



Título: Gestor de Revistas y Congresos en Joomla

Volumen: 1

Alumno: Nick Avalos Rojas

Director/Ponente: Enrique Mayol Sarroca

Codirectora: M^a José Casany

Departamento: Enginyeria de Serveis i Sistemes d'Informació

Titulación: Ingeniería Informática

Fecha: 18 de junio de 2014

DADES DEL PROJECTE

Títol del Projecte: *Gestor de Revistas y Congresos en Joomla*

Nom de l'estudiant: *Nick Avalos Rojas*

Titulació: *Enginyeria en Informàtica*

Crèdits: *37.5*

Director/Ponent: *Enrique Mayol Sarroca*

Codirectora: *M^a José Casany*

Departament: *Enginyeria de Serveis i Sistemes d'Informació*

MEMBRES DEL TRIBUNAL *(nom i signatura)*

President: *Marc Alier Forment*

Vocal: *Jorge García Vidal*

Secretari: *Enrique Mayol Sarroca*

QUALIFICACIÓ

Qualificació numèrica:

Qualificació descriptiva:

Data: *18-06-2014*

Agradecimientos

*A la gran parte de mi familia que tengo a miles de kilómetros
pero que siempre me han mostrado su cercanía*

*A los directores del proyecto, Enric y M^o José por la orientación y
por confiar en mí para este proyecto.*

*Pero quería agradecer principalmente a mi madre Fabiola Rojas,
por su esfuerzo, dedicación y la paciencia que ha
mostrado conmigo a lo largo de los años.*

Índice de Contenido

1 Introducción	1
1.1 Contexto	2
2 Visión	3
2.1 Descripción del problema.....	3
2.2 GesCORE como posible solución	5
3 Objetivos	7
4 Planificación del Proyecto	8
4.1 Organización Temporal del Proyecto.....	8
4.2 Seguimiento.....	11
4.3 Estimación de Costes.....	12
5 Requerimientos	15
5.1 Requisitos Funcionales.....	15
5.2 Requisitos No Funcionales	17
6 Especificación	18
6.1 Modelo Conceptual	19
6.2 Casos de uso	26
7 Diseño	48
7.1 Arquitectura.....	49
7.2 Diseño del modelo de comportamiento.....	51
7.3 Patrones de diseño del sistema	57
7.4 Diseño de las interfaces del proyecto	58
7.5 Diseño de la BBDD.....	65
8 Implementación y Funcionamiento	67
8.1 Tecnologías del Proyecto	68
8.2 APIgesCORE.....	68
8.3 Gestor de Call For Papers.....	82
8.4 Método de desarrollo y seguimiento del proyecto.....	86
9 Conclusiones	88

9.1	Objetivos Cumplidos	88
9.2	Futuras ampliaciones del Proyecto	89
9.3	Conclusiones Personales	89
A Instalación de GesCORE		90
Lista de Código Fuente		92
Referencias		93
Herramientas Utilizadas		94

Lista de Ilustraciones

Figura 1.1 Logotipo Sushitos	2
Figura 2.1: Logotipo de Joomla	6
Figura 2.2: Arquitectura de Joomla	6
Figura 4.1: Primera fase de la organización del proyecto (Gantt)	9
Figura 4.2: Segunda Fase de la organización del proyecto (Gantt)	9
Figura 4.3: Tercera fase de la organización del proyecto (Gantt)	10
Figura 4.4: Cuarta fase de organización del proyecto, desviación (Gantt)	11
Figura 6.1: Diagrama conceptual	19
Figura 6.2: Diagrama de Casos de Uso [1]	26
Figura 6.3: Diagrama de Casos de Uso [2]	27
Figura 7.1: Arquitectura del proyecto	49
Figura 7.2: Arquitectura del web Service	49
Figura 7.3: Flujo básico de la arquitectura	50
Figura 7.4: Integración de GesCORE con Joomla	50
Figura 7.5: Diagrama de secuencia Consultar Entidad de Calidad	51
Figura 7.6: Diagrama de secuencia Nueva Issue	52
Figura 7.7: Diagrama de secuencia Buscar (APIgesCORE)	54
Figura 7.8: Comunicación entre GesCORE y Scopus	55
Figura 7.9: Diagrama de secuencia, operación Importar	56
Figura 7.10: Patrón de diseño de GesCORE	57
Figura 7.11: Diseño del menú de GesCORE	58
Figura 7.12: Mockup - Operación nueva issue, apartado de información general	59
Figura 7.13: Mockup - Operación nueva issue, apartado de tópicos	60
Figura 7.14: Mockup - Operación nueva issue, apartado de entidades de calidad	61
Figura 7.15: Mockup - Operación consultar entidad de calidad	62
Figura 7.16: Mockup - Operación Buscar (APIgesCORE)	63
Figura 7.17: Mockup – Operación importar desde Scopus	64
Figura 7.18: Diseño lógico de la Base de Datos	65

Figura 7.19: Diseño físico de la Base de Datos.....	66
Figura 8.1: Ejemplo de utilización de APIgesCORE	72
Figura 8.2: Retorno de resultados usando JSONP para solucionar problemas de cross domain.....	77
Figura 8.3: Salida del debugger, OK.....	81
Figura 8.4: Salida del debugger, Error Timeout.....	81
Figura 8.5: Captura de GesCORE, consultar tópico.....	82
Figura 8.6: Captura de GesCORE, buscador rápido.....	82
Figura 8.7: Captura de GesCORE, buscador.....	83
Figura 8.8: Captura de GesCORE, consultar entidad de índice de calidad	84
Figura 8.9: Captura de GesCORE, nueva issue.....	84
Figura 8.10: Captura de GesCORE, importar desde Scopus.....	85
Figura 8.11: Desarrollo del proyecto utilizando Scrum	86
Figura 8.12: Captura de un sprint a lo largo del proyecto	87

Lista de Tablas

Tabla 4.1: Estimación de costes de Recursos Humanos.....	12
Tabla 4.2: Estimación de costes de Hardware.....	13
Tabla 4.3: Coste total del Proyecto	14
Tabla 8.1: APIgesCORE, métodos de configuración de la búsqueda	73
Tabla 8.2: APIgesCORE, métodos usados en la función de retorno de resultados	73
Tabla 8.3: APIgesCORE, métodos usados para imprimir	73
Tabla 8.4: Operaciones de filtrado de la clase SearchObj.....	74
Tabla 8.5: Operaciones de filtrado de la clase editionObj.....	75
Tabla 8.6: Operaciones de filtrado de la clase issueObj.....	76
Tabla 8.7: Valores generales por defecto de la API	76
Tabla 8.8: Operaciones del Debugger	79

1

Introducción

1.1 Contexto	2
1.1.1 Organización	2
1.1.2 Motivación Personal.....	2

Hoy en día existen diversas plataformas de bases de datos de congresos y revistas científicas (entidades publicadoras) y también existen algunas plataformas con interfaz web que son útiles a la hora de buscar algún congreso o revista que se haya publicado con anterioridad. Estas plataformas sirven de soporte principalmente a las personas que se dedican a la investigación. Pero la mayoría tienen muchos años y creemos que han quedado un poco desfasadas en funcionalidades y esto se ha convertido en un obstáculo para la mayoría de grupos de investigación.

Otro de los problemas con los que se topa cualquier organización que se dedica a la investigación es como gestionar todos los Call For Papers¹ que le llegan de muchos sitios, muchas veces llega a pasar que los investigadores se olvidan de enviar sus publicaciones porque olvidaron/perdieron el plazo máximo de respuesta.

El Gestor de Congresos y Revistas (GesCORE) nace con la finalidad de ayudar y mejorar la eficiencia de un grupo de investigación que se encuentre en estas tesituras, pero no sólo eso, GesCORE es un proyecto que pretende dar un paso más adelante en la gestión de publicaciones. Lo que pretendemos con este proyecto es **unificar** todo lo que pueda necesitar una organización de investigación en un mismo sitio, su web.

¹ Call For Paper: Método o mecanismo usado en el mundo académico para recoger los artículos del libro o de diario o las presentaciones de la conferencia.

1.1 Contexto

1.1.1 Organización

GesCORE es un proyecto ajustado principalmente para el grupo de investigación Sushitos, la cual lleva a cabo actividades de investigación, desarrollo, innovación, difusión, comunicación y transferencia de tecnología en el ámbito de las TIC inmersas en la sociedad, poniendo especial énfasis en las tecnologías móviles y ubicuas, la interoperabilidad, el software libre y el paradigma de la orientación a servicios y el *cloud computing*².



Figura 1.1 Logotipo Sushitos

Este proyecto se desarrolla con la finalidad no sólo de proporcionar una herramienta de ayuda para los investigadores de *Sushitos* sino también está pensado para servir de conexión o puente entre el *Sushitos* y otras organizaciones o grupos de investigación.

Así pues, *Sushitos* podrá compartir con terceros, la información que crea oportuna sobre los congresos y revistas almacenados en su base de datos.

1.1.2 Motivación Personal

A nivel personal quería hacer este proyecto porque desde siempre he sido un apasionado de las nuevas tecnologías, y particularmente de las tecnologías web. Desde que era pequeño siempre quería entender las entrañas de un sistema web, navegadores, servidores, y el funcionamiento en general de una aplicación web.

Durante la carrera opté por completar el perfil de Ingeniería del Software, y quería desarrollar un proyecto que me motive personalmente y a la vez poder confirmar lo aprendido durante la carrera.

Hace unos meses trabajé desarrollando una aplicación web sobre encuestas, desde luego fue un impulso más para escoger el proyecto GesCORE.

² Cloud Computing: Es un paradigma que permite ofrecer servicios de computación a través de Internet, es decir “en la nube”, donde todo lo que puede ofrecer un sistema informático se ofrece como servicio.

2

Visión

2.1 Descripción del problema	3
2.1.1 P1: Desorganización con los CFP	4
2.1.2 P2: No integración de la información.....	4
2.1.3 P3: Información no adaptable.....	4
2.1.4 P4: Información no dinámica, dificultad de búsqueda	4
2.2 GesCORE como posible solución	5
2.2.1 GesCORE: componente de Joomla	5

2.1 Descripción del problema

Antes de explicar el problema, pasaremos a identificar las necesidades que tiene Sushitos como grupo de investigación.

Sushitos, como cualquier organización, necesita herramientas que le faciliten su labor durante su tarea de investigación. Además de ello, como cualquier organización informática, es un grupo de investigación interesada por la actualidad, así pues, periódicamente les llega información de Entidades Publicadoras (congresos y revistas) que desearían disponer cuando quisiesen. De este modo, en este contexto, podríamos identificar las siguientes necesidades o deseos de la organización:

- **N1:** Tener un lugar donde almacenar la información facilitada en los *Call For Papers* (CFP) que le llegan a los emails, con la finalidad de gestionarlos.
- **N2:** Herramienta que les permita buscar con relativa facilidad los congresos y revistas que se han celebrado anteriormente y la información contenida en sus publicaciones

En el caso de la necesidad “N2” los investigadores de Sushitos podrían recurrir a Sistemas de Bases de Datos especializados como *Scopus*³ para obtener la información que crean oportuna y necesaria.

El primer caso (“N1”) es un poco más complejo ya que no existen muchos sistemas que permita a los investigadores almacenar de forma organizada y óptima la información que les llega a través de los Call for Paper.

Finalmente, mediante sus necesidades podríamos identificar sus problemas en este contexto, los cuales son:

2.1.1 P1: Desorganización con los CFP

Muchos de los Call For Papers(CFP) llegan al equipo de investigación a través del correo electrónico. Obviamente no es recomendable tener guardado esta información en el email por los problemas que ello conlleva, además, es muy complicado buscar datos de un CFP (como las fechas de recepción y fechas de inicio de una edición de un congreso) en una bandeja de email.

2.1.2 P2: No integración de la información

No tener un sistema centralizado y organizado donde guardar la información de las publicaciones puede llevar a una incorrecta sincronización entre los miembros del grupo, de tal forma que no sepan donde y cuando se celebra algún congreso del cual estaban interesados.

2.1.3 P3: Información no adaptable

Para el grupo de investigación *Sushitos* era importante, entre otras cosas, que las publicaciones almacenadas en su sistema pudieran reflejar las valoraciones de las Entidades de Índice de Calidad⁴. La mayoría de sistemas (como Scopus) que se encuentran en la actualidad no permiten modificar las publicaciones que tienen en su sistema debido a que ese no es su propósito.

2.1.4 P4: Información no dinámica, dificultad de búsqueda

Sabemos que la cantidad de publicaciones que hay en cualquier Base de Datos es muy extensa. Eso hace que se derive en una cierta dificultad a la hora de buscar información de las publicaciones.

³ Scopus: Es una base de datos bibliográfica de resúmenes y citas de artículos de revistas de carácter científico

⁴ Entidad de Índice de Calidad: Entidades que se encargan de valorar las publicaciones hechas por revistas y ediciones de los congresos.

2.2 GesCORE como posible solución

GesCORE es un proyecto, que consiste en crear una aplicación web, la cual se desarrollará como un componente principalmente de Backend creado sobre el CMS⁵ Joomla, este componente se instalará en el sitio web de la organización *Sushitos*.

Este proyecto se desarrolla con la finalidad de gestionar los Congresos y Revistas tratando de satisfacer las necesidades de *Sushitos* y corregir así sus problemas en el contexto descrito.

Esta aplicación permitirá a los usuarios tener una base activa de información sobre las entidades publicadoras (congresos y revistas científicas), interactuar con esta información permitiendo crear, modificar y eliminar elementos con la finalidad de satisfacer “N1” y poder corregir “P1”, “P2” y “P3”.

Y dado que la base activa de información es bastante amplia, GesCORE está pensado con el objetivo de interconectar, en la mayor medida posible, toda la información para que sea mucho más sencillo a la hora de buscar publicaciones en ella, esto tratará de satisfacer “N2” para corregir “P4”.

Adicionalmente, era deseable por la organización poder importar desde otros servicios la información de entidades publicadoras y sus publicaciones.

Finalmente, como ya hemos mencionado este gestor tendrá una API⁶ destinada a terceras personas (o miembros del equipo) que les permitirá conectarse a la información almacenada en GesCORE. Cabe resaltar, que GesCORE es una aplicación web de backend para la organización, es por ello que la API será de gran ayuda para los usuarios que no puedan acceder al backend de la organización.

2.2.1 GesCORE: componente de Joomla

Como ya hemos mencionado, GesCORE es un componente que se integrará con Joomla, el cual es un Sistema de Gestión de Contenidos (CMS en sus siglas en inglés).

Dicho esto, antes de nada explicaremos que es Joomla y cuáles son sus características como CMS.

⁵ CMS: Content Management System, en español: Sistema de Gestión de Contenidos

⁶ API: (del inglés Application Programming Interface) es el conjunto de funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción

Joomla

Joomla es un Sistema de Gestión de Contenido de código abierto, utilizado para la publicación de contenidos web dinámicos e interactivos. Este CMS está desarrollado en PHP y liberado bajo licencia GPL⁷. Este administrador de contenidos puede utilizarse en una PC local (en Localhost), en una Intranet o a través de Internet y requiere para su funcionamiento una base de datos creada con un gestor de bases de datos (MySQL es lo más habitual), así como de un servidor HTTP Apache.



Figura 2.1: Logotipo de Joomla

▪ ¿Qué es un sistema gestor de contenidos?

Un sistema gestor de contenidos es un software que mantiene la trazabilidad o seguimiento de cada pieza de contenido en un sitio web, muy parecido a una biblioteca pública que almacena libros y mantiene el seguimiento de los mismos. El contenido puede ser texto, fotos, música, video, documentos varios, o cualquier cosa similar que quiera administrar.

▪ Características

Sus características principales son la generación de código HTML bien formado, gestión de blogs, vistas de impresión de artículos, flash con noticias, foros, polls (encuestas), calendarios, búsquedas integradas al sitio y soporte multi-idioma.

▪ Arquitectura

Está construido sobre el patrón de diseño MVC (Modelo Vista Controlador).

▪ ¿Por qué Joomla?

Joomla ha sido descargado más de 50 millones de veces y actualmente 30 millones de webs funcionan bajo Joomla, además existe una enorme comunidad de desarrolladores y diseñadores que trabajan con y para Joomla de forma libre, esto hace que Joomla sea un sistema **fiable** para trabajar, fácil de **mantener** y fácil de **actualizar**.

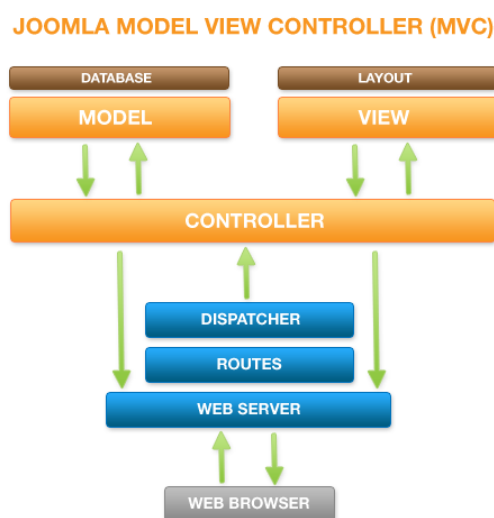


Figura 2.2: Arquitectura de Joomla

⁷ GNU: General Public License, es una licencia pública que garantiza a los usuarios finales (personas, organizaciones, compañías) la libertad de usar, estudiar, compartir (copiar) y modificar el software.

3

Objetivos

El objetivo fundamental de GesCORE es mejorar la eficiencia de la organización *Sushitos* en la gestión de publicaciones con la finalidad de tener una mejor organización en sus labores de investigación.

El proyecto pretende cubrir este fundamental objetivo basándose en cubrir los siguientes:

- Informatización del sistema de gestión de Call For Papers (CFP) de la organización de investigación *Sushitos*.
- Integración: Uno de los objetivos que teníamos con el proyecto, era que tenía que poderse integrar con otros sistemas de manera que podamos aprovechar sus recursos.

Creemos que la suma de estos objetivos servirá para mejorar particularmente la eficiencia de los investigadores de *Sushitos*, y en lo general a las personas interesadas en tener una base sólida de publicaciones en un lugar centralizado, robusto, escalable y de fácil manejo.

Adicionalmente, como objetivo secundario, era deseable por la organización disponer de una interfaz en el frontend de la aplicación por donde cualquier persona que consulte su web pueda ver la base de datos de publicaciones que tienen disponibles. Esto se convirtió en uno de los objetivos que queríamos cubrir con GesCORE.

4

Planificación del Proyecto

4.1 Organización Temporal del Proyecto	8
4.1.1 Primera Fase: Preparación del entorno.....	9
4.1.2 Segunda Fase: Desarrollo (1ra parte)	9
4.1.3 Tercera Fase: Desarrollo (2ra parte).....	10
4.2 Seguimiento	11
4.2.1 Desviaciones y ampliación.....	11
4.3 Estimación de Costes	12
4.3.1 Coste de Recursos Humanos	12
4.3.2 Costes de Hardware.....	13
4.3.3 Costes de Software	13
4.3.4 Gastos Generales	14
4.3.5 Total	14

A continuación explicaremos la planificación inicial que se tuvo con el proyecto, el timing planificado y las desviaciones que se produjeron.

Así también, se estimará el coste de la realización a lo largo del proyecto.

4.1 Organización Temporal del Proyecto

Basándonos del tiempo que disponíamos, la organización temporal inicial del proyecto se dividió inicialmente en 3 fases.

Es importante destacar que existen funcionalidades que no se contemplaron inicialmente y se ampliaron a posteriori, esa es la razón de su ausencia en la organización inicial del proyecto.

4.1.1 Primera Fase: Preparación del entorno

En esta primera fase de organización del proyecto se llevaron a cabo tareas externas a los objetivos iniciales del proyecto debido a la poca familiarización del desarrollador en un entorno de Joomla.

La duración de la primera fase fue de 7 días laborables, 56 horas.

