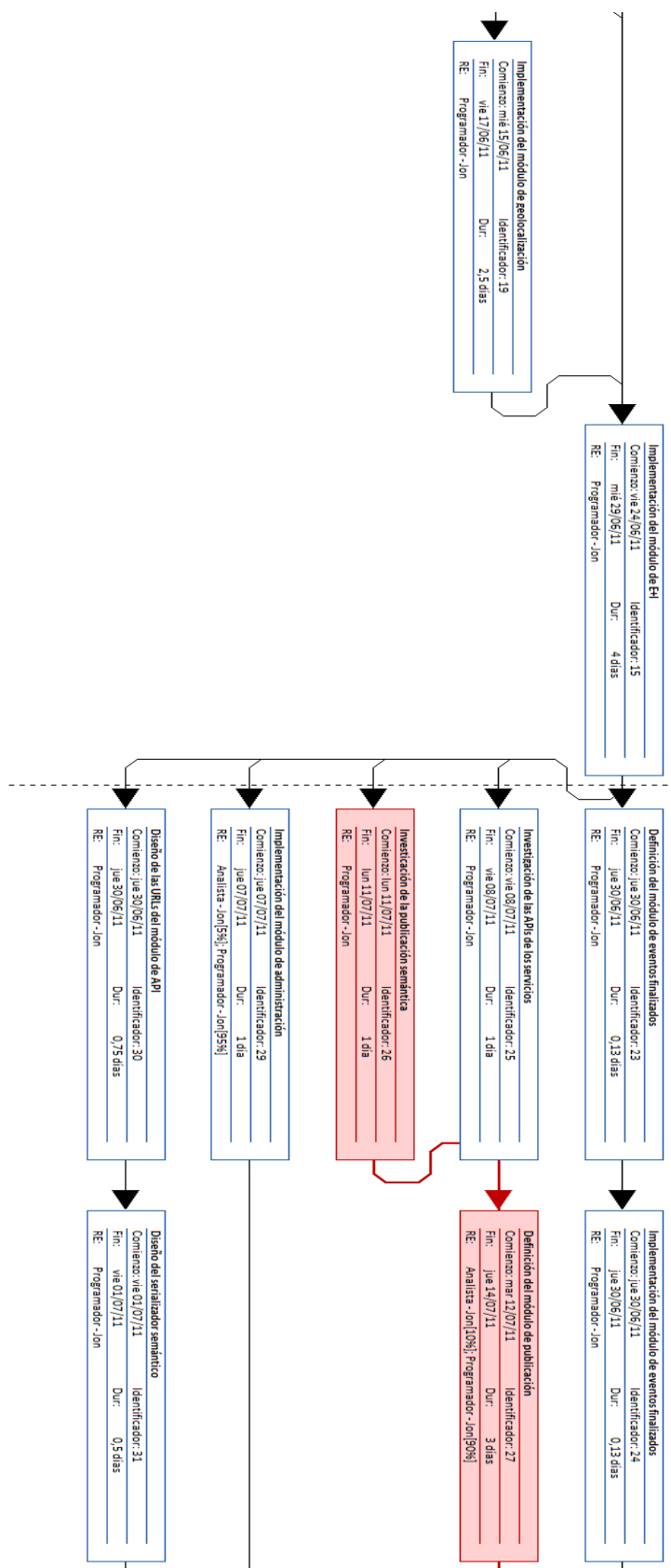


Figura 8.32: Diagrama de precedencias IV

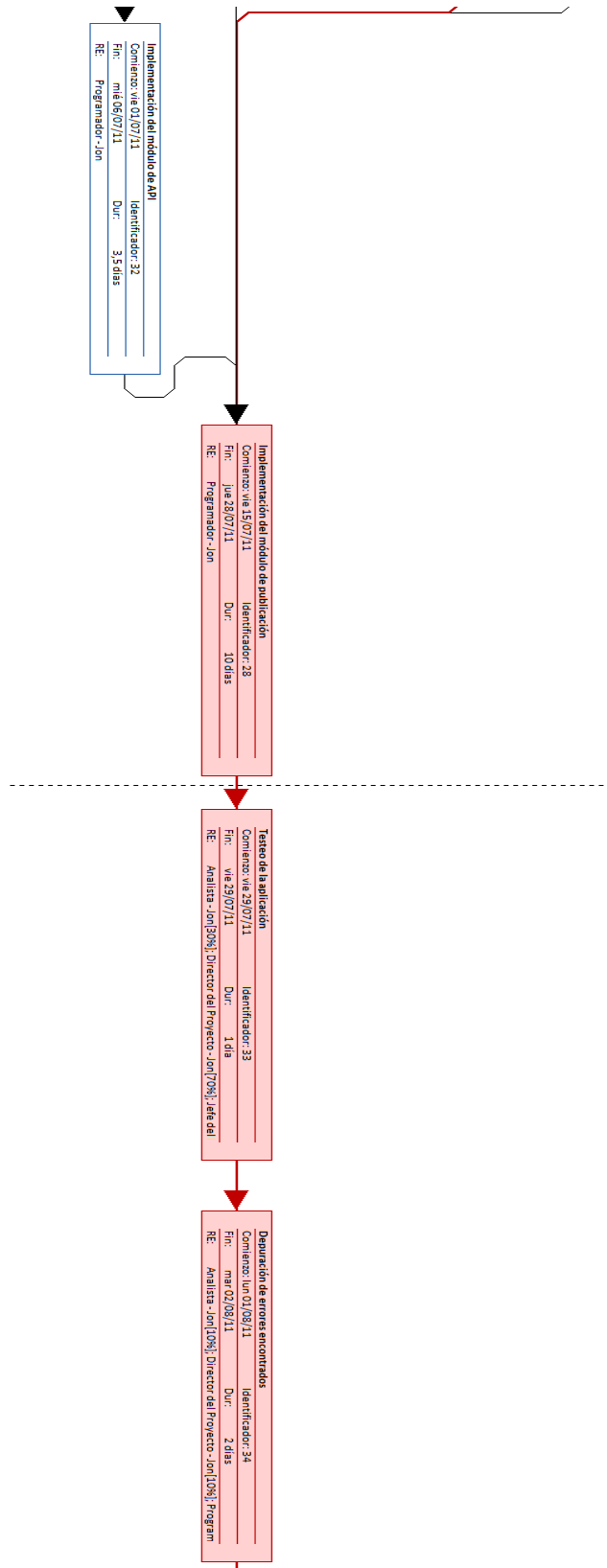


Figura 8.33: Diagrama de precedencias V

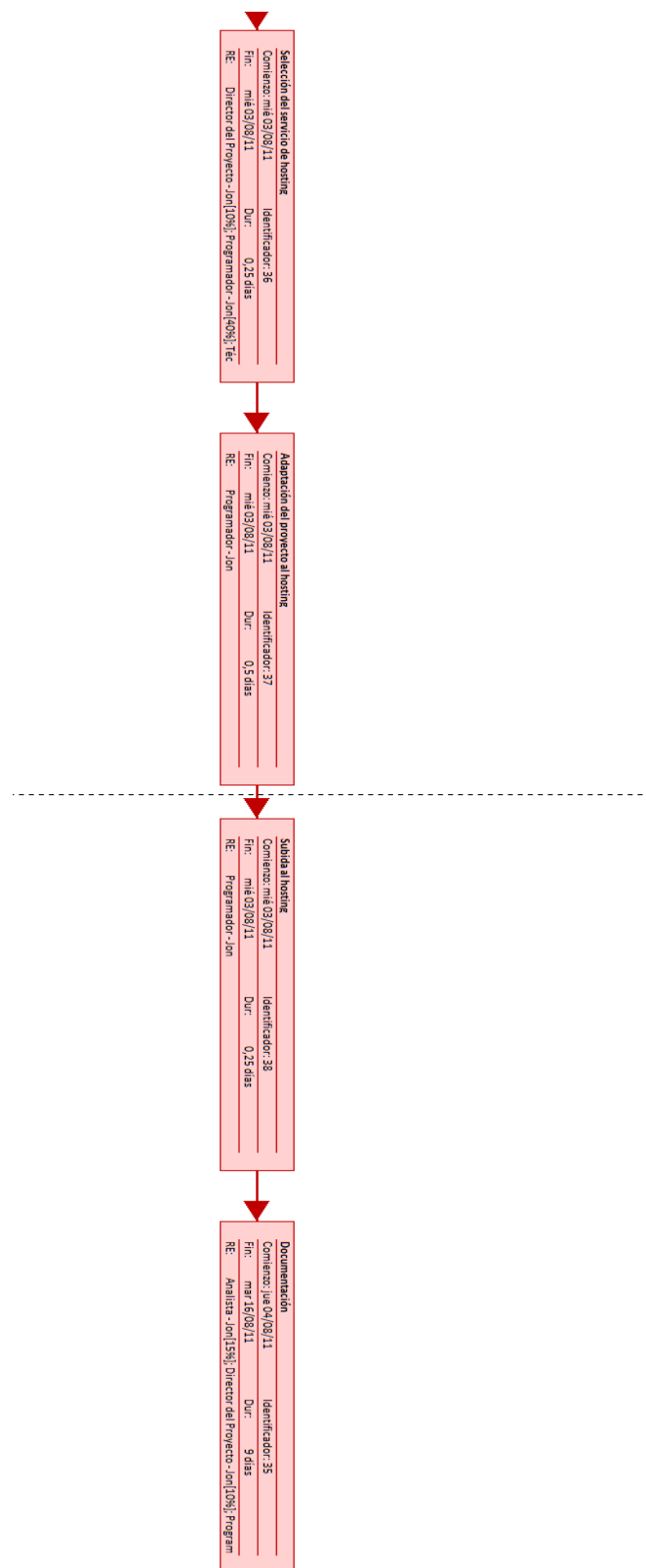


Figura 8.34: Diagrama de precedencias VI

9. CONCLUSIONES Y LÍNEAS FUTURAS

9.1 CONCLUSIONES FINALES

En un mundo como el de la web, en el que el contenido publicado en internet crece exponencialmente, la necesidad de aprovechamiento de dicha información es la que ha motivado la creación de la llamada Web Semántica. Este nuevo paradigma no debe verse como una ruptura con lo anterior, si no como algo complementario, algo que ayuda a mejorar lo presente, y que marca cómo prepararse para el aprovechamiento futuro del contenido publicado en la Web.

De hecho, la realización de este proyecto pretende demostrar que herramientas como los mashups semánticos, como el resultante del desarrollo de este proyecto, son puentes de unión entre la llamada Web 2.0 y la Web Semántica, ya que integra en una sola aplicación web tecnologías de ambas evoluciones, para dar un servicio añadido, mejorando el concepto y adaptándolo a esas nuevas tecnologías web.