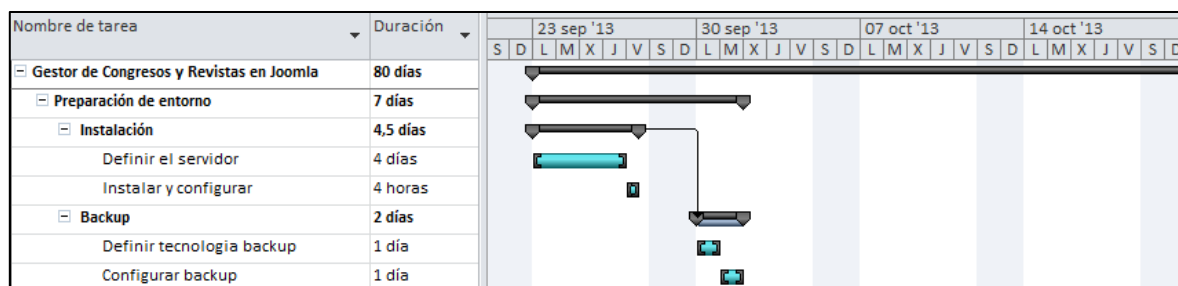


Figura 4.1: Primera fase de la organización del proyecto (Gantt)

4.1.2 Segunda Fase: Desarrollo (1ra parte)

La fase de desarrollo se dividió en 2 partes, la primera parte consistió en la especificación y diseño de la fase de desarrollo de GesCORE.

Durante la fase de especificación hubo reuniones con el equipo de *Sushitos* con la finalidad de determinar con más precisión el desarrollo del proyecto.

En la fase de Diseño e Implementación, a nivel físico, se diseñó e implementó la capa de datos del sistema.

La duración de la segunda fase se estimó en 24 días, es decir 192 horas.

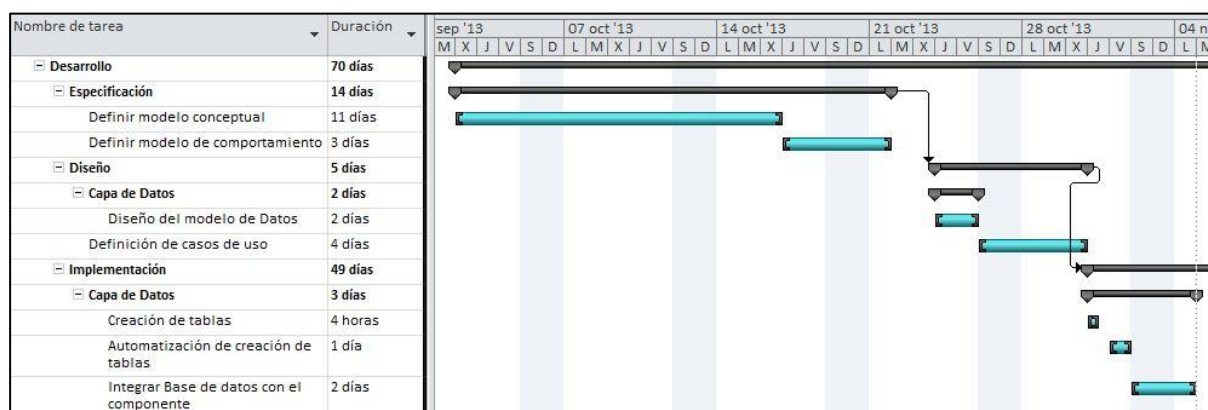


Figura 4.2: Segunda Fase de la organización del proyecto (Gantt)

4.1.3 Tercera Fase: Desarrollo (2ra parte)

Una vez definido los casos de uso en la anterior fase, esta fase consiste en la implementación de los mismos.

Durante esta fase de desarrollo se llevaron a cabo Tests de funcionalidades e integración.

La duración de esta fase es de 49 días laborables, es decir 392 horas.

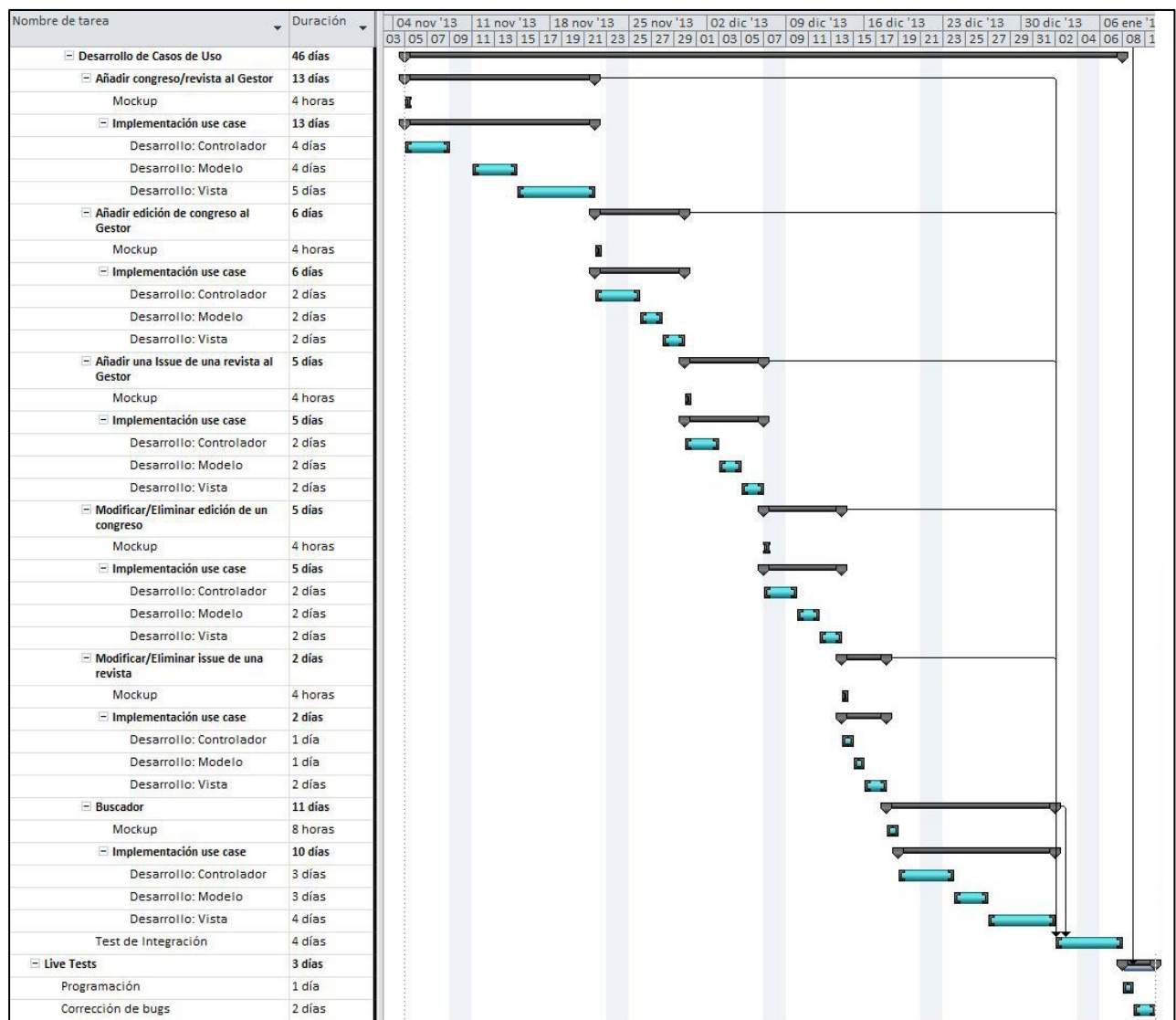


Figura 4.3: Tercera fase de la organización del proyecto (Gantt)

4.2 Seguimiento

4.2.1 Desviaciones y ampliación

Se produjeron desviaciones de la planificación inicial del proyecto debido a que se empezó la etapa de desarrollo con 2 semanas de retraso.

Esto hizo que el proyecto tal como se planificó no acabara en el tiempo previsto, impidiendo así realizar una documentación del proyecto en óptimas condiciones.

Debido a que el proyecto se ubica en un contexto de Proyecto Final de Carrera, decidimos postergar la defensa del proyecto para cuando este todo listo.

De este modo, se decide hacer una ampliación del proyecto e incluir la funcionalidad de “Integración con otras APIs” (importar) que inicialmente estaba previsto como “opcional”, y con la finalidad de que GesCORE pueda integrarse en plataformas de terceros, se decide crear una API pública para desarrolladores.

A continuación mostramos el Diagrama Gantt del trabajo desarrollado con las nuevas funcionalidades añadidas y descritas anteriormente al que llamaremos 4ta Fase – 3ra parte del Desarrollo.

La duración de esta fase fue de 47 días, es decir 376 horas.

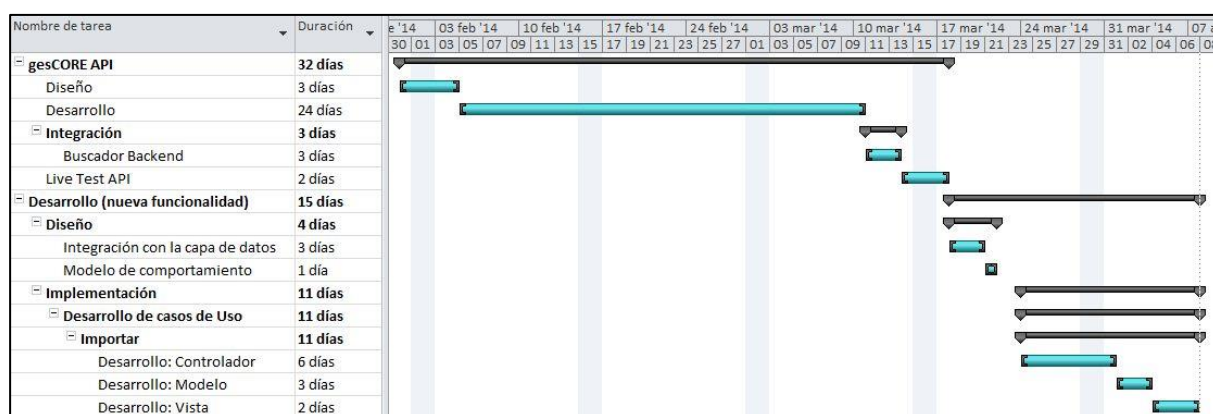


Figura 4.4: Cuarta fase de organización del proyecto, desviación (Gantt)

4.3 Estimación de Costes

Dividiremos el coste de realización del proyecto en 4 tipos:

- Coste de Recursos Humanos
- Costes de Hardware
- Costes de Software
- Gastos Generales

4.3.1 Coste de Recursos Humanos

Para calcular el coste total dedicado a Recursos Humanos debemos tener en cuenta las personas involucradas en el proyecto, el número de horas dedicadas y el precio por hora de las personas involucradas.

En primer lugar tenemos que el proyecto ha necesitado de un diseñador y un desarrollador, que en este caso ha sido la misma persona.

En cuanto al precio/hora del desarrollador-diseñador, teniendo en cuenta los estándares que hay en la actualidad se ha fijado en 8€/h.

A partir de la información proporcionada en la “Organización Temporal del Proyecto” podemos obtener el número de horas totales dedicadas al proyecto, el cual se resume en la siguiente tabla.

Fase	Duración	precio/hora	Coste
Preparación del entorno	56 h.	8 €	448 €
Desarrollo (1ra parte)	192 h.	8 €	1536 €
Desarrollo (2ra parte)	392 h.	8 €	3136 €
Desarrollo (3ra parte)	376 h.	8 €	3008 €
Total	1016 h		8128 €

Tabla 4.1: Estimación de costes de Recursos Humanos

4.3.2 Costes de Hardware

Para el desarrollo del proyecto, se ha utilizado un portátil de la marca Toshiba con Windows7 preinstalado con las siguientes características: Procesador Intel Core i3, 2.4Ghz y 4 GB de RAM.

El coste del ordenador es de: 629€ (sin IVA).

Además del portátil, se ha utilizado un servidor de pruebas, el cual no tiene ningún coste.

En caso del portátil, según datos de la PGC⁸, se estima una vida útil de 4 años, por tanto el coste real del proyecto sería el equivalente a 9 meses dentro de su vida útil.

Descripción	Coste	Vida útil	Tiempo de Uso	%imputable	Coste Real
Portátil con Windows7 preinstalado	629 €	48 meses	9 meses	18,75%	117,93 €
Servidor	0 €	-	-	-	0 €
Total					117,93 €

Tabla 4.2: Estimación de costes de Hardware

4.3.3 Costes de Software

Para la realización de este proyecto se ha utilizado el entorno de programación Netbeans⁹, y la mayor parte del código se hizo sobre una distribución Linux.

En cuanto a la documentación las aplicaciones utilizadas fueron: Office 2010 para el documento, y websequencediagrams.com y draw.io para los diagramas de secuencia y casos de uso.

En cuanto al coste, en el caso del Office 2010 se utilizó una licencia proporcionada por la universidad, de este modo su coste es de 0€, draw.io es gratuito y en cuanto a websequencediagrams.com se utilizó la versión básica gratuita.

Por tanto, dentro de esta sección el coste total es de cero euros.

⁸ PGC: Acrónimo del Plan General Contable, texto legal que regula la contabilidad de las empresas en España, se encarga de la normalización de la contabilidad y los principios contables.

⁹ Netbeans: Es un entorno de desarrollo integrado libre.

4.3.4 Gastos Generales

Los costes en esta sección tienen en cuenta el coste de alquiler de la oficina y sus facturas (agua, luz, calefacción, etc.). Dado que el proyecto en su mayoría fue desarrollado en la biblioteca de la universidad los costes de este apartado son de cero euros.

4.3.5 Total

De este modo tenemos que el coste total del proyecto es la suma del coste de los recursos humanos y el inmovilizado material¹⁰.

Por tanto el coste total es:

Tipo	Coste
Coste Recursos Humanos	8128,00 €
Costes de Hardware	117, 93 €
Costes de Software	0 €
Servicios y Estructura	0 €
Total	8245,93 €
Total (impuesto: IVA = 21%)	9977,58 €

Tabla 4.3: Coste total del Proyecto

¹⁰ Inmovilizado material: Se trata de bienes que se utilizan en la actividad permanente y productiva de la empresa

5

Requerimientos

5.1 Requisitos Funcionales	15
5.1.1 Gestor de Call For Papers (CFP)	15
5.1.2 Alta interconectividad entre publicaciones	16
5.1.3 Buscador de publicaciones	16
5.1.4 Integración con otras plataformas	16
5.1.5 API para desarrolladores	16
5.2 Requisitos No Funcionales	17
5.2.1 Extensibilidad.....	17
5.2.2 Escalabilidad	17
5.2.3 Usabilidad.....	17
5.2.4 Portabilidad	17

5.1 Requisitos Funcionales

5.1.1 Gestor de Call For Papers (CFP)

Uno de los requisitos más importantes que tiene que cubrir GesCORE es que tiene que ser capaz de permitir a los usuarios gestionar libremente los Call For Papers que le llegan por terceras vías. Este requisito funcional implica:

- Gestión de congresos
- Gestión de ediciones
- Gestión de revistas, issues
- Gestión de Entidades de índice de calidad y Gestión de tópicos de las publicaciones.

5.1.2 Alta interconectividad entre publicaciones

Este es un requisito diferenciador entre otras plataformas de gestión de publicaciones. Consiste en que, dado que la información activa es muy amplia, los elementos y publicaciones que se encuentren en GesCORE deben estar interconectados entre sí, de manera que a partir de un elemento cualquiera (“x”), podamos acceder a los elementos que tengan algo en común con “x”.

Un ejemplo claro son los tópicos de las publicaciones, a partir de un tópico se debería navegar fácilmente a los elementos comunes del tópico y sus tópicos similares.

5.1.3 Buscador de publicaciones

Dado que el volumen de datos que manejará GesCORE es muy amplio, es importante que el buscador sea una herramienta potente y robusta que permita encontrar en el menor tiempo posible la publicación o elemento que está buscando el usuario.

5.1.4 Integración con otras plataformas

Uno de los requisitos importantes era que GesCORE debería permitir importar publicaciones desde terceros. Finalmente se decidió optar por Scopus como plataforma para importar ya que cuenta con miles de publicaciones en su base de datos.

5.1.5 API para desarrolladores

Uno de los requisitos que tenía que cubrir GesCORE es que la información almacenada en el backend de la web de la organización debería poder ser accesible desde el frontend.

Dado este requisito, generalizamos la idea, y se decidió crear una API para desarrolladores, con el fin de cumplir este requerimiento y el de las personas interesadas en las publicaciones almacenadas en GesCORE.

5.2 Requisitos No Funcionales

5.2.1 Extensibilidad

GesCORE se diseñó e implementó con la finalidad de poder crecer tanto como se quiera. La capacidad de añadir nuevas funcionalidades al gestor es relativamente fácil. La API creada para desarrolladores abre un mundo de posibilidad de extensión del proyecto, permitiendo entre otras cosas crear aplicaciones para dispositivos móviles.

5.2.2 Escalabilidad

Este es un requisito no funcional muy importante ya que el volumen de datos que manejará GesCORE es muy amplio, de esta forma la carga de trabajo que se manejará es muy grande y con facilidad a ser aún mayor. Es por ello que el sistema debe prever estos cambios de escala a nivel físico (BBDD) y en la interfaz.

5.2.3 Usabilidad

La Usabilidad tenía es un requisito importante ya que la interfaz de la aplicación debería ser lo más clara posible. Con la finalidad de cubrir este requisito, se diseñó GesCORE de manera que debería cumplir lo siguiente:

- Velocidad alta de operación
- Tasa baja de errores.
- Facilidad de aprendizaje
- Facilidad de retención

5.2.4 Portabilidad

Dado que este proyecto se sustenta sobre una plataforma ya existente (Joomla), era importante cumplir este requisito de manera que los dos sistemas puedan coexistir sin problemas y permitir a los usuarios una fácil instalación del componente en su web,

GesCORE también da facilidades a la hora de portar el componente hacía otro sistema.