Acerca de la Web Semántica, después de investigar y experimentar con varias de las herramientas que la rodea, puedo concluir que, bajo mi punto de vista, se halla en un estado un tanto confuso, con unas líneas claras que seguir, con un mismo objetivo común, pero con muchos estándares y formas de uso por definir; y esto se debe principalmente a que existen diversos puntos de vista acerca de cómo implementarla, y a que muchos de los métodos existentes tienen una complejidad de uso un tanto compleja, y con una curva de aprendizaje bastante pronunciada.

Aun así, era un paso necesario, una evolución que puede traer a la web, y por tanto, a sus usuarios, un beneficio importante en cuanto a usabilidad y aprovechamiento del contenido. De hecho, uno de los movimientos surgidos a partir de la web semántica es otra de las partes importantes de este proyecto: la apertura de datos públicos, o “Open Data”. Se trata de una práctica que supone una forma de aprovechar datos de dominio público para crear aplicaciones de valor añadido que si no fuera por la apertura de dicha información quizás nunca se construirían, y que ayudan a dar a conocer dichos datos de formas diferentes a la habitual. Además, son una forma de dar transparencia a gobiernos e instituciones públicas, lo cual puede repercutir en una evaluación más objetiva del trabajo de dichas instituciones. En resumen, se trata de un movimiento que persigue que la inteligencia colectiva y la participación ciudadana, sirvan para generar valor y riqueza a partir de datos de interés general.