6

Especificación

6.1 Modelo Conceptual	19
6.1.1 Clases	20
6.1.2 Restricciones de integridad	25
6.2 Casos de uso	26
6.2.1 Nueva Revista.....	28
6.2.2 Buscar.....	29
6.2.3 Consultar Revista	30
6.2.4 Editar Revista	31
6.2.5 Eliminar Revista	32
6.2.6 Nueva Issue	33
6.2.7 Editar Issue	34
6.2.8 Eliminar Issue.....	35
6.2.9 Importar Publicaciones.....	36
6.2.10Búsqueda Avanzada	37
6.2.11Eliminar Editorial	38
6.2.12Nueva Editorial.....	39
6.2.13Consultar Entidad de Calidad.....	40
6.2.14Nueva Entidad de Calidad	41
6.2.15Editar Entidad de Calidad.....	42
6.2.16Eliminar Entidad de Calidad	43
6.2.17Consultar Tópicos.....	44
6.2.18Nuevo Tópico	45
6.2.19Editar Tópico.....	46
6.2.20Eliminar Tópico.....	47

6.1.1 Clases

Editorial

Clase	Editorial
Descripción	Representación de la editorial encargada de la organización del congreso o la publicación de las revistas
Atributos	<p><i>-<u>nombre</u></i>: Nombre de la editorial</p> <p><i>-<u>siglas</u></i>: Siglas de la editorial</p> <p><i>-<u>link</u></i>: Dirección web de la editorial</p>
Operaciones	-

Entidad Publicadora

Clase	EntidadPublicadora
Descripción	Clase que representa la entidad (revista o congreso) la cual usaremos para gestionar su información.
Atributos	<p><i>-<u>nombre</u></i>: Nombre de la entidad (congreso o revista)</p> <p><i>-<u>acrónimo</u></i>: Acrónimo del congreso o revista.</p> <p><i>-<u>link</u></i>: Dirección web de la entidad publicadora al congreso o revista en general.</p>
Operaciones	-

Congreso

Clase	Congreso
Descripción	Representa a la reunión o conferencia, que se conforma por ediciones periódicas.
Atributos	- <i>periodicidad</i> : Indica cada cuanto tiempo se realiza el congreso
Operaciones	-

Edición

Clase	Edicion
Descripción	Representa a una de las ediciones realizadas de un determinado congreso. El congreso organiza ediciones periódicas.
Atributos	<p>- <i>orden</i>: Orden de la edición dentro del congreso.</p> <p>- <i>lugar</i>: Lugar donde es llevado a cabo la edición del congreso</p> <p>- <i>link</i>: Enlace web de la edición</p> <p>- <i>limRecepción</i>: Fecha hasta donde el congreso en esa edición permite la entrega de trabajos.</p> <p>- <i>fechaDefinitiva</i>: Fecha hasta donde el congreso en esa edición, permite entregar un trabajo después de haber dado una respuesta sobre un trabajo recibido.</p> <p>- <i>fechaInicio</i>: Fecha en la que inicia la edición del congreso</p> <p>- <i>fechaFin</i>: Fecha de finalización de la edición del congreso</p>
Operaciones	-

Tópico

Clase	Topico
Descripción	Representación del tema o temas que trata el congreso/revista y sus publicaciones
Atributos	<u>-nombre</u> : Nombre del tema <u>-descripcion</u> : Descripción del tópico
Operaciones	-

Revista

Clase	Revista
Descripción	Clase que representa el conjunto de ediciones generalmente periódicas.
Atributos	<u>-issn</u> : (International Standard Serial Number, Número Internacional Normalizado de Publicaciones Seriadas) Identificador de la revista. <u>-periodicidad</u> : Indica cada cuanto tiempo se hará una nueva publicación de la revista
Operaciones	-

Issue

Clase	Issue
Descripción	Representación de una edición de la revista
Atributos	<u>-volumen</u> : Volumen de la edición <u>-numero</u> : Número de la edición <u>-limRecepción</u> : Fecha hasta donde la revista permite la entrega de trabajos. <u>-fechaDefinitiva</u> : Fecha hasta donde la revista permite entregar un trabajo después de haber dado una respuesta sobre un trabajo recibido. <u>-fechaPublicación</u> : Fecha en la que se publica la revista
Operaciones	-

Especial Issue

Clase	EspecialIssue
Descripción	Representación de una edición especial de una revista no prevista inicialmente.
Atributos	- <u>temaPrincipal</u> : Tema principal de la edición especial.
Operaciones	-

Entidad de Índice de calidad

Clase	EntidadIndiceCalidad
Descripción	Entidad encargada de evaluar las ediciones de los congresos que se realicen, así también como de las ediciones de revistas que se publiquen.
Atributos	<p>-<u>nombre</u>: Nombre de la entidad evaluadora</p> <p>-<u>siglas</u>: Siglas de la entidad evaluadora</p> <p>-<u>link</u>: Dirección web donde se puede consultar la entidad evaluadora.</p> <p>-<u>categorias</u>: Conjunto de categorías de valoración de la entidad evaluadora tenidas en cuenta para efectuar su veredicto.</p> <p>-<u>criterios</u>: Descripción de los criterios utilizados para efectuar su valoración de la edición de una revista o congreso.</p> <p>-<u>tipoCategorias</u>: Indica si la entidad de calidad valora las publicaciones numéricamente o mediante una lista de valoraciones.</p>
Operaciones	-

Valoración de Edición

Clase	ValoracionEdicion
Descripción	Clase que representa el índice de calidad otorgado por una entidad evaluadora hacia la edición de un congreso.
Atributos	- <u>indice</u> : Valor otorgado por la entidad evaluadora, el cual es un valor dentro de los especificados por la entidad
Operaciones	-

Valoración de una Issue

Clase	ValoracionIssue
Descripción	Clase que representa el índice de calidad otorgado por una entidad evaluadora hacia la edición de una revista.
Atributos	- <u>indice</u> : Valor otorgado por la entidad evaluadora, el cual es un valor dentro de los especificados por la entidad
Operaciones	-

6.1.2 Restricciones de integridad

Restricciones de clave primaria

- No existen dos Editoriales diferentes con el mismo nombre.
- No existen dos congresos diferentes que tengan el mismo nombre.
- No existen dos revistas diferentes con el mismo issn.
- No existen dos ediciones de un mismo congreso con el mismo orden.
- No existen dos Issues de la misma revista con el mismo volumen y número.
- No existen dos entidades de índice de calidad que tengan el mismo nombre.
- No existen dos tópicos diferentes con el mismo nombre.
- Una misma Entidad de Índice de Calidad no puede tener dos valoraciones diferentes de una misma edición de un congreso
- Una misma Entidad de Índice de Calidad no puede tener dos valoraciones diferentes de una misma Issue de una revista

Restricciones de integridad de las asociaciones

- El índice del valor asignado por la Entidad de Índice de Calidad a una edición de un congreso pertenece a una de las categorías dadas por la entidad.
- El índice del valor asignado por la Entidad de Índice de Calidad a una edición de una revista pertenece a una de las categorías dadas por la entidad.

Información derivada

- *Edicion.orden*: Identificador correlativo que identifica a una edición dentro de un mismo congreso.

6.2 Casos de uso

A continuación mostramos los diagramas de casos de uso del proyecto. Cabe resaltar que el caso de uso “Gestión de Congresos” es análogo a la Gestión de Revistas, es por ello que decidimos poner sólo uno de ellos, el siguiente gráfico muestra el diagrama de casos de uso de “Gestión de Revistas”.



Figura 6.2: Diagrama de Casos de Uso [1]

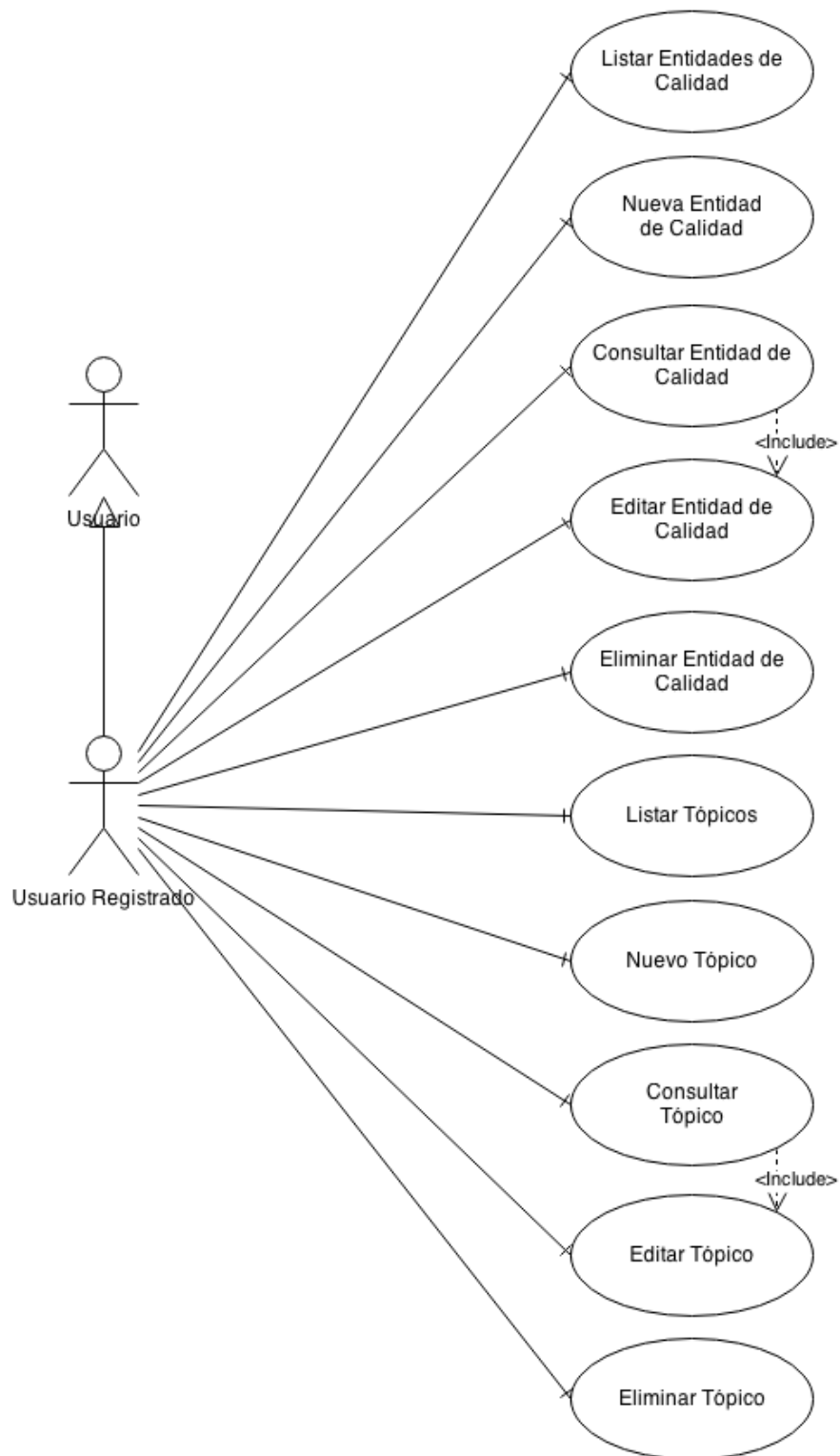


Figura 6.3: Diagrama de Casos de Uso [2]

6.2.1 Nueva Revista

Caso de uso

Nueva Revista

Actor: Usuario Registrado

Precondición: -Usuario tiene acceso al backend de la web.

Trigger: -

Escenario Principal

1. El usuario indica que quiere dar de alta una nueva revista e introduce los datos de la revista y selecciona su editorial
2. El sistema valida los datos introducidos por el usuario
3. El sistema crea la nueva revista.
4. El sistema confirma redireccionando al usuario a la ficha de la nueva revista.

Extensiones

[1-2].a. El usuario cancela el alta

[1-2].a.1 Acaba el caso de uso

1.a. La editorial de la revista no existe en el sistema.

1.a.1 El usuario indica que quiere crear una nueva Editorial

1.a.2 Se inicia el caso de uso "Nueva Editorial"

2.a Falta algún campo obligatorio

2.a.1 El sistema comunica el error, el caso de uso continúa en el punto 1 manteniendo la información ya introducida

2.b. Ya existe la revista en el sistema

2.b.1 El sistema comunica el error, el caso de uso continúa en el punto 1 manteniendo la información ya introducida

Satisfacción: 2

No Satisfacción: 4

Dependencias: -

Includes:

Extends: Nueva Issue

6.2.2 Buscar

Caso de uso

Buscar

Actor: Usuario Registrado

Precondición: -

Trigger: -

Escenario Principal

1. El usuario introduce el criterio de búsqueda
2. El sistema valida los datos introducidos por el usuario
3. El sistema crea un “Request” a la API
4. El sistema muestra el resultado de la búsqueda ofrecidos por el “Response” de la API

Extensiones

- 1.a. El usuario es un usuario Registrado que se encuentra en Backend que decide filtrar por Ediciones.
 - 1.a.1 El usuario registrado indica hacer un filtrado por Ediciones
 - 1.a.2 Se inicia el caso de uso “Búsqueda avanzada”
- 1.b. El usuario es un usuario registrado que se encuentra en Backend que quiere buscar en los tópicos de las publicaciones.
 - 1.b.1 El usuario registrado indica que quiere buscar en los tópicos de las publicaciones
 - 1.b.2 Se inicia el caso de uso “Búsqueda avanzada”

Satisfacción: 4

No Satisfacción: 4

Dependencias

Includes: -

Extends: -

6.2.3 Consultar Revista

Caso de uso

Consultar Revista

Actor: Usuario Registrado, sistema

Precondición:

-El usuario es un usuario dado de alta en el sistema

Trigger: -

Escenario Principal

1. A partir de la lista de revistas el Usuario Registrado selecciona la Revista que desea consultar.
2. El sistema obtiene los datos básicos de la Revista
3. El sistema obtiene los tópicos de la Revista
4. El sistema obtiene las valoraciones de la Revista
5. El sistema muestra la información de la Revista seleccionada al usuario

Extensiones

5.a. El usuario decide editar la información mostrada por la Revista

5.a.1 Se inicia el caso de uso "Editar Revista"

Satisfacción: 3

No Satisfacción: 3

Dependencias: -

Includes: - Editar Revista

Extends: -

6.2.4 Editar Revista

Caso de uso

Editar Revista

Actor: Usuario Registrado

Precondición: -Usuario tiene acceso al backend de la web.

Trigger: -

Escenario Principal

1. El usuario indica al sistema cual es el campo o atributo que quiere editar.
2. El sistema responde permitiendo al usuario editar el atributo seleccionado
3. El usuario introduce el nuevo valor del atributo
4. El sistema valida el nuevo valor
5. El sistema confirma el cambio mostrando al usuario el nuevo valor del atributo.

Extensiones

- 1.a. El usuario indica que quiere añadir un nuevo tópico a la Revista
 - 1.a.1 El sistema muestra la lista de tópicos disponibles en el sistema que no se encuentran ya asociados a la revista.
 - 1.a.2 El usuario selecciona los tópicos que desea
 - 1.a.3 El sistema asocia los tópicos seleccionados a la revista y acaba el caso de uso
- 1.b. El usuario indica que quiere eliminar un nuevo tópico a la Revista
 - 1.b.1 El sistema pide una confirmación al usuario y acaba el caso de uso
- 1.c El usuario indica que quiere ver los tópicos similares de uno en concreto.
 - 1.c.1 El sistema muestra la lista de tópicos disponibles en el sistema, que son similares y que no se encuentran ya asociados a la revista.
 - 1.c.2 El usuario selecciona los tópicos que desea
 - 1.c.3 El sistema asocia los tópicos seleccionados a la revista y acaba el caso de uso

Satisfacción: 3

No Satisfacción: 3

Dependencias:

Includes: -

Extends: -

6.2.5 Eliminar Revista

Caso de uso

Eliminar Revista

Actor: Usuario Registrado

Precondición: -Usuario tiene acceso al backend de la web.

Trigger: -

Escenario Principal

1. El usuario lista las revistas almacenadas en el sistema y selecciona las que quiere dar de baja.
2. El sistema solicita una confirmación
3. El usuario confirma que quiere eliminar las revistas seleccionadas
4. El sistema da de baja las revistas seleccionadas.
5. El sistema da de baja las issues asociadas a la revista.
6. El sistema confirma mostrando la nueva lista de revistas disponibles.

Extensiones

- 3.a. El usuario cancela la petición de eliminación y termina el caso de uso.
- 4.a. El sistema detecta que la edición de la revista no tiene ninguna otra entidad publicadora
 - 4.a.1 El sistema inicia el caso de uso "Eliminar Editorial"
 - 4.a.2 El caso de uso continua en el punto 2.

Satisfacción: 2

No Satisfacción: 3

Dependencias: Listar Revistas

Includes: -

Extends: -

6.2.6 Nueva Issue

Caso de uso

Nueva Revista

Actor: Usuario Registrado

Precondición: -Usuario tiene acceso al backend de la web.

Trigger: -

Escenario Principal

1. El usuario indica que quiere dar de alta una nueva Issue
2. El sistema carga los tópicos de la revista a la que pertenece la Issue y le permite añadir sus valoraciones.
3. El usuario introduce los datos de la Issue, selecciona sus tópicos, revista, y añade valoraciones.
4. El sistema valida los datos introducidos por el usuario
5. El sistema crea una nueva Issue asociada a una revista añadida o existente.
6. El sistema confirma redireccionando al usuario a la ficha de la nueva Issue.

Extensiones

- [1-3].a. El usuario cancela el alta
- [1-2].a.1 Acaba el caso de uso
- 2.a. La revista a la que pertenece la nueva Issue no se encuentra en el sistema.
- 1.a.1 El usuario indica que dará de alta una nueva Revista
- 1.a.2 Se inicia el caso de uso “Nueva Revista”
- 3.a. Los tópicos que se desean añadir no existen en el sistema
- 1.a.1 El usuario indica que dará de alta nuevos tópicos.
- 1.a.2 El sistema se prepara para recibir nuevos tópicos.
- 4.a. Falta algún campo obligatorio
- 4.a.1 El sistema comunica el error, el caso de uso continúa en el punto 3 manteniendo la información ya introducida

Satisfacción: 3

No Satisfacción: 4

Dependencias

Includes: -

Extends: -

6.2.7 Editar Issue

Caso de uso

Editar Issue

Actor: Usuario Registrado

Precondición: -Usuario tiene acceso al backend de la web.

Trigger: -

Escenario Principal

1. En la vista de la Revista el usuario selecciona la Issue que desea editar.
2. El sistema permite al usuario editar los campos que desee, carga los tópicos de la revista a la que pertenece la Issue y le permite modificar sus valoraciones.
3. El usuario modifica los datos que crea necesario.
4. El sistema valida los datos introducidos por el usuario
5. El sistema confirma redireccionando al usuario al sitio de donde llegó.

Extensiones

[2-3].a. El usuario cancela la edición

[2-3].a.1 Acaba el caso de uso

2.a. La revista a la que pertenece la Issue no se encuentra en el sistema.

1.a.1 El usuario indica que dará de alta una nueva Revista

1.a.2 Se inicia el caso de uso "Nueva Revista"

3.a. Los tópicos que se desean añadir no existen en el sistema

1.a.1 El usuario indica que dará de alta nuevos tópicos.

1.a.2 El sistema se prepara para recibir nuevos tópicos.

4.a. Falta algún campo obligatorio

4.a.1 El sistema comunica el error, el caso de uso continúa en el punto 3 manteniendo la información ya introducida

Satisfacción: 2

No Satisfacción: 4

Dependencias

Includes: -

Extends: -

6.2.8 Eliminar Issue

Caso de uso

Eliminar Issue

Actor: Usuario Registrado

Precondición:

-Usuario tiene acceso al backend de la web.

Trigger: -

Escenario Principal

1. En la vista de la Revista el usuario selecciona la Issue que desea eliminar.
2. El sistema solicita una confirmación
3. El usuario confirma que quiere eliminar las Issues seleccionadas
4. El sistema da de baja las Issues seleccionadas
5. El sistema confirma mostrando la nueva lista de Issues disponibles de la revista.

Extensiones

- 3.a. El usuario cancela la petición de eliminación y termina el caso de uso.

Satisfacción: 3

No Satisfacción: 3

Dependencias:

Includes: -

Extends: -

6.2.9 Importar Publicaciones

Caso de uso

Importar Publicaciones

Actor: Usuario Registrado

Precondición: -Usuario tiene acceso al backend de la web.

Trigger: -

Escenario Principal

1. El usuario desea importar publicaciones desde Scopus e introduce los criterios de búsqueda.
2. El sistema conecta con la API de Scopus y crea una conexión para el usuario y la mantiene.
3. El sistema obtiene los resultados y lista los recursos para el usuario.
4. El usuario selecciona las publicaciones que desea importar
5. El sistema pide una confirmación indicando el número de publicaciones a importar.
6. El usuario confirma las publicaciones a importar.
7. El sistema da de alta la nueva publicación asociándola a su correspondiente Revista.
8. El sistema informa que las publicaciones se importaron correctamente al usuario y muestra un link de las publicaciones importadas.

Extensiones

- 6.a. El usuario cancela la importación.
 - 6.a.1 El caso de uso vuelve al punto 4.
- 7.a. La Revista de la nueva publicación no se encuentra en el sistema
 - 7.a.1 El sistema da de alta la revista
 - 7.a.2 El sistema da de alta la publicación y continúa el caso de uso en el punto 8.
- 8.a. Existen publicaciones que no se han importado correctamente
 - 8.a.1 El sistema informa las publicaciones que no se han podido importar y cuál es la razón.

Satisfacción: 4

No Satisfacción: 4

Dependencias:

Includes: - Nueva Revista, Nueva Issue

Extends: -

6.2.10 Búsqueda Avanzada

Caso de uso

Búsqueda Avanzada

Actor: Usuario Registrado

Precondición:

-Usuario tiene acceso al backend de la web.

Trigger: -

Escenario Principal

1. El usuario solicita hacer una búsqueda filtrando por ediciones o issues.
2. El sistema permite al usuario filtrar por sus principales campos.
3. El usuario introduce los datos
4. El sistema valida los datos introducidos por el usuario
5. El sistema muestra los resultados de la búsqueda al usuario

Extensiones

-

Satisfacción: 4

No Satisfacción: 3

Dependencias:

Includes: -

Extends: - Buscar

6.2.11 Eliminar Editorial

Caso de uso

Eliminar Editorial

Actor: Sistema

Precondición:

Trigger: El usuario ha eliminado una Revista/Congreso cuya editorial no tenía más entidades publicadoras asignadas a él.

Escenario Principal

1. El sistema detecta que se acaba de eliminar un congreso o revista.
2. El sistema obtiene el número de congresos y revistas que tiene asignado la editorial al cual pertenece el congreso o revista que se acaba de eliminar.
3. El sistema comprueba que sólo tiene una.
4. El sistema da de baja la editorial.

Extensiones

- 3.a La editorial tiene más de una entidad publicadora asignada.
- 2.a.1 El sistema acaba el caso de uso.

Satisfacción: 3

No Satisfacción: 3

Dependencias

Includes: -

Extends: - Eliminar Revista

6.2.12 Nueva Editorial

Caso de uso

Nueva Editorial

Actor: Usuario Registrado

Precondición: -Usuario tiene acceso al backend de la web.

Trigger: El usuario ha creado una nueva Revista/Congreso que desea asociar a una nueva editorial que dará de alta.

Escenario Principal

1. El usuario indica que quiere dar de alta una nueva editorial e introduce sus datos
2. El sistema valida los datos introducidos por el usuario
3. El sistema crea una nueva editorial.

Extensiones

2.a Falta algún campo obligatorio

2.a.1 El sistema comunica el error, el caso de uso continúa en el punto 1 manteniendo la información ya introducida

2.b. Ya existe la editorial en el sistema

2.b.1 El sistema comunica el error, el caso de uso continúa en el punto 1 manteniendo la información ya introducida

Satisfacción: 3

No Satisfacción: 3

Dependencias

Includes:

Extends: Nueva Revista

6.2.13 Consultar Entidad de Calidad

Caso de uso

Consultar Entidad de Calidad

Actor: Usuario Registrado, sistema

Precondición:

-El usuario es un usuario dado de alta en el sistema

Trigger: -

Escenario Principal

1. A partir de la lista de Entidades de Calidad, el usuario selecciona la entidad de calidad que desea consultar.
2. El sistema obtiene la información básica de la Entidad de Calidad consultada
3. El sistema obtiene todas Ediciones e Issues que han sido evaluadas por la Entidad de Calidad seleccionada
4. El sistema muestra la información de la Entidad de Calidad consultada.

Extensiones

4.a. El usuario decide editar la información

4.a.1 Se inicia el caso de uso Editar Entidad de Calidad

Satisfacción: 2

No Satisfacción: 3

Dependencias: -

Includes: - Editar Entidad de Calidad

Extends: -

6.2.14 Nueva Entidad de Calidad

Caso de uso

Nueva Entidad de Calidad

Actor: Usuario Registrado

Precondición: -Usuario tiene acceso al backend de la web.

Trigger: -

Escenario Principal

1. El usuario indica que quiere dar de alta una nueva Entidad de Calidad e introduce sus datos.
2. El usuario indica que la entidad de Calidad valora de forma “numérica”.
3. El sistema valida los datos introducidos por el usuario.
4. El sistema crea la nueva Entidad de Calidad
5. El sistema confirma redireccionando al usuario a la ficha de la nueva Entidad de Calidad.

Extensiones

[1-2].a. El usuario cancela el alta

[1-2].a.1 Acaba el caso de uso

2.a. El usuario indica que la entidad de calidad valora mediante una “lista” de valoraciones.

2.a.1 El usuario introduce la lista de valoraciones de mayor a menor categoría.

2.a.2 El sistema valida los datos introducidos por el usuario.

2.a.3 El sistema crea las categorías y las asigna a la Entidad de Calidad y el caso de uso continua en el punto 4.

3.a Falta algún campo obligatorio

2.a.1 El sistema comunica el error, el caso de uso continúa en el punto 1 manteniendo la información ya introducida

3.b. Ya existe la Entidad de Calidad con el mismo nombre

3.b.1 El sistema comunica el error, el caso de uso continúa en el punto 1 manteniendo la información ya introducida

Satisfacción: 2

No Satisfacción: 4

Dependencias -

Includes:

Extends:

6.2.15 Editar Entidad de Calidad

Caso de uso

Editar Entidad de Calidad

Actor: Usuario Registrado

Precondición:

-Usuario tiene acceso al backend de la web.

Trigger: -

Escenario Principal

1. El usuario indica al sistema cual es el campo o atributo que quiere editar de la Entidad de Calidad
2. El sistema responde permitiendo al usuario editar el atributo seleccionado
3. El usuario introduce el nuevo valor del atributo
4. El sistema valida el nuevo valor
5. El sistema confirma el cambio mostrando al usuario el nuevo valor del atributo.

Extensiones

- 4.a El usuario modifica el nombre de la Entidad de Calidad y ya existe en el sistema una entidad de calidad con el mismo nombre.
 - 4.a.1 El sistema comunica el error
- 4.b El usuario modifica el nombre de la Entidad de Calidad
 - 4.b.1 El sistema confirma el cambio mostrando al usuario el nuevo valor del atributo y recargando la vista con el nuevo valor del identificador.

Satisfacción: 3

No Satisfacción: 3

Dependencias: -

Includes: -

Extends: -

6.2.16 Eliminar Entidad de Calidad

Caso de uso

Eliminar Entidad de Calidad

Actor: Usuario Registrado

Precondición:

-Usuario tiene acceso al backend de la web.

Trigger: -

Escenario Principal

1. A partir de la lista de Entidades de Calidad el usuario selecciona las Entidades de Calidad que desea dar de baja.
2. El sistema solicita una confirmación
3. El usuario confirma que quiere eliminar las Entidades de Calidad seleccionadas
4. El sistema da de baja las Entidades de Calidad seleccionadas
5. El sistema elimina las valoraciones de Issues y Ediciones que ha hecho la Entidad de Calidad eliminada.
6. El sistema confirma mostrando la nueva lista de Entidades de Calidad disponibles en el sistema.

Extensiones

- 3.a. El usuario cancela la petición de eliminación y termina el caso de uso.

Satisfacción: 3

No Satisfacción: 2

Dependencias

Includes: -

Extends: -

6.2.17 Consultar Tópicos

Caso de uso

Consultar Tópicos

Actor: Usuario Registrado, sistema

Precondición:

-El usuario es un usuario dado de alta en el sistema

Trigger: -

Escenario Principal

1. A partir de la lista de Tópicos, el usuario selecciona el tópico que desea consultar.
2. El sistema obtiene la información básica del tópico consultado
3. El sistema obtiene todas Ediciones, Issues, Revistas y Congresos que tienen como tópico al tópico consultado.
4. El sistema obtiene los tópicos similares del tópico consultado.
5. El sistema muestra al usuario toda la información del tópico consultado.

Extensiones

5.a. El usuario decide editar la información

5.a.1 Se inicia el caso de uso Editar Tópico

Satisfacción: 2

No Satisfacción: 3

Dependencias: -

Includes: - Editar Tópico

Extends: -

6.2.18 Nuevo Tópico

Caso de uso

Nuevo Tópico

Actor: Usuario Registrado

Precondición: -Usuario tiene acceso al backend de la web.

Trigger: -

Escenario Principal

1. El usuario indica que quiere dar de alta un nuevo tópico e introduce los datos del tópico.
2. El usuario utiliza un “Buscador de tópicos” para añadir los tópicos similares al tópico que se desea dar de alta.
3. El sistema valida los datos introducidos por el usuario
4. El sistema crea el nuevo Tópico y asocia sus tópicos similares
5. El sistema confirma redireccionando al usuario a la ficha de la nuevo Tópico creado.

Extensiones

- 1.a. El usuario cancela el alta
 - 1.a.1 Acaba el caso de uso
- 2.a Falta algún campo obligatorio
 - 2.a.1 El sistema comunica el error, el caso de uso continúa en el punto 1 manteniendo la información ya introducida
- 2.b. Ya existe el tópico en el sistema
 - 2.b.1 El sistema comunica el error, el caso de uso continúa en el punto 1 manteniendo la información ya introducida

Satisfacción: 2

No Satisfacción: 3

Dependencias

Includes:

Extends:

6.2.19 Editar Tópico

Caso de uso

Editar Tópico

Actor: Usuario Registrado

Precondición:

-Usuario tiene acceso al backend de la web.

Trigger: -

Escenario Principal

1. El usuario indica al sistema cual es el campo o atributo que quiere editar del tópico
2. El sistema responde permitiendo al usuario editar el atributo seleccionado
3. El usuario introduce el nuevo valor del atributo
4. El sistema valida el nuevo valor
5. El sistema confirma el cambio mostrando al usuario el nuevo valor del atributo.

Extensiones

- 4.a El usuario modifica el nombre del Tópico y ya existe en el sistema un tópico con el mismo nombre.
 - 4.a.1 El sistema comunica el error
- 4.b El usuario modifica el nombre del Tópico
 - 4.b.1 El sistema confirma el cambio mostrando al usuario el nuevo valor del atributo y recargando la vista con el nuevo valor del identificador.

Satisfacción: 3

No Satisfacción: 3

Dependencias: -

Includes: -

Extends: -

6.2.20 Eliminar Tópico

Caso de uso

Eliminar Tópico

Actor: Usuario Registrado

Precondición:

-Usuario tiene acceso al backend de la web.

Trigger: -

Escenario Principal

1. A partir de la lista de Tópicos el usuario selecciona los Tópicos que desea dar de baja.
2. El sistema solicita una confirmación
3. El usuario confirma que quiere eliminar los Tópicos seleccionados
4. El sistema da de baja los Tópicos seleccionados
5. El sistema elimina las asociaciones de similitud con otros tópicos.
6. El sistema confirma mostrando al usuario la nueva lista de Tópicos disponibles en el sistema.

Extensiones

- 3.a. El usuario cancela la petición de eliminación y termina el caso de uso.

Satisfacción: 3

No Satisfacción: 2

Dependencias

Includes: -

Extends: -

7

Diseño

7.1 Arquitectura	49
7.1.1 Integración con Joomla	50
7.2 Diseño del modelo de comportamiento	51
7.2.1 Operaciones Básicas.....	51
7.2.2 APIgesCORE.....	53
7.2.3 Conexión con Scopus	55
7.3 Patrones de diseño del sistema	57
7.4 Diseño de las interfaces del proyecto	58
7.4.1 Diseño del Menú de la aplicación	58
7.4.2 Storyboards e interacción con el usuario.....	59
7.5 Diseño de la BBDD	65
7.5.1 Diseño Lógico de la BBDD.....	65
7.5.2 Diseño Físico de la BBDD	66

7.1 Arquitectura

La arquitectura de la aplicación se basa en el patrón MVC (Modelo-Vista-Controlador) y utiliza la técnica de arquitectura REST¹¹ para implementar el Web Service, el cual dará servicio al cliente, que se conectará mediante una API.

En cuanto a las tecnologías, la aplicación por el lado del servidor corre sobre Apache¹² y utiliza MySQL¹³ como sistema de gestión de la base de datos.

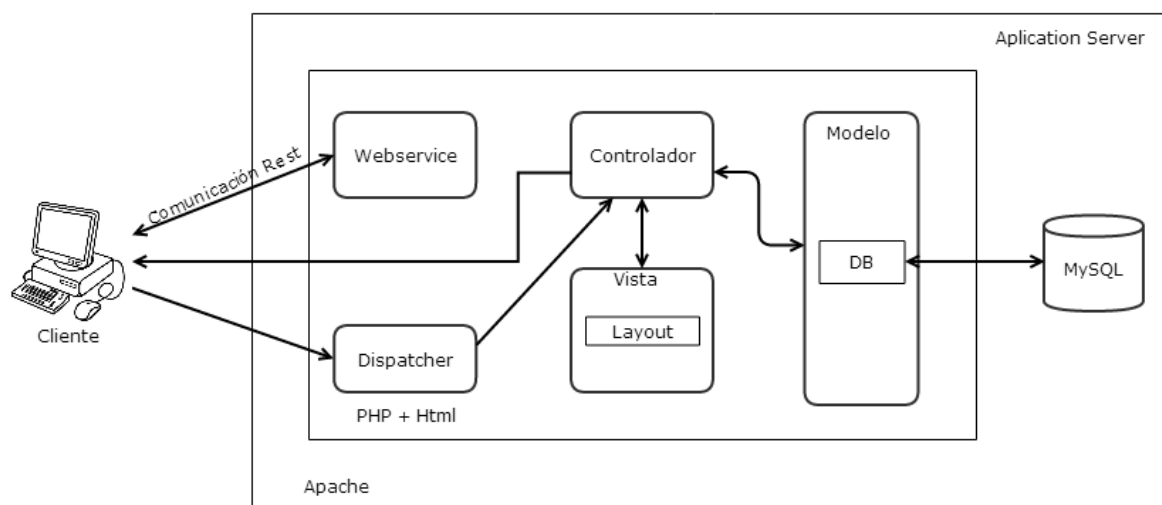


Figura 7.1: Arquitectura del proyecto

En cuanto al Web Service (RESTful), si hiciésemos un zoom en el servidor obtendríamos:

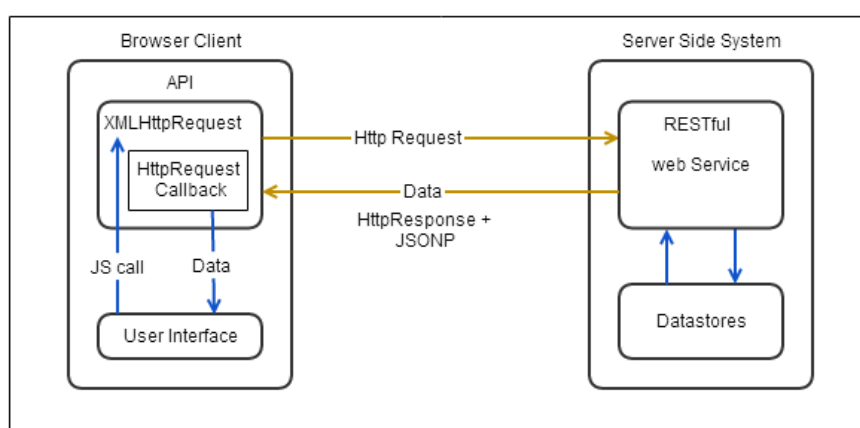


Figura 7.2: Arquitectura del web Service

¹¹ **REST**: La Transferencia de Estado Representacional o REST es una técnica de arquitectura software para sistemas hipermedia distribuidos

¹² **Apache**: El servidor HTTP Apache es un servidor web HTTP de código abierto, que implementa el protocolo HTTP/1.1 y la noción de sitio virtual.

¹³ **MySQL**: Es un sistema de gestión de bases de datos relacional, multihilo y multiusuario.

En cuanto al flujo básico de la arquitectura, es la siguiente:

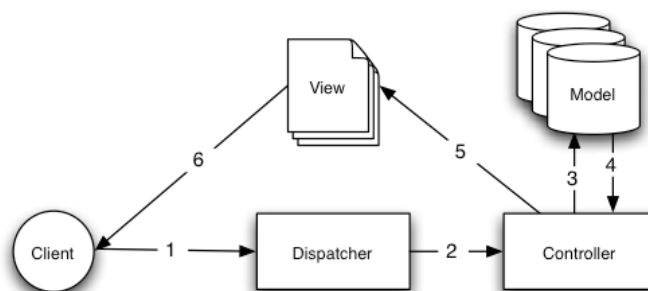


Figura 7.3: Flujo básico de la arquitectura

7.1.1 Integración con Joomla

Como hemos comentado anteriormente Joomla también basa su sistema bajo el patrón de diseño MVC. Es un sistema que separa completamente la parte de backend y frontend, siendo la primera destinada a los usuarios que tienen acceso como administradores a la web.

Joomla está dividido en plugins, módulos y componentes, siendo estos últimos los de mayor tamaño. El proyecto se integra con Joomla como un componente externo el cual pasará a formar parte de su lista de componentes.

Dado que GesCORE es un proyecto destinado básicamente al backend de Joomla, la mayoría de elementos estarán ubicados en esa zona. Así pues, el siguiente gráfico nos indica la integración de cada componente de GesCORE con el sistema general de Joomla.

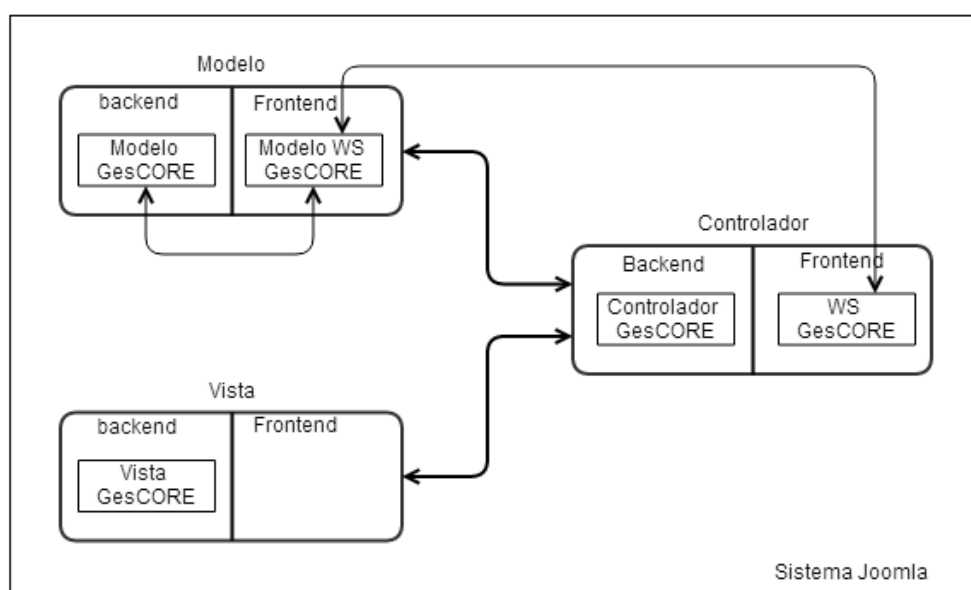


Figura 7.4: Integración de GesCORE con Joomla

7.2 Diseño del modelo de comportamiento

Con la finalidad de conocer mejor el funcionamiento de la arquitectura del proyecto, hemos decidido poner 4 operaciones descritas en un marco de **diagrama de secuencia**.

7.2.1 Operaciones Básicas

Operación 1: Consultar Entidad de Calidad

Esta operación (13° caso de uso) consiste en que el usuario solicita al sistema ver una Entidad de Calidad determinada, para ello el “Dispatcher”¹⁴ indicará al controlador la vista solicitada por el cliente, el cual a su vez preparará la vista solicitando al modelo los datos que crea oportuno para presentar al cliente la vista con los datos de la Entidad de Calidad solicitada.