Debido a que aún queda mucho por hacer en cuanto a la apertura de datos en Euskadi (aunque se estén dando avances muy significativos desde las propias instituciones), este proyecto ha tenido parte de su motivación en aportar su granito de arena en este área, extrayendo datos del área de innovación del Gobierno Vasco, y poniéndolos a disposición de la ciudadanía para su reutilización.

En cuanto a la realización del proyecto en sí, cabe comentar que este desarrollo me ha permitido tomar una visión práctica de todas las fases de la creación de un proyecto comenzado desde cero: desde una propuesta de proyecto hasta su paso a producción. El hecho de haber comenzado desde cero, desarrollando la idea de la propuesta de proyecto hasta su implementación final, ha servido, por un lado, como experiencia práctica profesional, ya que el hecho de haber recorrido todos los pasos de un desarrollo software basado en web y haber tenido que lidiar con los problemas y beneficios propios de cada uno, ha servido para encontrarle utilidad práctica a muchas de esas cosas que hemos aprendido durante la carrera acerca de conceptos como el análisis, el diseño, la arquitectura del software, etc.

Pero además de como experiencia profesional, este desarrollo ha servido como una experiencia personal muy valiosa y enriquecedora, ya que el hecho de tratarse del desarrollo completo de una idea basada en una propuesta sin demasiado detalle, ha servido para potenciar la creatividad y, a la postre, para crear un producto con marca propia.

Por otra parte, el hecho de haber sido un proyecto que ha requerido una base importante de investigación, ha hecho que creciera mi interés por conocer nuevas áreas de conocimiento en torno a la tecnología en general y a la informática en particular, y que aumentara mi motivación por explorar y explotar las posibilidades que ofrece la tecnología para mejorar la sociedad.

En conclusión, se ha tratado de un proyecto en general gratificante, en el que se ha ganado tanto en nuevos conocimientos, como en lo personal, y que me ha servido para, por un lado, darle un punto de vista más práctico a los contenidos de la carrera, y por otro lado, para conocer nuevas áreas de la informática no tratados en la universidad.

9.2 LÍNEAS FUTURAS

Algunas de las posibles líneas futuras sobre los que podría evolucionar este proyecto son las siguientes:

- **Almacenaje de RDF:** Sería interesante la utilización de un Sistema de Gestión de Bases de Datos que permitiera el almacenaje de tripletas RDF de forma nativa, para evitar las tareas de exportación dinámica.

- **End-point SPARQL para linked-data:** En caso de mantener una base de datos de almacenaje RDF, sería una buena alternativa crear un end-point SPARQL sobre ella, para favorecer la reutilización y linkado de los datos guardados.
- **Nuevas fuentes de datos:** La adición de fuentes de datos además de Euskadi+Innova, daría valor añadido a esta aplicación, al mantener una mayor base de datos de eventos. Además, la adición de nuevas fuentes de datos podría romper con la limitación geográfica de esta aplicación, abriéndose a usuarios de fuera de la Comunidad Autónoma Vasca.
- **Versiones nativas para smartphones:** Aprovechando que la versión móvil de la web está basada en HTML5 y JavaScript, y que existen herramientas capaces de crear aplicaciones nativas para smartphones a partir de ellas, sería buena idea crear dichas versiones con el objetivo de que consumieran menos recursos y estuvieran más acopladas con el sistema operativo.

10. BIBLIOGRAFÍA

Evolución de la web

- Artículo que analiza la evolución hasta la web 2.0, escrito por Jairo Alexander Ceballos, de la Universidad Nacional de Colombia:

<http://www.virtual.unal.edu.co/unvPortal/articles/ArticlesViewer.do?reqCode=viewDetails&idArticle=2>

- Entrada de blog acerca de las características principales de cada una de las evoluciones de la web:

http://e-global.es/b2b-blog/2005/11/23/caracteristicas-principales-de-web-1_0-web-1_5-y-web-2_0/

- Artículos de la Wikipedia acerca de la historia de la World Wide Web y acerca de la Web 2.0:

http://es.wikipedia.org/wiki/World_Wide_Web#Historia

http://es.wikipedia.org/wiki/Web_2.0

- Paper acerca del concepto de la Web 2.0: “What Is Web 2.0: Design Patterns and Business Models for the Next Generation of Software” por Tim O’Reilly:

http://mpa.ub.uni-muenchen.de/4578/1/MPRA_paper_4578.pdf

- Artículo que analiza la historia de la World Wide Web:

<http://html.conclase.net/articulos/historia>

Web semántica

- Artículo de Dolores Reig acerca del concepto de Web 3.0, con múltiples vídeos explicativos del concepto de Web Semántica:

<http://www.dreig.eu/caparazon/2010/05/11/web-3-0-web-semantica-la-pelicula-traduccion-y-vision-critica/>

- Guía breve acerca de la Web Semántica, publicada por W3C:

<http://www.w3c.es/divulgacion/guiasbreves/websemantica>

- Especificaciones de RDF, SPARQL y OWL, publicadas por W3C:

<http://www.w3.org/RDF/>

<http://www.w3.org/TR/rdf-sparql-query/>

<http://www.w3.org/TR/owl-features/>

- Transparencias de una charla acerca de los conceptos básicos de la Web Semántica, publicadas por W3C:

[http://www.w3.org/2008/Talks/0616-CWI/HCLS#\(1\)](http://www.w3.org/2008/Talks/0616-CWI/HCLS#(1))

- Paper acerca de las características de la web semántica: “La Web Semántica”, por Pablo Castells, de la Universidad Autónoma de Madrid:

<http://arantxa.ii.uam.es/~castells/publications/castells-uclm03.pdf>

- Portal acerca de la iniciativa Linked Data:

<http://linkeddata.org/>

- Artículo comparativo entre Microformatos, RDFa y Microdatos HTML5:

<http://blog.foolip.org/2009/08/23/microformats-vs-rdfa-vs-microdata/>

- Artículos de la Wikipedia acerca de Web Semántica, GRDDL y Linked Data:

http://es.wikipedia.org/wiki/Web_sem%C3%A1ntica

http://es.wikipedia.org/wiki/Datos_vinculados

<http://en.wikipedia.org/wiki/GRDDL>

Open Data

- Portal de las webs del Gobierno Vasco:
<http://www.euskadi.net>
- Portal web de Open Data Euskadi:
<http://opendata.euskadi.net/w79-home/eu>
- Artículo en la Wikipedia acerca de Open Data:
http://en.wikipedia.org/wiki/Open_data
- Blog acerca de la apertura y reutilización de datos:
<http://datos.fundacionctic.org/>
- Portal de la Open Data Foundation:
<http://www.opendatafoundation.org/>

Mashups

- Artículo de la Wikipedia acerca de mashups:
[http://es.wikipedia.org/wiki/Mashup_\(aplicaci%C3%B3n_web_h%C3%ADbrida\)](http://es.wikipedia.org/wiki/Mashup_(aplicaci%C3%B3n_web_h%C3%ADbrida))
- Artículo acerca de fuentes de datos y tipos de mashups:
<http://www.techtear.com/2007/03/26/los-mashups-uno-de-los-pilares-de-la-web-20>
- Webs de las APIs de Google Maps y Google Calendar:
<http://code.google.com/intl/es/apis/maps/>
<http://code.google.com/intl/es-ES/apis/gdata/>
- Transparencias acerca de la creación de Mashups Semánticos:
http://assets.en.oreilly.com/1/event/3/Creating%20Semantic%20mashups_%20Bridging%20Web%202.0%20and%20the%20Semantic%20Web%20Presentation%201.pdf

11. AGRADECIMIENTOS

No se puede finalizar este documento sin dar las gracias a todas las personas que han sido partícipes directos o indirectos del desarrollo del proyecto.

En primer lugar, me gustaría dar las gracias a todas las personas que indirectamente han ayudado en el desarrollo de este proyecto mediante la publicación de sus investigaciones y artículos acerca de los temas tratados. Del mismo modo, agradecer también a todos los desarrolladores que comparten su software con los demás bajo licencias libres. Su trabajo ha constituido la base para la consecución de este proyecto.

En segundo lugar, agradecer también al director del proyecto, Diego López-de-Ipiña, por la propuesta y por los recursos facilitados.

Agradecer también a mis compañeros de Universidad su compañerismo y ayuda en todo momento.

Y finalmente, un especial agradecimiento a mi familia y amigos más cercanos, por la paciencia, el apoyo incondicional y los ánimos que me han transmitido, no solo durante la realización del Proyecto de Fin de Carrera, sino durante estos cinco años de carrera.

Muchas gracias a todos.

ANEXO I: MANUAL DE USUARIO

Versión de escritorio de la aplicación web

Página inicial (Mapa)

Nada más entrar en <http://www.innovagenda.com>, el visitante verá un mensaje de su navegador preguntándole si desea o no desea compartir su ubicación con el sitio.

Si decide compartir su ubicación, InnovAgenda mostrará la siguiente interfaz, con el mapa centrado en su ubicación:



Pantalla inicial de InnovAgenda

Si en caso de decidir no compartir la ubicación, el mapa se centrará en el centro geográfico de la Comunidad Autónoma Vasca.

Debe saberse que la información acerca de la localización geográfica del usuario sólo se utiliza para personalizar el mapa y que nunca es almacenada para ningún otro tipo de uso.

En esta primera vista de bienvenida se muestra un mapa con la localización de los próximos eventos. Las localizaciones en las cuales hay más de un evento, aparecen marcadas con el número de eventos próximos que van a ocurrir en dicho lugar.

Además del mapa, se puede visualizar el mensaje de bienvenida a la página, en el que se explican las funcionalidades de la web y sus diferentes zonas, y un recuadro con los próximos eventos.

En esta vista se puede observar la barra superior, que está formada por el título de la web, una serie de pestañas, y un recuadro de selección de idioma.



Barra superior de InnovAgenda

Esta barra superior es común a todas las vistas de la versión de escritorio, y es el acceso a las diferentes partes de la web.

Eventos

En la pestaña de Eventos, se muestra la lista de los eventos no finalizados, ordenados por la fecha de inicio, de más cercana a la más tardía:



Lista de eventos

Cada link de esta lista lleva a una visualización individual de los detalles de cada evento. Esta vista está formada por un mapa estático no navegable y por los diferentes datos del evento:

CURSO | CREA, EDITA Y COMPARTE DOCUMENTOS DE TU NEGOCIO CON GOOGLE DOCS.

Lugar: Centro KZGUNEA ERNEST LLUCH - Paseo de Anoeta 7 - 20014



Fecha inicio: 2011-09-20
Fecha fin: 2011-09-20
Horario: 14:00:00 - 16:00:00
Duración: 2 h
Precio: 0 €
Temática: Empresa Digital
Idioma de impartición: Castellano
Agencia Organizadora: SPRI
Agencia Coordinadora: SPRI, S.A. (Formación)
Impartido por: SIC S.A
[Información adicional](#)

Almacena tus documentos en un servidor de Google para que estén disponibles desde cualquier ordenador y otras personas puedan trabajar con ellos.

Toda la información acerca de este evento está publicada en la notación de HTML5 Microdata [Schema.org](#)

Evento visible en formato:



Y en notación [RDFa](#)

Evento exportable a [Google Calendar](#)

Compartir



Añadir Comentario

[Ingresar](#)



Escriba su comentario.