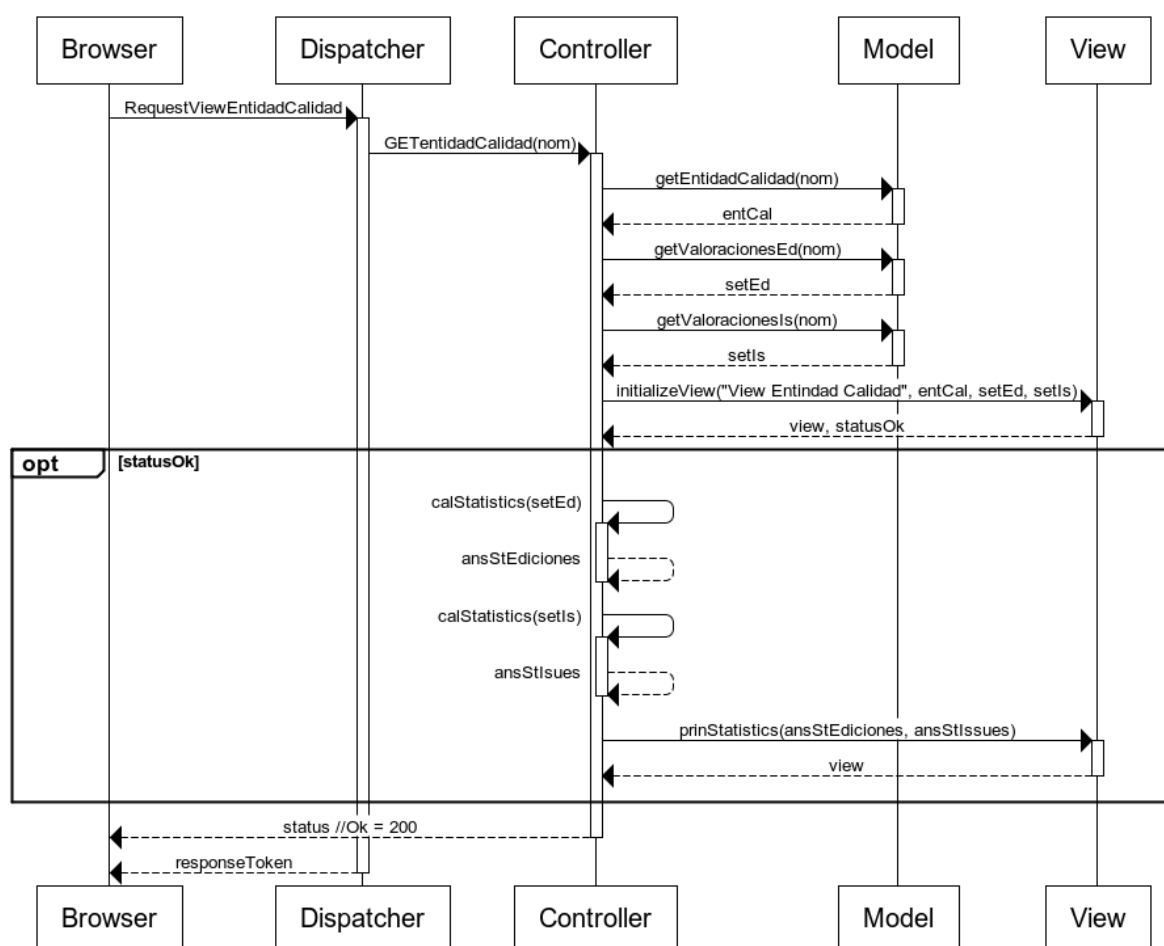


Figura 7.5: Diagrama de secuencia Consultar Entidad de Calidad

¹⁴ Dispatcher: Proceso central en una instancia. Es responsable, entre otras cosas, para iniciar los procesos de trabajo y la distribución de la carga de transacciones a través de procesos de trabajo. En Joomla es el encargado de hacer de “puente” entre las peticiones del cliente y el servidor.

Operación 2: Nueva Issue

Esta operación consiste en dar de alta una nueva Issue. Lo primero es obtener la vista, para ello el “Dispatcher” comunica al controlador que hay un Request de “Nueva Issue”, como en la anterior operación el Controlador se encarga de preparar la vista e iniciarla. El Cliente hacer Post para crear la nueva Issue. Adicionalmente el cliente, en caso de requerirlo, puede crear una nueva revista para la nueva Issue que quiere dar de alta.

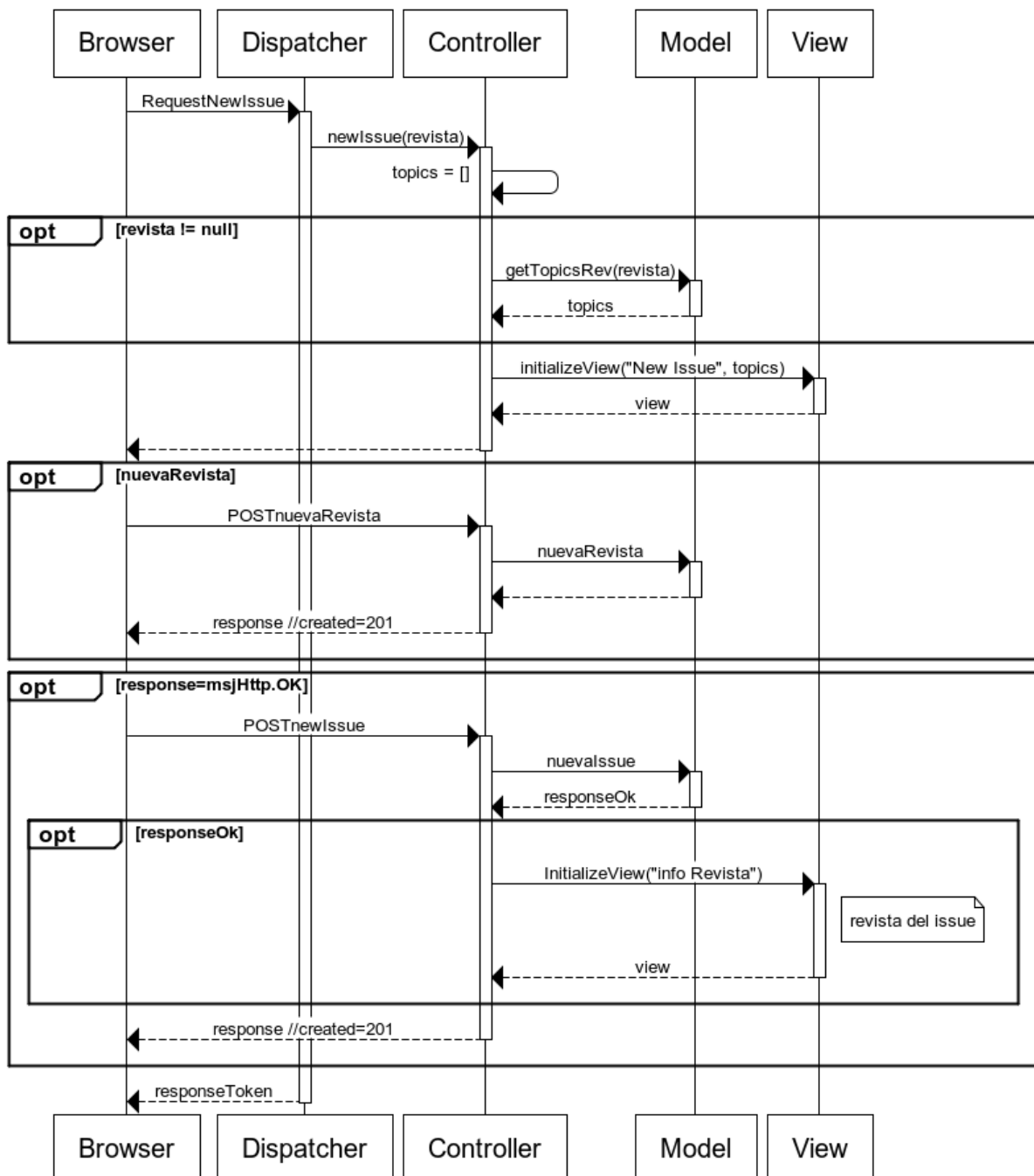


Figura 7.6: Diagrama de secuencia Nueva Issue

7.2.2 APIgesCORE

¿Qué es APIgescore?

El GesCORE Application Programmer Interface (API) es una herramienta que se importa el cliente con la finalidad de conectarse al sistema. Esta API también permite crear un buscador de publicaciones eficiente y fácil de utilizar, el cual se podrá integrar en cualquier plataforma, además permite presentar los resultados de búsqueda de GesCORE a través de una interfaz de usuario del producto que estas personas desarrollen.

¿Qué se puede hacer con APIgesCORE?

- Obtener las publicaciones que se encuentran en GesCORE mediante un filtrado.
- Hacer una búsqueda avanzada o más personalizada de los elementos en GesCORE.
- Obtener un enlace de los elementos accedidos (publicaciones, tópicos, entidades de calidad, etc.) para luego gestionarlos en la aplicación.
- Imprimir los resultados en la interfaz de usuario de la web. También existe la posibilidad de darle a la API una salida de resultados personalizada.
- Posibilidad de ejecutar una función de callback en el momento del retorno de los resultados.

¿Cómo está diseñado?

Como podemos ver en la arquitectura del proyecto (Figura 7.1) la API es una herramienta que permite conectarse al Web Service, el cual basa su arquitectura sobre RESTful, este diseño permite que el sistema sea transparente al desarrollador.

A continuación un ejemplo del uso de APIgesCORE.

Operación 3: Buscar (APIgesCORE)

Esta operación (2do caso de uso) consiste en que hay un usuario que quiere buscar publicaciones de su interés dentro de la aplicación de GesCORE. Para ello utiliza la API proporcionada por la aplicación. La API envía al Controlador los datos proporcionados por el cliente (filtro, tipo de elemento a buscar, fechas, etc.) y adicionalmente le envía una función de “Callback¹⁵”. Como ya hemos explicado antes, la API ya tiene una función de imprimir básica, por lo que no debería preocupar de ello.

¹⁵ Callback: También conocidas como funciones de devolución de llamada o retro-llamada. Se usa en el desarrollo de aplicaciones y sirve para encapsular partes del código dentro de una función.

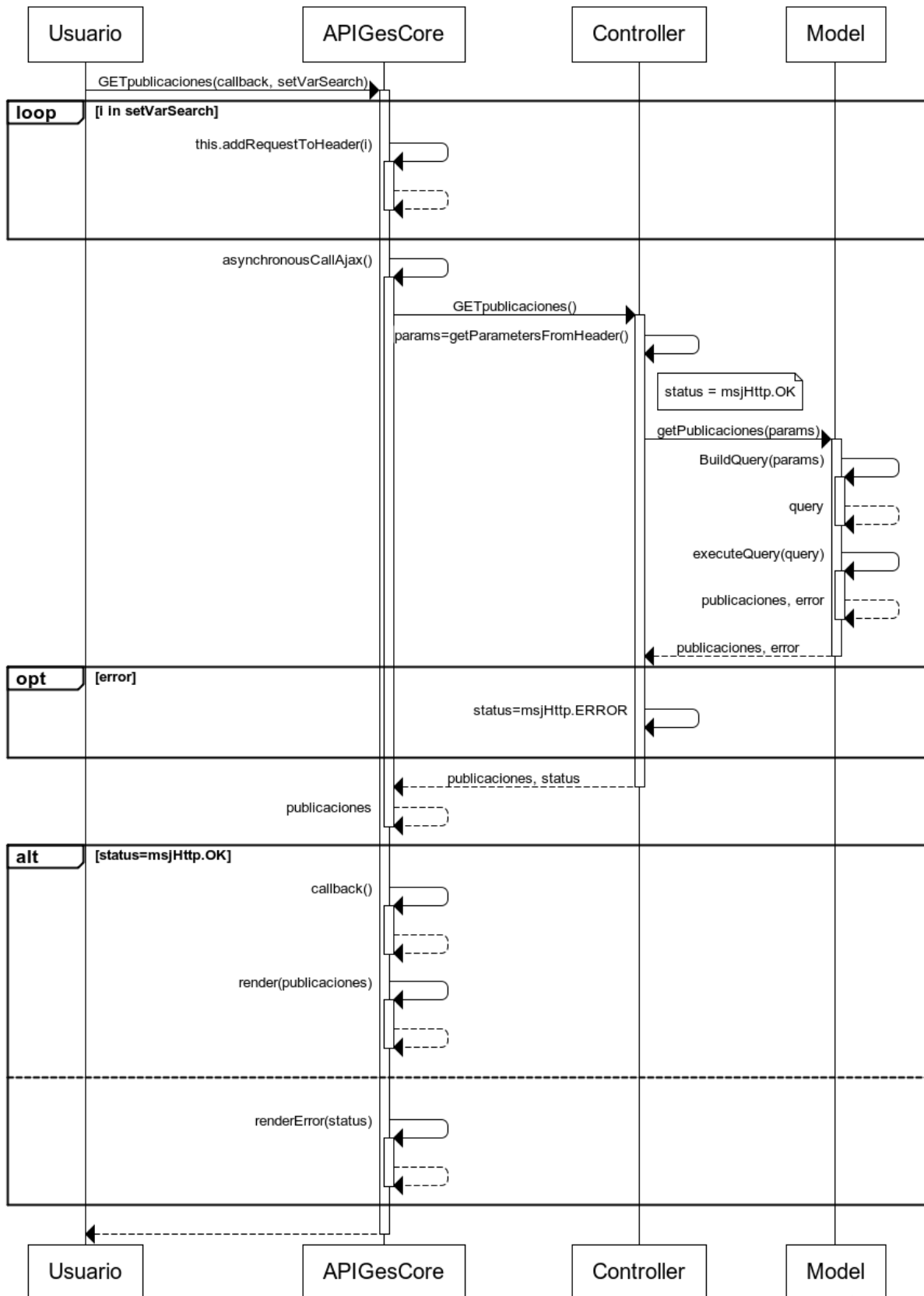


Figura 7.7: Diagrama de secuencia Buscar (APIgesCORE)

7.2.3 Conexión con Scopus

La conexión con Scopus se realiza a través de su API. Esta API nos permite hacer una búsqueda dentro de su sistema.

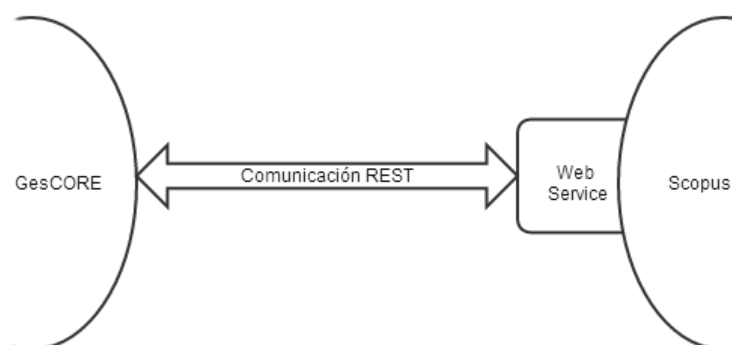


Figura 7.8: Comunicación entre GesCORE y Scopus

El funcionamiento de la API de Scopus es parecido a APIgesCORE con ciertos detalles que le hacen diferentes. Una de las diferencias es que la API de Scopus es más restrictiva a la hora de acceder a su sistema y también en los resultados que se obtienen. Estas restricciones son:

- Se ha de registrar en Scopus y solicitar una APIkey¹⁶ para acceder al contenido de Scopus.
- No se pueden solicitar más de 25 publicaciones en una misma transferencia.
- No se puede listar más de 5000 publicaciones en una misma búsqueda.

Estas restricciones se deben a que dado que Scopus es una plataforma mundialmente conocida, desean controlar el tráfico a su sistema en la mayor medida posible.

La arquitectura de Scopus se basa en REST. No podemos determinar cuál es su funcionamiento interno. Aun así, tenemos una serie de operaciones que nos brinda la posibilidad de tener en todo momento un control sobre las publicaciones que deseemos importar a GesCORE.

A continuación mediante un ejemplo práctico explicaremos el funcionamiento de la API de Scopus.

Operación 4: Importar (desde Scopus)

Esta operación consiste en utilizar la API de Scopus para importar a GesCORE las publicaciones que queramos. Con las publicaciones obtenidas de Scopus damos de alta en GesCORE las nuevas publicaciones que seleccionemos.

¹⁶ **APIkey:** Es un identificador el cual se asigna a una persona u organización con tal de controlar el acceso a un sistema mediante su API.

El procedimiento es el mismo que en las dos primeras operaciones, el “Dispatcher” comunica al Controlador que hay una nueva petición de importación y envía al Controlador la información proporcionada por el cliente para mostrar la vista solicitada. El cliente se encarga de hacer POST sobre el controlador con la finalidad de dar de alta las nuevas publicaciones y hacer persistente los datos en el Modelo del sistema. Luego el controlador inicia la vista que se le enviará al cliente con el resumen de la importación (fallos y aciertos).

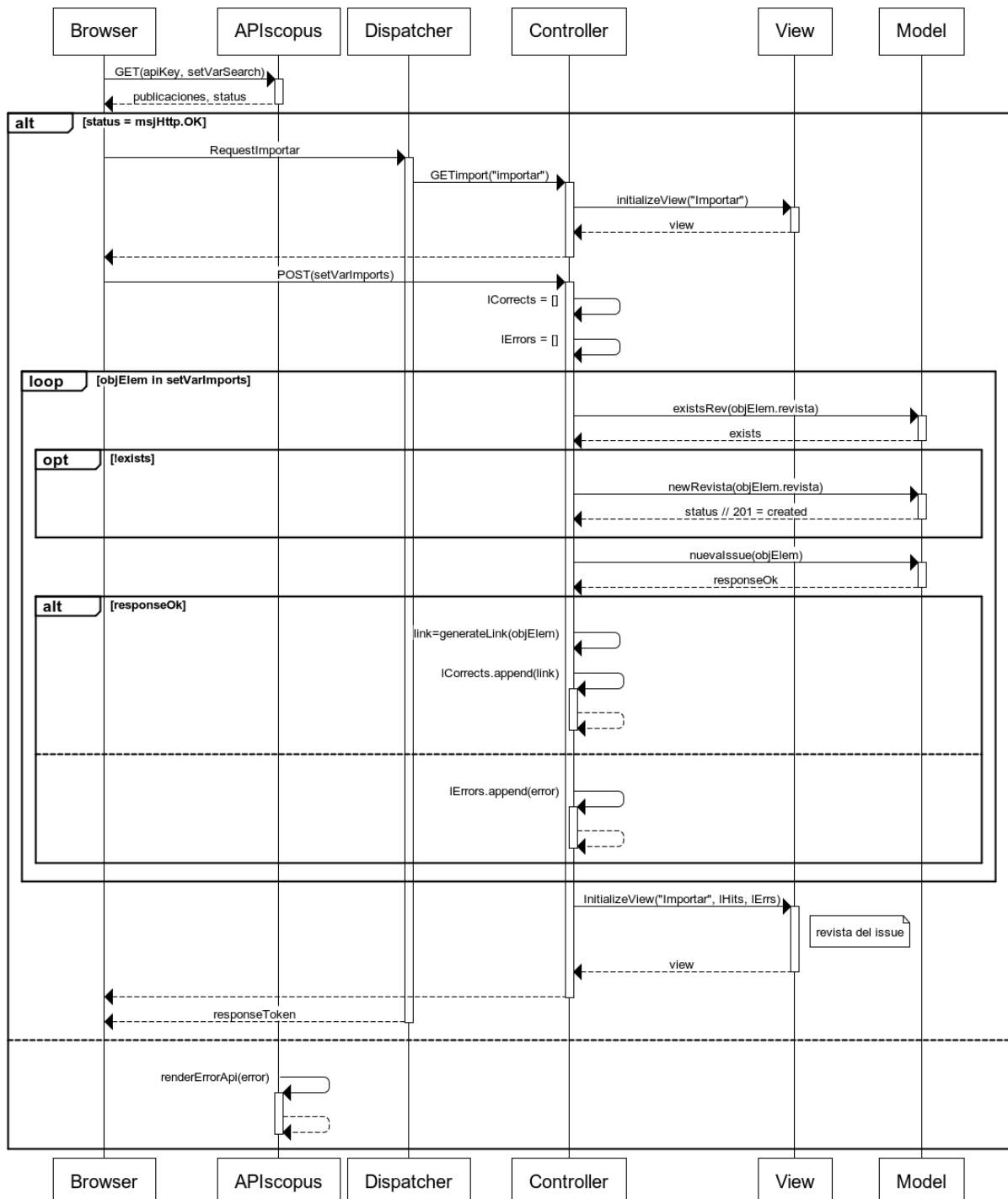


Figura 7.9: Diagrama de secuencia, operación Importar

7.3 Patrones de diseño del sistema

Como ya hemos explicado anteriormente, el sistema está basado sobre el patrón MVC (Model-Vista-Controlador).

En cuanto al patrón de diseño de la capa de dominio, para el Controlador se ha usado el patrón de diseño “Front Controller” de manera especializada.

Las responsabilidades básicas del controlador son:

- Decodificar la URL y extraer cualquier información para realizar acciones sobre el sistema.
- Crear y llamar a cualquier objeto del Modelo para preparar y procesar los datos para el cliente. De esta manera añadir seguridad al sistema evitando tener acceso directo a los objetos del modelo.
- Obtener la información del “Dispatcher” para instanciar las vistas que se muestran al cliente.

El siguiente gráfico explica el diseño del sistema:

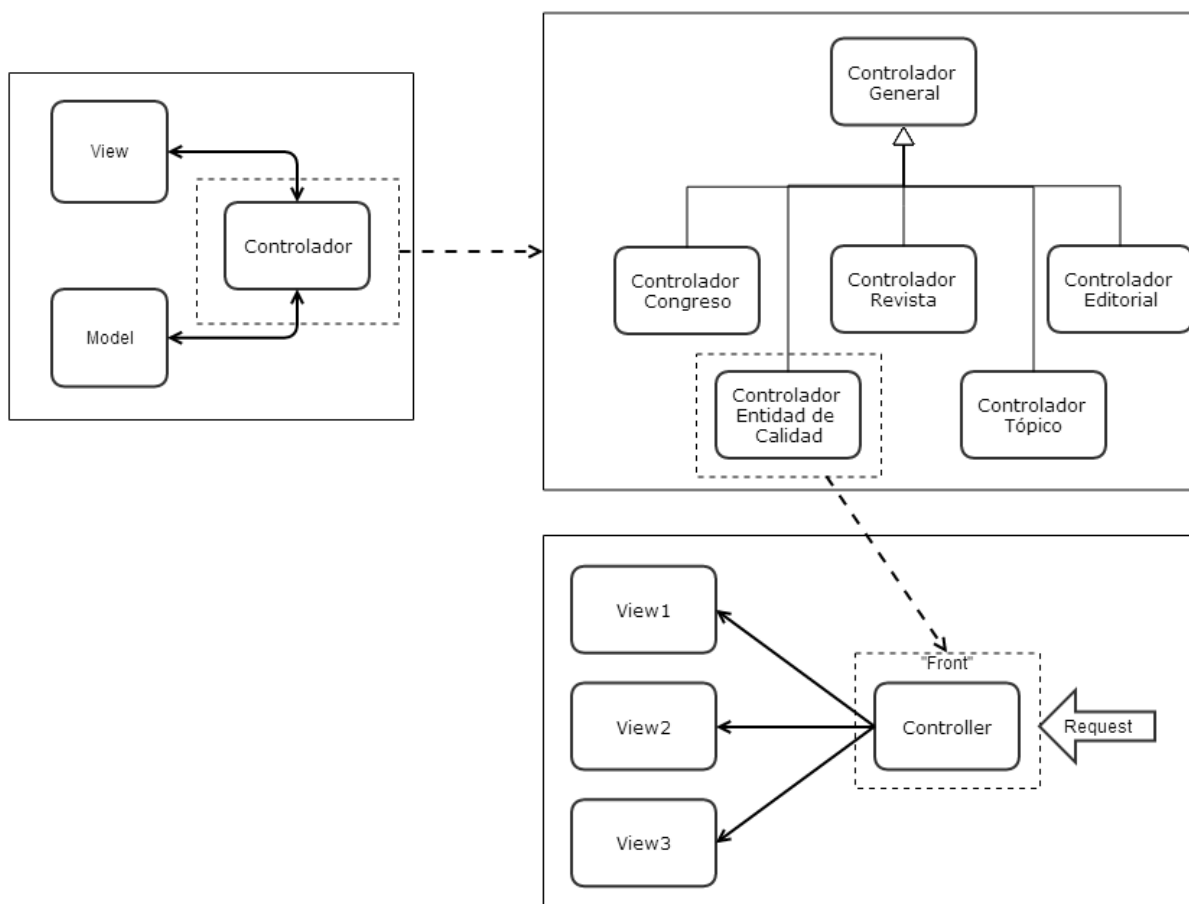


Figura 7.10: Patrón de diseño de GesCORE

7.4 Diseño de las interfaces del proyecto

El diseño de las interfaces del componente se realizó respetando el patrón establecido por Joomla para la creación de componentes en Backend.

De esta forma siguiendo las reglas de diseño de Joomla para la creación de interfaces se respetó su diseño con los siguientes elementos:

- Menú
- Tablas
- Entradas de formularios
- Paginadores
- Botones

7.4.1 Diseño del Menú de la aplicación

Como ya hemos mencionado, el Menú de GesCORE se realizó siguiendo las reglas de implementación de Joomla. No obstante, los elementos del Menú son propios del desarrollador. Y el diseño de esta interfaz es la siguiente:

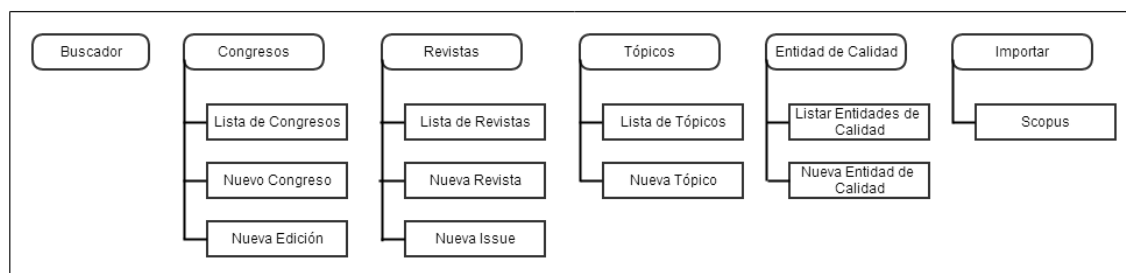


Figura 7.11: Diseño del menú de GesCORE

7.4.2 Storyboards e interacción con el usuario

El diseño de las interfaces (mockups¹⁷) pretende definir cómo será la parte del sistema que interactúa con el usuario.

En este apartado definiremos los mecanismos con los cuales los usuarios podrán interactuar con el sistema y la manera de mostrar la información al usuario.

Los mockups que hemos diseñado son los mismos ejemplos que hemos usado en los diagramas de secuencia.

Mockup 1: Nueva Issue

Esta operación al ser un poco larga, la hemos dividido en 3 diagramas, a nivel de interacción con el usuario se podría ver como una detrás de la otra.

Como podemos observar, cuando queramos dar de alta una nueva issue, se podrá crear la Revista en caso que su revista no exista en el sistema.

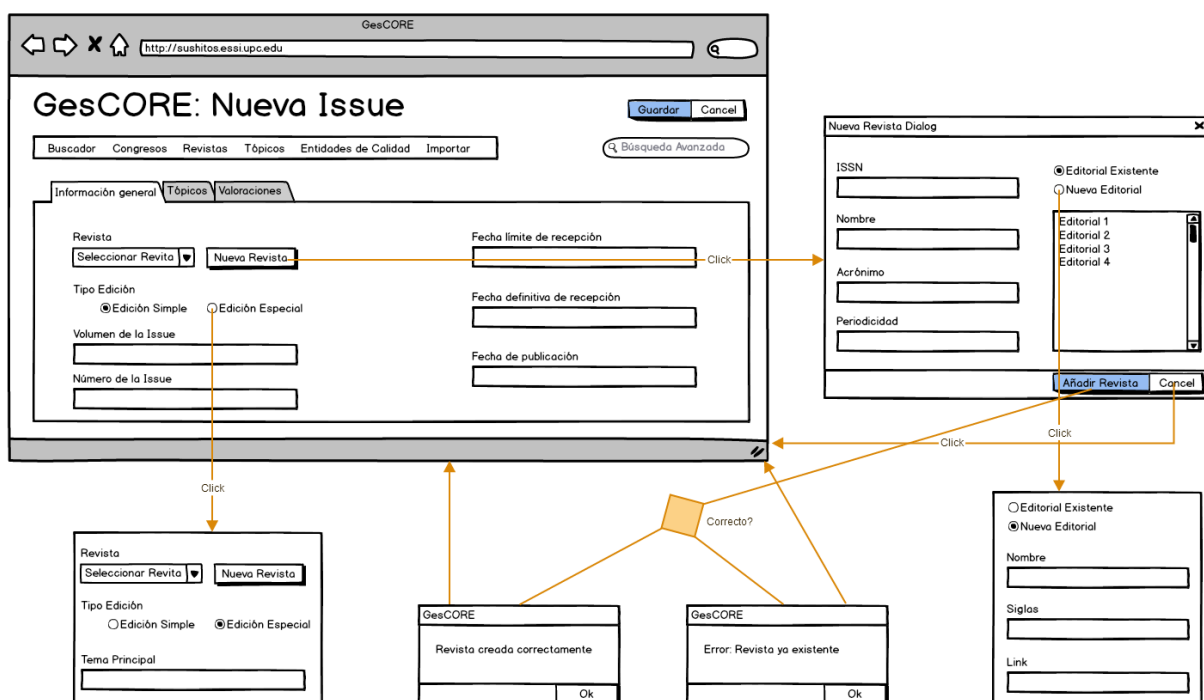


Figura 7.12: Mockup - Operación nueva issue, apartado de información general

¹⁷ **Mockup:** Es un modelo a escala o tamaño real de un diseño, utilizado para la demostración, evaluación del diseño, promoción, y para otros fines.

En la sección de tópicos, podemos añadir tópicos de 3 formas: Mediante los tópicos de la revista a la que pertenece, buscando tópicos en un buscador o creando nuevos tópicos.

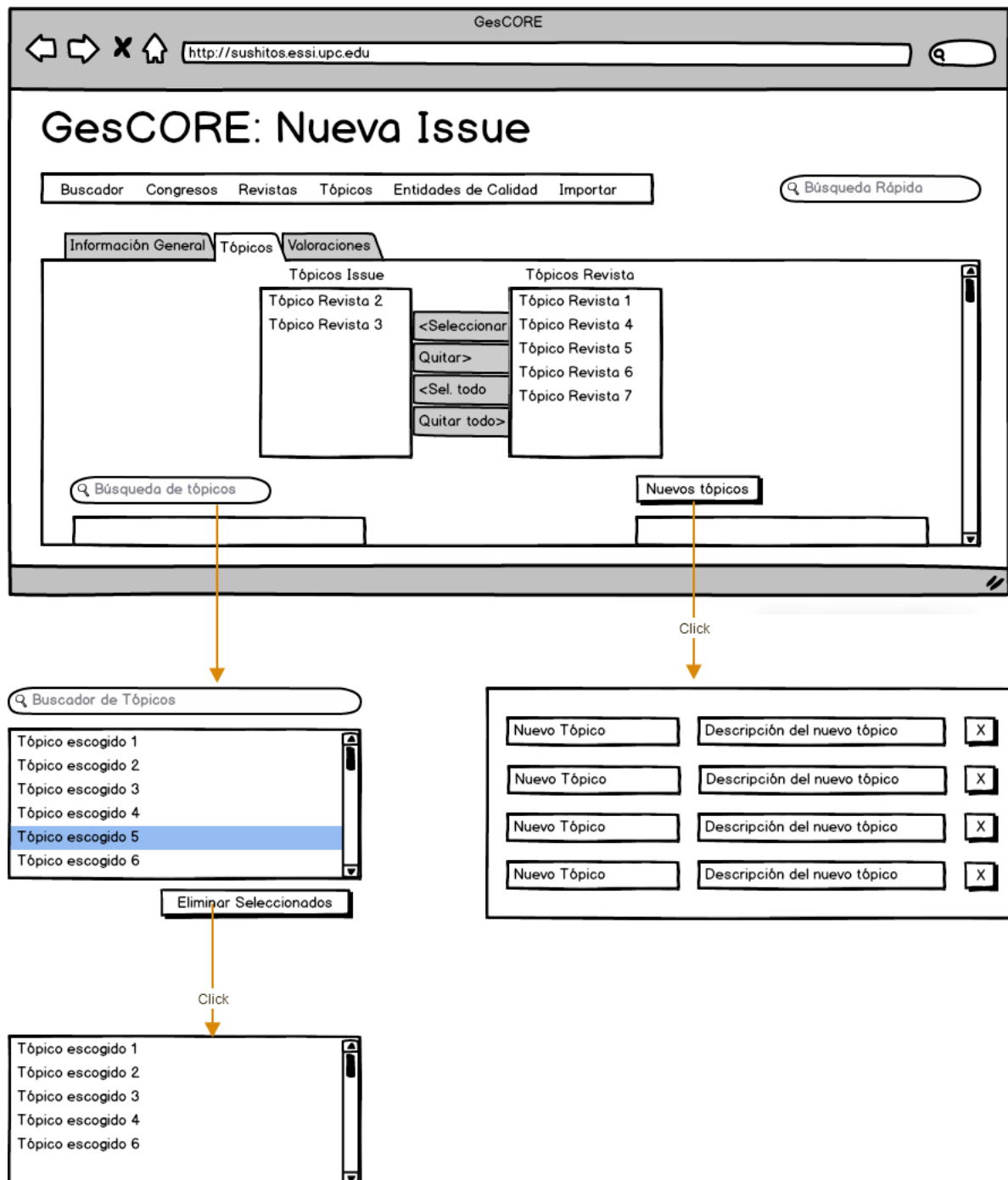


Figura 7.13: Mockup - Operación nueva issue, apartado de tópicos

Finalmente, añadimos las valoraciones de la nueva Issue. Para ello, el usuario selecciona la entidad de Índice de Calidad de una lista y la valoración que desee añadir repitiendo el paso tantas veces como quiera.

Por último, tiene los botones de guardar y cancelar. En caso que la issue que está dando de alta ya exista o falte algún campo obligatorio el sistema le comunicará y le indicará los errores. De este modo finaliza la operación “Nueva Issue”.

Click

Entidad de índice de valoración	Valoración de la Issue	Eliminar
Seleccionar Revista	valoración	✕
Seleccionar Revista	valoración	✕
Seleccionar Revista	valoración	✕

Añadir nueva valoración

Figura 7.14: Mockup - Operación nueva issue, apartado de entidades de calidad

Mockup 2: Consultar Entidad de Calidad

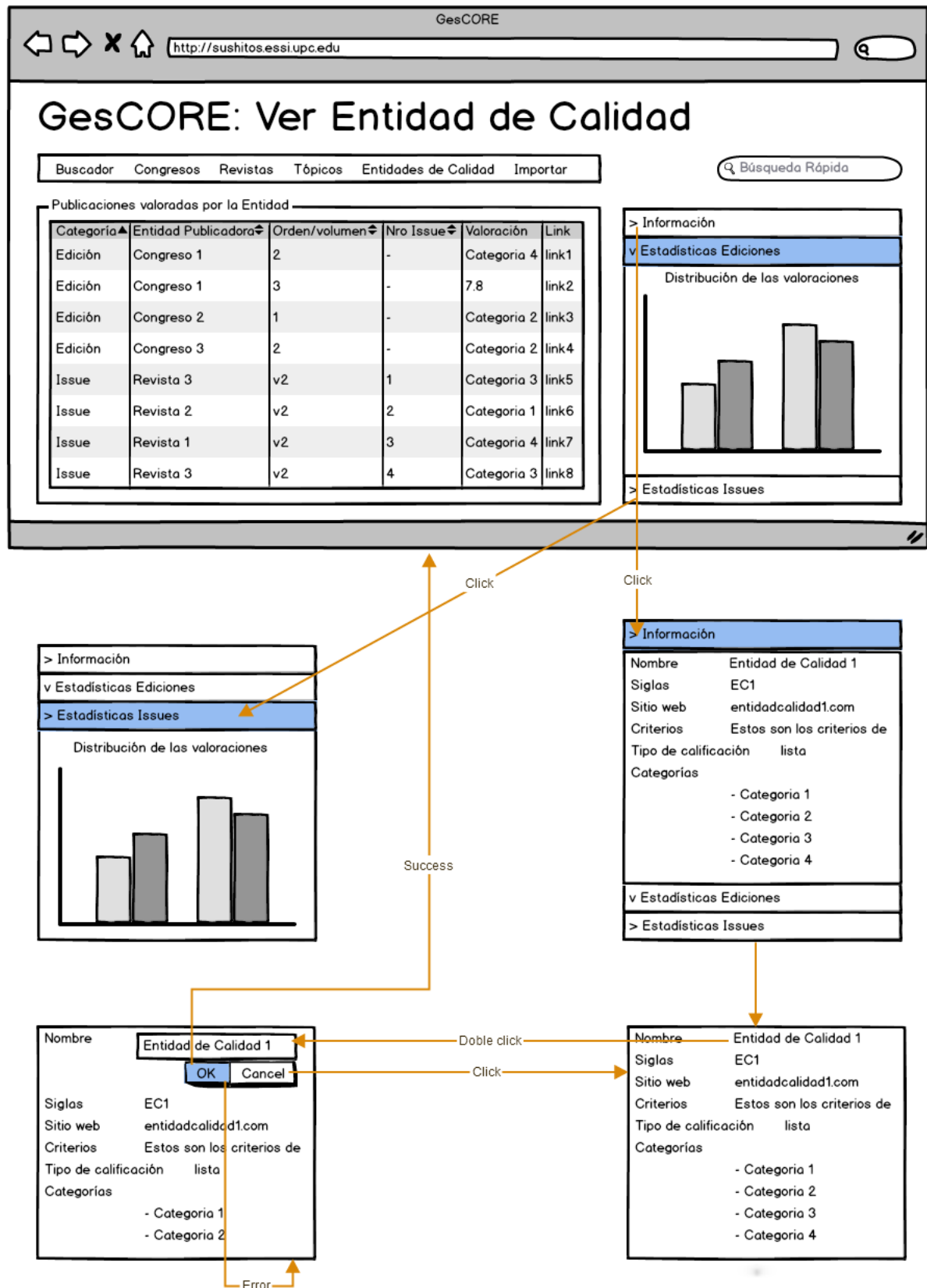


Figura 7.15: Mockup - Operación consultar entidad de calidad

Mockup 3: Buscar (APIgesCORE)

Este es un ejemplo de buscar publicaciones y elementos en general usando APIgesCORE. El usuario introduce un criterio de búsqueda y una de las opciones que le ofrece la API es poder filtrar su búsqueda para tener mayor precisión.

Cabe destacar que la API de GesCORE permite muchas más funcionalidades, muchas de las cuales no son visibles a nivel de interfaz de usuario y que explicaremos en la fase de implementación.

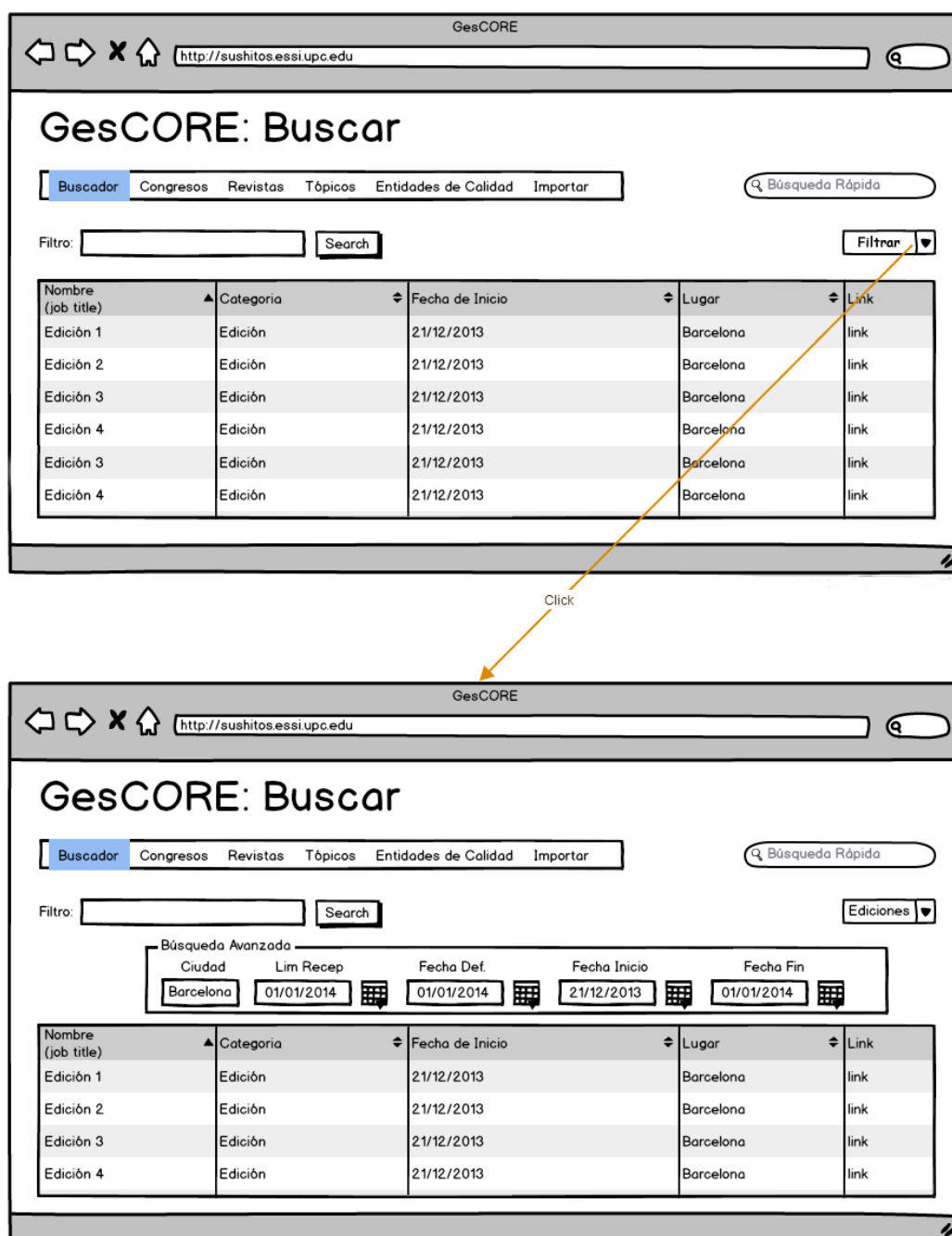


Figura 7.16: Mockup - Operación Buscar (APIgesCORE)

Mockup 4: Importar (desde Scopus)

La siguiente operación aprovecha todos los elementos que nos brinda la API de Scopus. De esta forma el programador sólo tiene que preocuparse por diseñar la interfaz y ocuparse de la comunicación con la API.