Mostrando 0 comentarios

Ordenar por: los más populares

[Suscribirse por email](#) [RSS](#)

Visualización de un evento

Las opciones de exportación a formatos semánticos se incluyen en la derecha de esta pantalla, las opciones son las siguientes:

- Exportación a RDF/XML, que devuelve la información del evento en dicho formato.
- Exportación a N-Triples, que devuelve la información del evento en dicho formato.
- Exportación a N3, que devuelve la información del evento en dicho formato.
- Exportación a Turtle, que devuelve la información del evento en dicho formato.
- Visualización de la Versión RDFa, que recarga la misma página pero con notación interna diferente. Para el usuario la interfaz apenas sufrirá cambios.

Además de las opciones de exportación a formatos semánticos, se ofrece la opción de exportar el evento individual a un calendario de Google Calendar.


Por último, esta vista incluye el sistema de comentarios basado en Disqus, y las opciones de compartición en redes sociales.

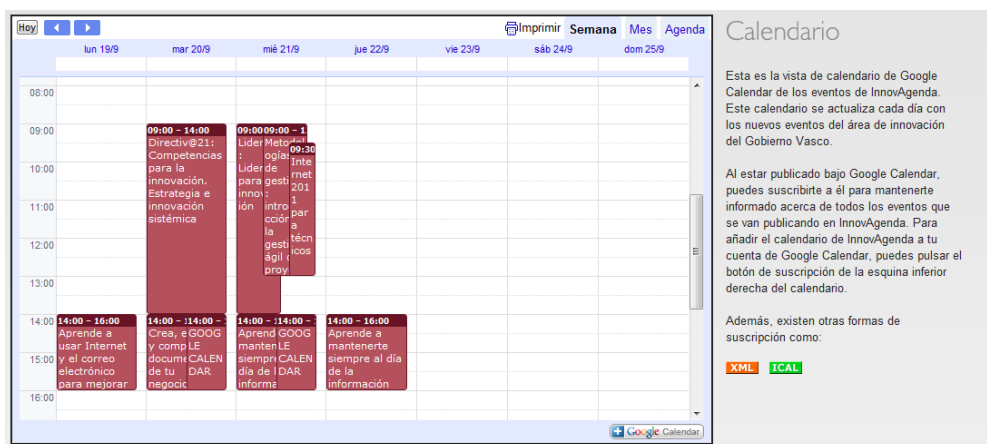
Para comentar existen varias formas de autenticación, y permite dar opinión acerca de cada evento. Se visualizan los mismos comentarios para la versión en castellano y para la versión en euskera.

Para compartir en las redes sociales Twitter, Facebook y Google+, tan solo hay que pulsar en el botón de la red social deseada.

Calendario

En la pestaña del Calendario, se muestra una visualización del calendario de Google Calendar correspondiente al idioma en el que se está navegando, además de una serie de opciones de exportación del calendario:

- Si se quiere exportar el calendario completo al Google Calendar del usuario, se debe pinchar en el botón  de la esquina inferior derecha del calendario.
- Si en cambio se quiere exportar el calendario a otro servicio de calendarios, se ofrece la posibilidad de exportarlo en formato XML y en formato ICAL, opciones a las cuales se accede pinchando sus respectivos botones.



Calendario de Google Calendar y opciones de exportación

Buscador de eventos

En la pestaña del buscador de eventos, se accede a un formulario para realizar búsquedas personalizadas sobre los eventos. En la parte izquierda se deben elegir los valores, y a la derecha aparecerán los resultados:

Formulario de búsqueda de eventos

Documentación de la API

La pestaña API muestra la documentación de la API con varios ejemplos aclarativos. Para más información acerca del funcionamiento de la API, ver capítulo 4 “Manual de la API”

API de InnovAgenda	Ejemplos
<p>Dado que la web del área de innovación del Gobierno Vasco, Euskadi+Innova, solo publica sus datos en texto plano, desde InnovAgenda ofrecemos a usuarios y desarrolladores una nueva forma de acceder a estos datos mediante una sencilla API.</p> <p>Además, la API de InnovAgenda no solo ofrece la posibilidad de acceder a estos datos en diversos formatos tradicionales como JSON, XML o YAML, si no que incluye también la posibilidad de recoger esos datos en formato RDF, sobre los cuales se pueden hacer consultas SPARQL.</p> <p>La forma de acceder a la API es mediante consultas del tipo: www.innovagenda.com/api/v1/event/?CONSULTA</p>	<p>Recoger todos los eventos en formato XML y en Euskera:</p> <p>www.innovagenda.com/api/v1/event/?format=xml&lang=eu</p> <p>Recoger todos los eventos que contengan en su nombre en castellano la palabra 'Google':</p> <p>www.innovagenda.com/api/v1/event/?</p>

Documentación de la API

Contacto

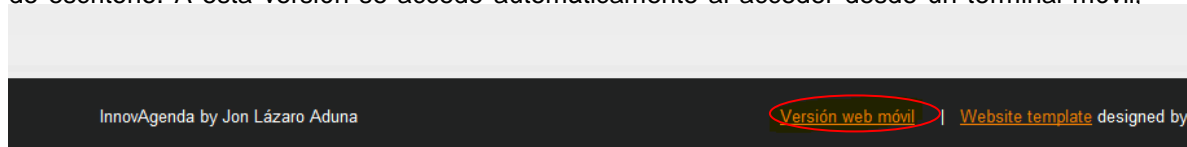
La pestaña de contacto muestra la información de contacto de InnovAgenda, y un formulario de contacto desde el cual se puede enviar un mail a los administradores de la web.

Contacta con nosotros	Formulario de Contacto
<p>E-mail: info@innovagenda.com</p> <p>Puedes contactar con nosotros para que recibamos tus dudas y/o sugerencias, mediante nuestra dirección de email o mediante el formulario de contacto. Muchas gracias.</p>	<p>Nombre: <input type="text"/></p> <p>E-mail: <input type="text"/></p> <p>Mensaje: <input type="text"/></p> <p><input type="button" value="Enviar"/></p>

Formulario de contacto

Versión móvil de la aplicación web

La versión móvil de InnovAgenda es una versión reducida de los contenidos de la web de escritorio. A esta versión se accede automáticamente al acceder desde un terminal móvil.