Como podemos ver, al finalizar la operación el usuario obtiene un resumen de la importación, con los elementos que se han importado correctamente y los que no lo han hecho.

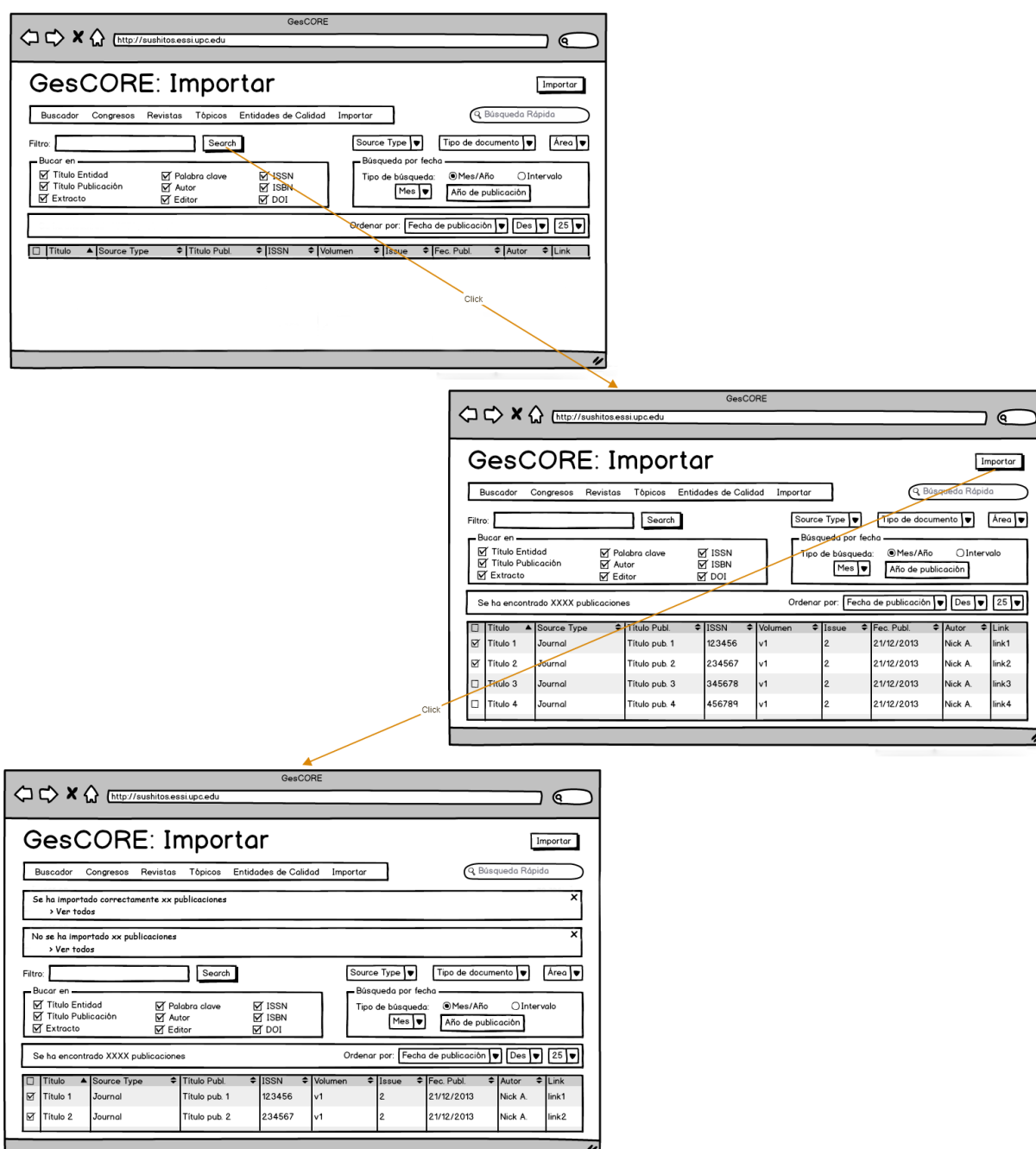


Figura 7.17: Mockup – Operación importar desde Scopus

7.5 Diseño de la BBDD

En este apartado definiremos el diseño de la Base de Datos de GesCORE. Primero mostraremos el diseño lógico de la Base de Datos de forma normalizada y a continuación el diseño físico de la misma.

7.5.1 Diseño Lógico de la BBDD

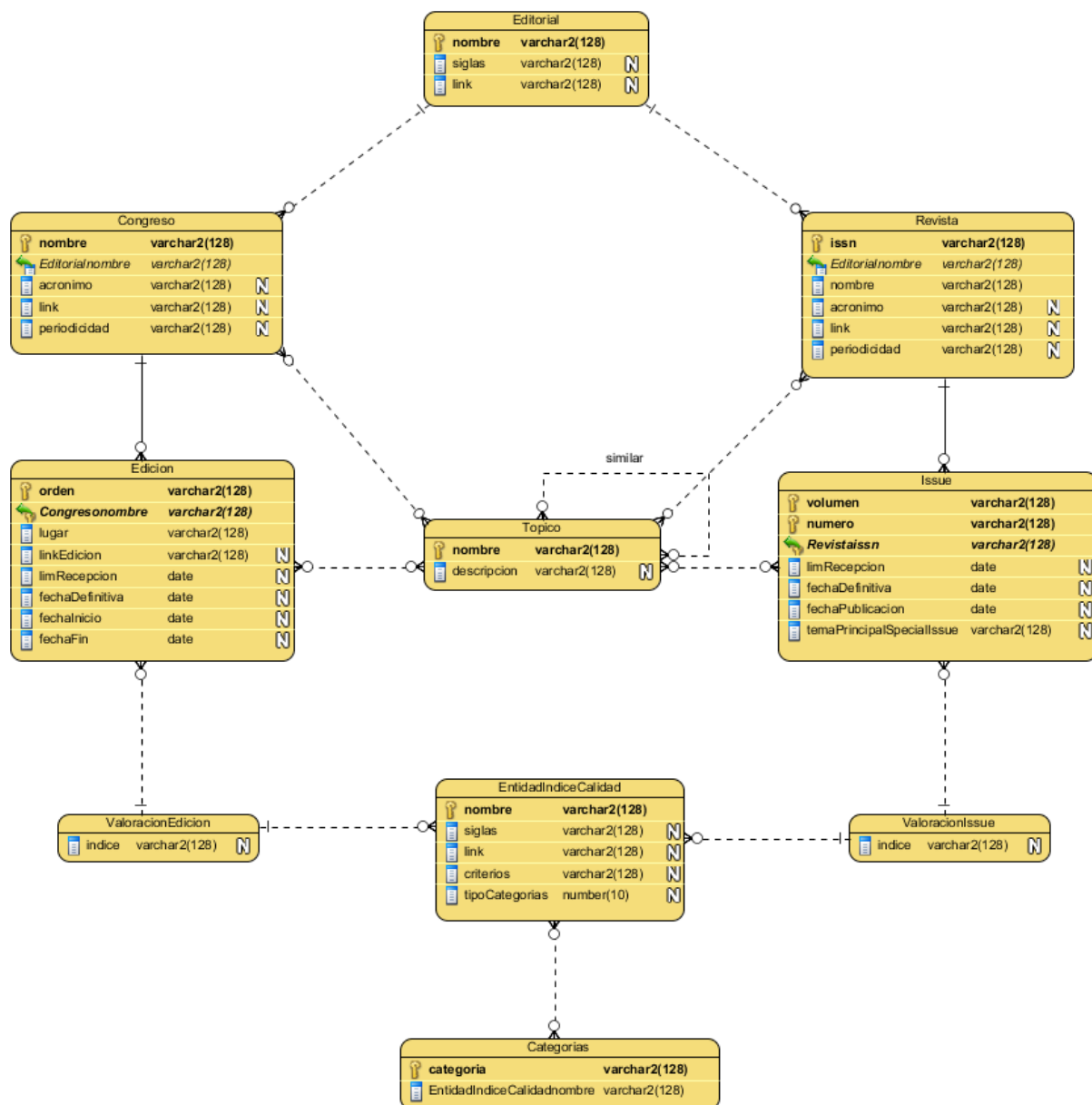


Figura 7.18: Diseño lógico de la Base de Datos

7.5.2 Diseño Físico de la BBDD

En el diagrama siguiente podemos apreciar las diferentes tablas que forman parte de la base de datos con sus respectivas relaciones.

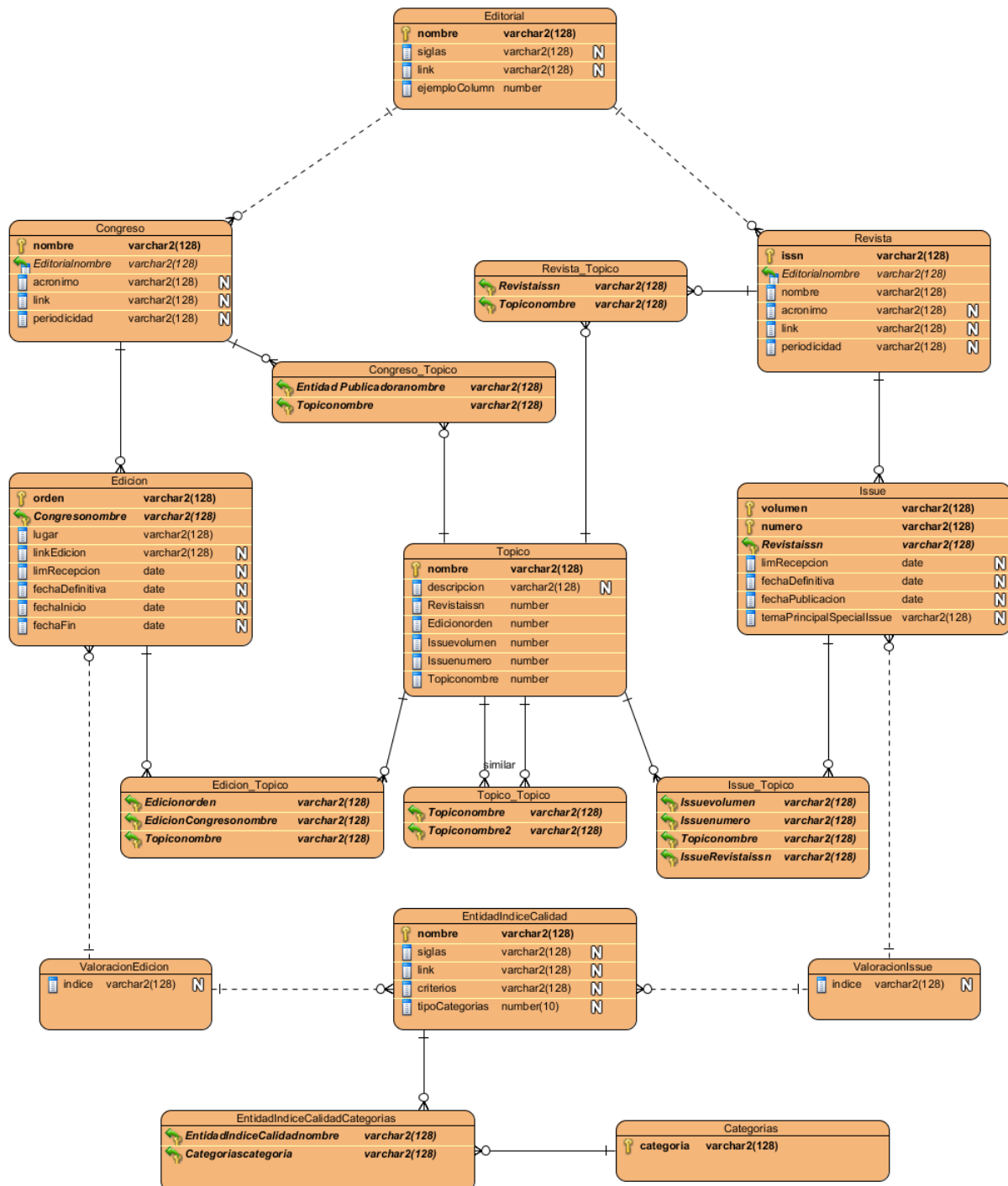


Figura 7.19: Diseño físico de la Base de Datos

8

Implementación y Funcionamiento

8.1 Tecnologías del Proyecto	68
8.1.1 Lado del Servidor	68
8.1.2 Lado del Cliente	68
8.2 APIgesCORE	68
8.2.1 Requisitos previos que ha conocer el desarrollador	68
8.2.2 Ejemplo	69
8.2.3 Operaciones de búsqueda de la API	72
8.2.4 Métodos de filtrado de la API	74
8.2.5 Clase SearchObj	74
8.2.6 Clase editionObj	75
8.2.7 Clase issueObj	76
8.2.8 Valores generales por defecto de la API	76
8.2.9 Comunicación y retorno de resultados	77
8.2.10 Tratamiento de errores y debugging	79
8.3 Gestor de Call For Papers	82
8.4 Método de desarrollo y seguimiento del proyecto	86
8.4.1 Scrum	86
8.4.2 El desarrollo del Proyecto mediante Scrum	87

8.1 Tecnologías del Proyecto

8.1.1 Lado del Servidor

En cuanto a las tecnologías usadas en el proyecto tenemos:

- Lenguajes de programación: PHP y SQL.
- El web service está diseñado sobre una arquitectura RESTful.
- El sistema de gestión de Base de Datos es MySQL

8.1.2 Lado del Cliente

Por el lado del cliente tenemos:

- Lenguajes de programación: Html, JavaScript, CSS.
- Comunicación: AJAX + JSON, AJAX + JSONP
- Librerías: jQuery, jQueryUI

8.2 APIgesCORE

Esta es una de las herramientas de GesCORE, el cual va dirigido a las personas que deseen tener acceso a la información contenida en GesCORE mediante su API.

Ahora definiremos como está implementado APIgesCORE, cuales son las operaciones que se puede utilizar, y finalmente daremos un ejemplo sencillo de su utilización.

8.2.1 Requisitos previos que ha conocer el desarrollador

Para poder utilizar la API el desarrollador ha de estar familiarizado con las siguientes tecnologías:

- Extensible Markup Language (XML). Para más información: <http://www.w3.org/XML/>
- Hypertext Transfer Protocol (HTTP). Para más información: <http://www.w3.org/Protocols/>
- JavaScript Object Notation (JSON). Para más información: <http://www.json.org/>
- Lenguajes: JavaScript y Html

8.2.2 Ejemplo

El siguiente código es un ejemplo básico de la utilización de la API. Como podemos ver, en la parte de la interfaz sólo hay un input, un botón y un “div” para la salida. Esta interfaz puede ser tan amplia como se quiera. Un ejemplo de como de amplia puede ser la interfaz es la operación “Buscar”, la cual se encuentra implementada dentro de GesCORE.

```

1  <html>
2    <head>
3      <title>GesCORE Test/Demo Tool</title>
4      <link REL="stylesheet" TYPE="text/css"
5 href="http://gescore.byethost11.com/components/com_gestorcore/ges
6 coreAPI_hilight.css"/>
7      <script type="text/javascript"
8 src="http://gescore.byethost11.com/components/com_gestorcore/gesc
9 oreAPI.jsp"></script>
10     <script type="text/javascript">
11       callback = function() {
12         document.gescoreForm.searchButton.disabled=false;
13         alert(gescore.getTotalHits());
14       };
15       runSearch = function(){
16         document.gescoreForm.searchButton.disabled= true;
17         var varSearchObj = new searchObj();
18         varSearchObj.setNumResults(50);
19         varSearchObj.setOffset(0);
20         varSearchObj.setSearch
21           (document.gescoreForm.searchString.value);
22         varSearchObj.setSearchInTopics(true);
23         varSearchObj.setSort('fecha');
24         varSearchObj.setSortDirection('desc');
25
26         var edition = new editionObj();
27         edition.setPlace('Barcelona');
28         edition.setLimReception_min('06/06/2013');
29         edition.setStartDate_min('01/06/2014');
30         edition.setStartDate_max('18/06/2014');
31         varSearchObj.setObjectType(edition);
32         gescore.searchAndPrint(varSearchObj);
33       };
34       gescore.setCallback(callback);
35     </script>
36   </head>
37   <body>
38     <form action="" method="post" name="gescoreForm">
39       <input type="text" name="searchString"/>
40       <button onClick="runSearch()" name="searchButton"/>
41       Search</button>
42       <div id="gescore_output"></div>
43     </form>
44   </body>
45 </html>

```

Código 8.1: Ejemplo de APIgesCORE

Funcionamiento del Ejemplo

El funcionamiento de la API es el siguiente:

1. Importamos los elementos para la utilización de la API (línea 5 y 7 del ejemplo).
2. Instanciamos a la clase “searchObj” (línea 16).
3. La instancia de “searchObj” se irá construyendo con la configuración que queramos en cuanto al retorno de los resultados. (línea 16 - 22).
4. En caso que queramos precisar el tipo de elemento que estamos buscando (en el ejemplo “Ediciones de Congresos”), lo podemos hacer instanciando a la clase del elemento a buscar. (línea 24)
5. Esta instancia de la clase se irá construyendo con los filtros que queramos realizar de los elementos a buscar (línea 25 - 28)
6. Asignamos esta instancia al objeto principal “searchObj” (línea 29)
7. Finalmente buscamos e imprimimos los elementos que queramos buscar mediante el objeto “searchObj” que se ha ido construyendo. (línea 30)

Callback: Tratamiento de los elementos a posteriori

La API permite recibir una función de callback (línea 32) para el tratamiento de los elementos que retornará la API y gestionar así los resultados que se enviará al cliente.

Diseño de la Interfaz

En cuanto al diseño de la interfaz del ejemplo. El programador sólo debe encargarse de desarrollar los elementos gráficos de su interfaz. Estos elementos pueden ser:

- Dropdown: Para seleccionar los filtros de búsqueda
- Campos de fecha: Para filtrar por fechas de las publicaciones (ediciones e issues).
- Checkbox: Para indicarle a APIgesCORE que también busque dentro de los tópicos de las publicaciones.
- Mapa: Por si desea ver en un mapa los lugares de las ediciones de los congresos.
- Calendario - Agenda: Para exportar las fechas de eventos a un calendario.

Utilizando el Ejemplo

Usando el código del ejemplo de APIgesCORE (código 8.1). Tenemos los siguientes filtros básicos:

- Tipo de elemento que buscamos: **Edición**
- Ubicación de la edición: **Barcelona**
- Fecha de límite de Recepción < **06 de junio del 2013**
- **01 de junio del 2014** < Fecha de inicio < **18 de junio del 2014**

Con la siguiente configuración:

- Buscar dentro de los tópicos: **Sí**
- Ordenar por: **Fecha** de forma: **Descendiente**
- Número de resultados máximos: **50**
- Buscar a partir del resultado: **0**

Con el siguiente tratamiento a posteriori:

- Mensaje del navegador con el número de resultados devueltos

Finalmente decidimos buscar las ediciones que contengan algún tópico que hable sobre “Modelado Geométrico”, el resultado es el siguiente.