La versión móvil de InnovAgenda se abre en un menú de opciones desde el cual se accede a las diferentes partes de la web. En esta misma pantalla de bienvenida se puede realizar la selección de idioma, y se puede elegir el ver la versión completa de la web.



Menú de la versión móvil de InnovAgenda

Las visualizaciones a las que se puede acceder desde este menú son:

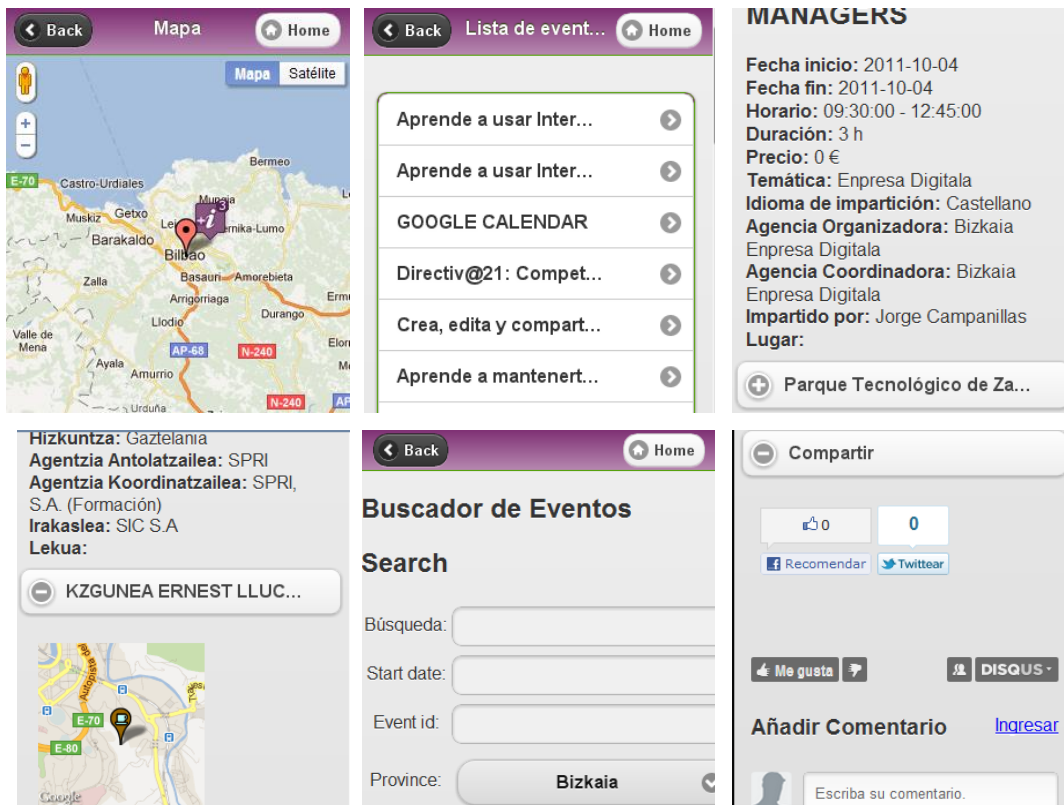
- Mapa
- Lista de eventos
- Búsqueda
- Calendario

La navegación entre las vistas se realiza mediante botones que incluye la propia interfaz web. Todas las vistas incluyen la siguiente barra superior, con botones para la pantalla de inicio y para retroceder.



Barra de navegación superior de la versión móvil de InnovAgenda

Algunas de las visualizaciones de las partes de la aplicación son las siguientes:



Visualizaciones de la versión móvil de InnovAgenda

Zona de administración del sitio

Pantalla de Login

Para acceder a la zona de administración de InnovAgenda, se deberá acceder a la URL <http://www.innovagenda.com/admin>. Lo primero que se verá nada más acceder a esta web es la pantalla de login, dónde habrá que meter usuario y contraseña para continuar:

Pantalla de login del módulo de administración

Inicio de la administración

Una vez autenticado en el sistema, el usuario administrador verá todas las opciones de administración disponibles para la aplicación:

InnovAgenda

Sitio administrativo

Auth	
Grupos	+ Añadir ✎ Modificar
Usuarios	+ Añadir ✎ Modificar

Sites

Sitios	+ Añadir ✎ Modificar
--------	--

Website

Activity types	+ Añadir ✎ Modificar
Agency	+ Añadir ✎ Modificar
Events	+ Añadir ✎ Modificar
Places	+ Añadir ✎ Modificar
Provinces	+ Añadir ✎ Modificar
Topics	+ Añadir ✎ Modificar

Traductor

Empezar a traducir

Acciones recientes

Mis acciones

Ninguno disponible

Pantalla de inicio del módulo de administración

Cómo se puede observar, desde esta pantalla se tiene acceso a la gestión de usuarios, y a la gestión de las diferentes tablas de la base de datos modificables. Además, se incluye la opción de acceder al módulo de traducción del sitio.

Altas, bajas y modificaciones

Una vez seleccionado el contenido a modificar, se debe hacer click en el link, y se mostrará una vista con el siguiente estilo:

InnovAgenda

Bienvenido/a, [Jonadmin](#). [Cambiar contraseña](#) / [Terminar sesión](#)

Inicio > Website > Events

Escoja event a modificar

Añadir event +

Acción: [dropdown] [icon] seleccionados 0 de 21

<input type="checkbox"/>	Event
<input type="checkbox"/>	2011-10-03 Ecodiseño en el sector Máquina-Herramienta
<input type="checkbox"/>	2011-09-22 Aprende a mantenerte siempre al día de la información para tu negocio: Alertas de redes sociales y sindicación de contenidos web
<input type="checkbox"/>	2011-09-21 Internet 2011 para técnicos
<input type="checkbox"/>	2011-09-21 GOOGLE CALENDAR
<input type="checkbox"/>	2011-09-21 Aprende a mantenerte siempre al día de la información para tu negocio: Alertas de redes sociales y sindicación de contenidos web
<input type="checkbox"/>	2011-09-20 GOOGLE CALENDAR
<input type="checkbox"/>	2011-09-20 Crea, edita y comparte documentos de tu negocio con Google Docs.
<input type="checkbox"/>	2011-09-19 Aprende a usar Internet y el correo electrónico para mejorar tu negocio.
<input type="checkbox"/>	2011-09-16 Aprende a usar Internet y el correo electrónico para mejorar tu negocio.
<input type="checkbox"/>	2011-10-04 Barnetegi Tecnológico (Conv. abierta): "Descubra el valor estratégico de las Nuevas Tecnologías"
<input type="checkbox"/>	2011-10-04 Herramientas Jurídicas Para Community Managers
<input type="checkbox"/>	2011-10-04 Directiv@21: Competencias para la innovación. Gestión de la innovación

Pantalla de administración del modelo de eventos

Para añadir un nuevo valor a la tabla, el botón de la esquina superior derecha abre un diálogo para la inserción de los valores para el nuevo registro.

Para modificar alguno de los valores ya insertados en la tabla, basta con hacer click encima del registro para poder acceder a la vista de modificación, similar a la de inserción.

Y si lo que se quiere es borrar algún valor, lo que se debe hacer es seleccionarlos marcando los ticks a la izquierda de los registros, y seleccionar en las acciones disponibles, la acción de borrar.

Diálogo de inserción de nuevo evento

Sitio de traducción

Al sitio de traducción se accede desde la pantalla principal de administración de InnovAgenda. Nada más entrar en el sitio de traducción, se ven los diferentes idiomas a los que se puede traducir la web. Para empezar a traducir basta con pulsar el idioma deseado.

Rosetta						
Inicio > Selección de idioma						
Castellano						
Application	Progreso	Mensajes	Traducido	Revisar	Obsoleto	Archivo
Current	2.00%	73	2	0	1	/home/dotcloud/rsync-1314710438.61/locale/es/LC_MESSA
Euskera						
Application	Progreso	Mensajes	Traducido	Revisar	Obsoleto	Archivo
Current	86.00%	73	63	0	4	/home/dotcloud/rsync-1314710438.61/locale/eu/LC_MESSA

Rosetta 0.6.2 SVN-unknown

Inicio del sitio de traducción

Una vez seleccionado el idioma al que traducir, se visualizará una pantalla con los valores a traducir en la izquierda, y cajas de texto a la derecha en los que introducir los valores traducidos.

Traducir al Euskera		Pantallas:
Original	Euskera	Revisar
Spanish	Gaztelania	<input type="checkbox"/>
	sugerir	
Basque	Euskera	<input type="checkbox"/>
	sugerir	
Mapa	Mapa	<input type="checkbox"/>
	sugerir	
Eventos	Ekintzak	<input type="checkbox"/>
	sugerir	
Calendario	Egutegia	<input type="checkbox"/>
	sugerir	
Buscador	Bilatzaila	<input type="checkbox"/>

Zona de traducción

Cuando se termine la traducción basta con pulsar el botón de guardado, y los cambios se verán reflejados en la web.

Manual de la API de InnovAgenda

Dado que la web del área de innovación del Gobierno Vasco, Euskadi+Innova, solo publica sus datos en texto plano, desde InnovAgenda ofrecemos a usuarios y desarrolladores una nueva forma de acceder a estos datos mediante una sencilla API.

Además, la API de InnovAgenda no solo ofrece la posibilidad de acceder a estos datos en diversos formatos tradicionales como JSON, XML o YAML, si no que incluye también la posibilidad de recoger esos datos en formato RDF, sobre los cuales se pueden hacer consultas SPARQL.

La forma de acceder a la API es mediante consultas del tipo: www.innovagenda.com/api/v1/event/?CONSULTA

El valor de CONSULTA puede tener 2 formatos: '/ID_DEL_EVENTO/' si es que conocemos el identificador numérico del evento concreto al que queremos acceder, o bien, mediante el formato '/?CLAVE__OPERACION=VALOR'

En las consultas del segundo tipo, la CLAVE puede ser 'name_es' (Título del evento en castellano), 'name_eu' (Título del evento en euskera), 'start_date' (Fecha de comienzo del evento) o 'finish_date' (Fecha de fin del evento).

La __OPERACION es un parámetro opcional que puede tener valores de comparación como '___startswith' (Para especificar que lo buscado empieza por VALOR), '___contains' (Para especificar que lo buscado contiene VALOR), '___gt' (Para especificar que lo buscado es mayor que VALOR), '___lt' (Para especificar que lo buscado es menor que VALOR), '___gte' (Para especificar que lo buscado es mayor o igual que VALOR) o '___lte' (Para especificar que lo buscado es menor o igual que VALOR). Siendo el VALOR es lo que se quiere encontrar. Cabe comentar que es posible concatenar CONSULTA-s mediante el carácter '&'

Para especificar el formato en que se quieren recibir los datos se debe incluir al final 'format=FORMATO'. Si se quiere especificar el idioma en el que se quieren recibir los datos (es opcional), se debe incluir 'lang=IDIOMA'.

Ejemplos de uso:

- Recoger todos los eventos en formato XML y en Euskera:

www.innovagenda.com/api/v1/event/?format=xml&lang=eu

- Recoger todos los eventos que contengan en su nombre en castellano la palabra 'Google':

www.innovagenda.com/api/v1/event/?name_es__contains=Google&format=json

- Recoger el evento 4435:

www.innovagenda.com/api/v1/event/4435/?format=rdf

- Recoger todos los eventos cuya fecha de fin sea 2011-09-15

www.innovagenda.com/api/v1/event/?finish_date=2011-09-15&format=xml

- Recoger todos los eventos cuya fecha de inicio sea mayor que 2011-09-10

www.innovagenda.com/api/v1/event/?start_date__gt=2011-09-10?format=yaml

ANEXO II: PASO A PRODUCCIÓN DEL PROYECTO

Para el paso a producción de la aplicación web se siguieron los siguientes pasos:

1. Selección del servicio de hosting
2. Adaptación del proyecto a las necesidades del servicio de hosting
3. Subida del proyecto
4. Creación y configuración del dominio www.innovagenda.com

Selección del servicio de hosting

Al tratarse de un proyecto basado en Django, requiere de ciertas necesidades especiales, como la posibilidad de ejecución en servidor de código Python. Para posibilitar el hosting de aplicaciones Django, es necesario que el servicio sea compatible con WSGI.

WSGI es una especificación para conectar servidores HTTP como Apache o IIS con aplicaciones y frameworks web basados en Python, como el caso de InnovAgenda.

Debido a esta necesidad existen múltiples servicios de hosting especializados en Python y Django. Dentro de éstos, existe una gama creciente de servicios que, siguiendo la filosofía de Django de simplificar la creación de webs, ofrecen la posibilidad de desplegar proyectos de forma sencilla sin necesidad de preocuparse por la gestión del servidor. El problema de muchos de estos servicios es que están aún en fase de pruebas.

Teniendo en cuenta que en el momento de paso a producción de la aplicación la demanda y el tráfico generado no iban a ser demasiado grandes, se decidió optar por uno de estos servicios, teniendo en cuenta que siempre es posible el cambio de plataforma si así se requiere.

Existen varias alternativas que ofrecen estos servicios: Ep.io, Gondor.io, DotCloud, AppHosted... Tras el estudio de dichas alternativas, y la lectura de la siguiente comparativa acerca de estos servicios (<http://kencochrane.net/blog/2011/06/django-hosting-roundup-who-wins/>), se decidió optar por la opción de DotCloud, por las siguientes razones:

- De entre las alternativas, es de las únicas que ha salido ya de beta.
- Múltiples opciones de configuración.
- Documentación clara.
- Acceso SSH al servidor.
- Opción de hosting gratuito.
- Soporte de dependencias vía PIP.

Adaptación del proyecto

Para adaptar el proyecto a su subida a DotCloud, fue necesaria la creación de un archivo YAML de especificación de los servicios a utilizar, así como un archivo de configuración de WSGI.

Los servicios a utilizar especificados en el archivo “dotcloud.yml” son 3: uno para la web basada en Django, otro para el demonio que ejecute las tareas periódicas del servidor (los módulos de ejecución en servidor) y otro para la base de datos basada en PostgreSQL.

El archivo de configuración de WSGI es un simple archivo .py en el que se especifica el archivo de configuración del proyecto a utilizar.

Para el entorno de producción, se creó un nuevo archivo de configuración, para especificar las opciones concretas de dicho entorno. Los cambios que incluye respecto al del entorno de desarrollo son los siguientes:

- Deshabilitación del modo de “debug”.
- Conexión a la base de datos de producción. Los datos acerca de la conexión los proporciona DotCloud en un archivo .json que se guarda en una ubicación por defecto del servidor, por lo que solo fue necesaria la inclusión de los métodos de lectura de dicho archivo.
- Cambio de rutas a directorios de python y django

Otra de las restricciones de DotCloud es que los archivos estáticos se encuentren en una carpeta llamada “static” en la raíz del proyecto. Para cumplir con ello no hubo que realizar ningún cambio en el proyecto, ya que se había seguido esa estructura desde el principio.

Por último, para el manejo de dependencias, DotCloud necesita de un archivo requirements.txt que especifique las dependencias en formato PIP. Este archivo se había ido creando según se utilizaban las diferentes librerías, por si era necesario para casos como este.

Subida del proyecto

Para subir el proyecto, fue necesaria la creación del entorno del proyecto en el servidor de DotCloud. Para ello fue necesaria la instalación del cliente de línea de comandos que proporciona DotCloud.

Con el comando “create” de dicha herramienta se puede crear el entorno, y con tan solo ejecutar el comando “push” comienza el despliegue de la aplicación.

La propia plataforma se encarga de detectar e instalar las dependencias automáticamente, así como de la configuración de WSGI.

Una vez subida y desplegada la aplicación, DotCloud te proporciona una URL dentro de su dominio, desde la cual se puede acceder a la web.

Creación y configuración del dominio

Para terminar, se decidió comprar el dominio innovagenda.com para su utilización como punto de acceso a la web. El dominio se compró mediante el gestor de dominios GoDaddy.

Junto con el dominio, GoDaddy proveía de una dirección de email: info@innovagenda.com. Se decidió adaptar esta dirección de email a las tecnologías de Google Apps, para poder trabajar en torno a esta dirección para la gestión de los calendarios de la versión de producción de InnovAgenda. Para este proceso, tanto Google como GoDaddy requieren de una serie de configuraciones para habilitar el registro de dicha dirección como una cuenta de Google y habilitar su gestión como cuenta de GMail. Estos pasos se realizan mediante el seguimiento del tutorial de Google para dicha tarea.

Una vez comprado el dominio y configurada la cuenta de email, el último paso era la configuración de DotCloud para que se utilizara ese alias para acceder al proyecto. La utilidad de línea de comandos de DotCloud provee el comando “alias” para esta tarea. Este comando configura el proyecto para el acceso desde el dominio especificado, y da las instrucciones de configuración para el acceso DNS, gestión que se debe hacer desde el panel de control de GoDaddy.

En resumen, el paso a producción de la aplicación resultó ser un proceso relativamente sencillo, gracias sobre todo al servicio de hosting DotCloud, que además de proveer de un servicio estable, flexible y seguro, facilitó de sobremanera tareas de configuración de servidor que de forma manual hubieran llevado mucho más tiempo.

La versión de producción de la web está disponible en: <http://www.innovagenda.com>.