Probando gscoreAPI

* Congress: Congreso Internacional de Informática
 Order edition: 3
 Appears in (topic): Modelado Geométrico
 Category: Edición
 Publication date: 2014-06-17
 Place of edition: Barcelona
 Public Link: barcelonainternacional.com
[View in gesCORE backend](#)

* Congress: Congreso Internacional de Informática
 Order edition: 4
 Appears in (topic): Modelado Geométrico
 Category: Edición
 Publication date: 2014-06-11
 Place of edition: Barcelona
 Public Link: barcelona1.com
[View in gesCORE backend](#)

* Congress: Congreso mundial de biotecnologías
 Order edition: 4
 Appears in (topic): Modelado Geométrico
 Category: Edición
 Publication date: 2014-06-07
 Place of edition: Barcelona
 Public Link: link.com
[View in gesCORE backend](#)

* Congress: Congreso mundial de biotecnologías
 Order edition: 1
 Appears in (topic): Modelado Geométrico
 Category: Edición
 Publication date: 2014-06-03
 Place of edition: Barcelona
 Public Link: edicion1congreso1.com
[View in gesCORE backend](#)

* Congress: Congreso mundial de biotecnologías
 Order edition: 3
 Appears in (topic): Modelado Geométrico
 Category: Edición
 Publication date: 2014-06-01
 Place of edition: Barcelona
 Public Link: barcelona3.com
[View in gesCORE backend](#)

Content Provided by gesCORE

Mensaje de la página gscore.byethost11.com: x

5

Figura 8.1: Ejemplo de utilización de APIgesCORE

8.2.3 Operaciones de búsqueda de la API

Métodos de configuración de la búsqueda

Elemento	Definición
setErrorCallback (JavaScriptFunction)	Estable el método que se llamará cuando se produzca un error durante la búsqueda.
setRenderer (render)	Anula el procedimiento predeterminado que usa la API para imprimir por un procedimiento determinado por el usuario.
setRenderLocation (divtag)	Establece el nuevo identificador de la etiqueta donde se imprimirán los resultados
setCallback	Establece el método que se llamará cuando el procedimiento de búsqueda haya finalizado.

(JavaScriptFunction)	
searchAndPrint(searchObj)	Procedimiento que ejecuta la búsqueda con el objeto de búsqueda que se ha pasado por parámetro. Luego imprime los resultados en la etiqueta que tenga asignada para imprimir.
search(searchObj)	Este procedimiento ejecuta la búsqueda en backend sin mostrar los resultados, luego en la llamada de callback el usuario tendrá el control total de los resultados.

Tabla 8.1: APIgesCORE, métodos de configuración de la búsqueda

Métodos usados en la función de retorno de resultados (callback)

Estos métodos se usan en la función de callback para tener un control total sobre los resultados.

Elemento	Definición
areSearchResultsValid()	Retorna cierto o falso dependiendo si la búsqueda se ha completado correctamente o no.
getLastSearchRequest()	Retorna el objeto “searchObj” que se ha usado en la última búsqueda. “Null”, si no se ha producido ninguna búsqueda anterior
getSearchResults()	Retorna el objeto JSON de los resultados de la búsqueda.
getTotalHits()	Devuelve el número total de aciertos en la búsqueda
getPosition()	Devuelve el desplazamiento (offset) de la última búsqueda
getField(position, field)	Devuelve un solo campo (field) en una posición específica de los resultados obtenidos en la búsqueda (position)

Tabla 8.2: APIgesCORE, métodos usados en la función de retorno de resultados

Métodos usados para imprimir

Elemento	Definición
renderDocument(position)	Imprime una sola tupla de los resultados obtenidos en la búsqueda. La tupla viene determinada por la posición.
renderField(position, field)	Función para imprimir un solo campo (field) en una posición específica de los resultados obtenidos en la búsqueda (position)

Tabla 8.3: APIgesCORE, métodos usados para imprimir

Métodos condicionales

Los siguientes métodos no funcionan correctamente a menos “AreSearchResultsValid()” devuelva cierto. Esto ocurre después de realizar la búsqueda.

- `getSearchResults()`
- `renderDocument (posición)`
- `renderField (posición, campo)`
- `getField (posición, campo)`
- `getTotalHits ()` y `getPosition ()`

8.2.4 Métodos de filtrado de la API

8.2.5 Clase SearchObj

Elemento	Definición
setSearch (search)	Operación para establecer la cadena a buscar.
setType (type)	Establece el filtro de tipo de elementos a retornar. Este tipo puede ser: 'editorial', 'congress', 'edition', 'journal', 'issue', 'qualityEntity', 'topic', '*'
setObjectType (objType)	Operación para añadir más filtros del elemento que estamos buscando. Este elemento se envía al objeto principal en forma de otro objeto.
setSort (sort)	Establece el campo de ordenación de los resultados. Los valores validos son: 'title', 'acronimo', 'categoria', 'fecha', 'lugar'
setSortDirection (dir)	Establece la dirección en la que se ordenarán los resultados de la búsqueda. Valores válidos son: 'asc', 'desc'
setNumResults (num)	Establece el número de resultados a devolver. Los valores válidos son: 1-50
setOffset (num)	Establece el desplazamiento dentro de los resultados. Si offset es igual a 50, significa que se listarán los resultados desde la posición 51.
setSearchInTopics (value)	Esta operación establece si se buscará dentro de los tópicos de las publicaciones o no.

Tabla 8.4: Operaciones de filtrado de la clase SearchObj

Los valores por defecto de los filtros son:

- `setType (type)` : Por defecto son todos los tipos, `type = '*'`
- `setSort (sort)` : Por defecto se ordena por título, `sort = 'title'`
- `setSortDirection (dir)` : Por defecto la ordenación es ascendente, `dir = 'ASC'`
- `setNumResults (num)` : Por defecto el número de resultados a retornar es 50, `num = 50`
- `setSearchInTopics (value)` : Por defecto sí buscamos dentro de los tópicos de las publicaciones, `value = true`

8.2.6 Clase `editionObj`

Clase que usaremos cuando queramos filtrar por edición, esta clase se añadirá a la clase principal de búsqueda mediante la operación: “`setObjectType (objType)`”

Elemento	Definición
<code>setPlace (place)</code>	Establece el filtro del lugar donde se ha efectuado la edición.
<code>setLimReception_min (value)</code>	Filtro para indicar que las ediciones que se retornarán tendrán una fecha de recepción superior o igual que 'value'
<code>setLimReception_max (value)</code>	Filtro para indicar que las ediciones que se retornarán tendrán una fecha de recepción inferior o igual que 'value'
<code>setDefinitiveDate_min (value)</code>	Filtro para indicar que las ediciones que se retornarán tendrán una fecha definitiva superior o igual que 'value'
<code>setDefinitiveDate_max (value)</code>	Filtro para indicar que las ediciones que se retornarán tendrán una fecha definitiva inferior o igual que 'value'
<code>setStartDate_min (value)</code>	Filtro para indicar que las ediciones que se retornarán tendrán una fecha de inicio de la edición superior o igual que 'value'
<code>setStartDate_max (value)</code>	Filtro para indicar que las ediciones que se retornarán tendrán una fecha de inicio inferior o igual que 'value'
<code>setEndDate_min (value)</code>	Filtro para indicar que las ediciones que se retornarán tendrán una fecha de fin de la edición superior o igual que 'value'
<code>setEndDate_max (value)</code>	Filtro para indicar que las ediciones que se retornarán tendrán una fecha de fin de la edición inferior o igual que 'value'

Tabla 8.5: Operaciones de filtrado de la clase `editionObj`

8.2.7 Clase issueObj

Clase que usaremos cuando queramos filtrar por “issues” de una Revista, esta clase se añadirá a la clase principal de búsqueda mediante la operación: “**setObjectType(objType)**”

Elemento	Definición
setLimReception_min(value)	Filtro para indicar que las issues que se retornarán tendrán una fecha de recepción superior o igual que ‘value’
setLimReception_max(value)	Filtro para indicar que las issues que se retornarán tendrán una fecha de recepción inferior o igual que ‘value’
setDefinitiveDate_min(value)	Filtro para indicar que las issues que se retornarán tendrán una fecha definitiva superior o igual que ‘value’
setDefinitiveDate_max(value)	Filtro para indicar que las issues que se retornarán tendrán una fecha definitiva inferior o igual que ‘value’
setPublicationDate_min(value)	Filtro para indicar que las issues que se retornarán tendrán una fecha de publicación superior o igual que ‘value’
setPublicationDate_max(value)	Filtro para indicar que las issues que se retornarán tendrán una fecha de publicación inferior o igual que ‘value’

Tabla 8.6: Operaciones de filtrado de la clase issueObj

8.2.8 Valores generales por defecto de la API

Los valores por defecto que tiene la API son:

Elemento	Nombre	Valor por defecto
Etiqueta de zona de debug	<code>\divTag_debug'</code>	<code>\gescoreDebugArea'</code>
Etiqueta de zona de render	<code>\renderLocation'</code>	<code>\gescore_output'</code>
Tiempo máximo de espera de una búsqueda (milisegundos)	<code>\gescoreTimeout'</code>	<code>\40000'</code>

Tabla 8.7: Valores generales por defecto de la API

8.2.9 Comunicación y retorno de resultados

API se comunica con el sistema mediante una llamada asíncrona (AJAX) y el controlador devuelve un JSONP.

¿Qué es el JSONP?

JSONP o JSON con padding es una técnica de comunicación utilizada en los programas JavaScript para realizar llamadas asíncronas a dominios diferentes. JSONP es un método concebido para suplir la limitación de AJAX entre dominios, que únicamente permite realizar peticiones a páginas que se encuentran bajo el mismo dominio y puerto por razones de seguridad.

¿Por qué JSONP?

Como sabemos, la API se importará desde el lado del cliente, por tanto la persona que lo use en su sitio web será de un dominio diferente al del sistema. Esto hace que AJAX no pueda obtener los resultados de otro dominio por seguridad. De este modo JSONP nos permite corregir este problema de ‘cross-domain’ entre ambos sistemas.

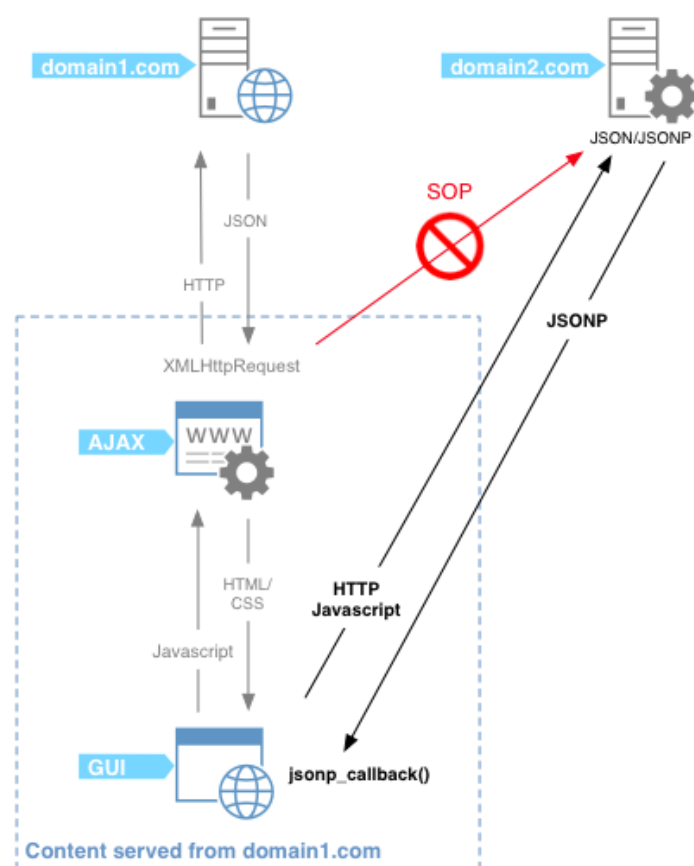


Figura 8.2: Retorno de resultados usando JSONP para solucionar problemas de cross domain

Formato de los resultados

El “response” del Web Service al cliente se hace mediante JSONP. El formato de la salida de una petición correcta es la siguiente:

```

1  gescore.Backend._requests.search[orden_busqueda].callback ({
2      "status": 200,
3      "OK": {
4          "position": [offset]
5          "returnedResults": [número de resultados retornados]
6          "totalResults": [número de resultados encontrados]
7          "results": [{
8              "id": [identificador del elemento]
9              "title": [título de la publicación]
10             "acronimo": [siglas de la entidad]
11             "categoría": [categoría del elemento]
12             "fecha": [fecha de inicio o de publicación]
13             "lugar": [lugar de la edición del congreso]
14             "link": [link del elemento]
15         }]
16     }
17 })

```

Código 8.2: Formato JSONP de la respuesta del servidor al cliente

Así pues, en el ejemplo descrito (code 8.1) el response del server es el siguiente:

```

1  gescore.Backend._requests.search0.callback({"status":200,"OK":{"p
2  osition":0,"returnedResults":5,"totalResults":5,"results":[{"id":
3  "3%Congreso Internacional de
4  Inform\u00eltica","title":"3%Congreso Internacional de
5  Inform\u00eltica%%Modelado
6  Geom\u00e9trico","acronimo":"","categoria":"Edici\u00f3n","fecha"
7  : "2014-06-
8  17","lugar":"Barcelona","linkEntidad":"barcelonainternacional.com
9  "},{ "id": "4%Congreso Internacional de
10 Inform\u00eltica","title": "4%Congreso Internacional de
11 Inform\u00eltica%%Modelado
12 Geom\u00e9trico","acronimo":"","categoria":"Edici\u00f3n","fecha"
13 : "2014-06-
14 11","lugar":"Barcelona","linkEntidad":"barcelonal.com"}, {"id": "4%
15 %Congreso mundial de biotecnologias","title": "4%Congreso mundial
16 de biotecnologias%%Modelado
17 Geom\u00e9trico","acronimo":"","categoria":"Edici\u00f3n","fecha"
18 : "2014-06-
19 07","lugar":"Barcelona","linkEntidad":"link.com"}, {"id": "1%Congr
20 eso mundial de biotecnologias","title": "1%Congreso mundial de
21 biotecnologias%%Modelado
22 Geom\u00e9trico","acronimo":"","categoria":"Edici\u00f3n","fecha"
23 : "2014-06-
24 03","lugar":"Barcelona","linkEntidad":"edicion1congreso1.com})

```

Código 8.3: Respuesta del servidor al cliente en el ejemplo

Formato de la Petición

Este es el formato de la petición que se realiza al web service:

```
1 GET urlServer/?option=com_gestorcore&controller=api&task=search
2 &format=[format de resultados]
3 &preventCache=[cadena aleatoria para prevenir el cache del nav.]
4 &search=[cadena a buscar]
5 [&filtros especificos]
6 &callback=[nombre de la función de callback]
```

Código 8.4 : formato de la petición al servidor

8.2.10 Tratamiento de errores y debugging

La API ofrece a los desarrolladores la posibilidad de hacer un seguimiento de realización sobre su código (debugging) con la finalidad de identificar los posibles errores que pueda tener en la comunicación con el servidor.

Activación del Debugger

Para activar esta herramienta el desarrollador debe de hacer 2 cosas:

Indicar a la API que deseamos activar el Debug:

```
1 debug.setDebug(true)
```

Código 8.5 : Activación del debugger de la API, lado javascript

Habilitar una zona para la salida del debug en la vista.

```
1 <div id="gescoreDebugArea"></div>
```

Código 8.6 : Activación del debugger de la API, lado html

Las operaciones que admite la clase “Debug” son las siguientes:

Elemento	Definición
setDebugDivTag(etiqueta)	Este método establece el nombre de la nueva etiqueta donde imprimirá el Debugger.
setDebug(etiqueta)	Permite personalizar la función de escritura del Debugger

Tabla 8.8: Operaciones del Debugger

Ejemplo

A continuación, mostraremos el mismo ejemplo que veníamos usando activando el Debugger de la API (línea 33 y 39).

```

2    <html>
3    <head>
4        <title>GesCORE Test/Demo Tool</title>
5        <link REL="stylesheet" TYPE="text/css"
6        href="http://gescore.byethost11.com/components/com_gestorcore/ges
        coreAPI_highlight.css"/>
7        <script type="text/javascript"
8        src="http://gescore.byethost11.com/components/com_gestorcore/gesc
        oreAPI.jsp"></script>
9        <script type="text/javascript">
10            callback = function() {
11                document.gescoreForm.searchButton.disabled=false;
12                alert (gescore.getTotalHits());
13            };
14            runSearch = function() {
15                document.gescoreForm.searchButton.disabled= true;
16                var varSearchObj = new searchObj();
17                varSearchObj.setNumResults(50);
18                varSearchObj.setOffset(0);
19                varSearchObj.setSearch
20                    (document.gescoreForm.searchString.value);
21                varSearchObj.setSearchInTopics(true);
22                varSearchObj.setSort('fecha');
23                varSearchObj.setSortDirection('desc');
24
25                var edition = new editionObj();
26                edition.setPlace('Barcelona');
27                edition.setLimReception_min('06/06/2013');
28                edition.setStartDate_min('01/06/2014');
29                edition.setStartDate_max('18/06/2014');
30                varSearchObj.setObjectType(edition);
31                gescore.searchAndPrint(varSearchObj);
32            };
33            debug.setDebug(true);
34            gescore.setCallback(callback);
35        </script>
36    </head>
37    <body>
38        <form action="" method="post" name="gescoreForm">
39            <div id="gescoreDebugArea"></div>
40            <input type="text" name="searchString"/>
41            <button onClick="runSearch()" name="searchButton"/>
42                Search
43            </button>
44            <div id="gescore_output"></div>
45        </form>
46    </body>
47    </html>

```

Código 8.7: Ejemplo de APIgesCORE utilizando el Debugger

Output del Debugger

El formato es el siguiente:

```
1 query enviada al servidor
2 dirección del GET
3 test de la comunicación Server - Cliente
4 test de los resultados recibidos en la función de callback
5 respuesta del servidor: OK = 200, ERROR = 400, etc.
```

Código 8.8 : Formato de salida del debugger de la API

En el ejemplo, en caso de realizar una petición sin problemas, el debugger devuelve lo siguiente:

Probando gescoreAPI

Search submitted: query=modelado geométrico

http://gescore.byethost11.com/index.php?

option=com_gestorcore&controller=api&task=search&format=raw&preventCache=TtqlLgCBFn5albo1RitM&search=geométrico&type=edition&sort=fecha&sortDirection=DESC&searchInTopics=true&place=Barcelona&limReception_r

Results recieved by callback...

...looking good

search: search results are OK

Checking for timedout search...

...search had already compleated

modelado geométrico

* Congress: Congreso Internacional de Informática
Order edition: 3

Figura 8.3: Salida del debugger, OK

En este caso se produjo algún error. Se ha producido un error de “Timeout”, es decir que el “request” se envió correctamente, pero que el “response” tardó más de lo permitido por el cliente.

Probando gescoreAPI

Search submitted: query=modelo geometrico

http://gescore.byethost11.com/index.php?

option=com_gestorcore&controller=api&task=search&format=raw&preventCache=trgDWCH872U3kiRadTe7&search=geometrico&type=edition&sort=fecha&sortDirection=DESCENDIENTE&searchInTopics=true&place=Barcelona&limRe

Checking for timedout search...

...Search was timed out

runSearch: search timed out

Results recieved by callback...

...Javascript just recieved a response for a search that had timed out.

modelo geometrico

Figura 8.4: Salida del debugger, Error Timeout

8.3 Gestor de Call For Papers

En cuanto a la gestión de CFP, es un requisito importante del sistema, mantener una alta integración entre elementos, es decir que a partir de un elemento concreto podamos asociar otros similares con la finalidad de mejorar la eficiencia de búsqueda.

Esto lo podemos lograr, entre otras cosas gracias a la similitud de tópicos, en la siguiente captura podemos ver que a partir de un tópico concreto, podemos saber en qué Revistas, Congresos, Issues y Ediciones aparece, y podemos navegar a cualquier tópico similar a través de un Búsqueda Rápida (ofrecida por el sistema) y obtener la misma información de su tópico similar.

GesCORE - Tópico: Interacción gráfica

Visualización ex

Apariciones del tópico

Categoría	Entidad Publicadora	Orden/Volumen	Número Issue	Link
Revista	Revista 3	-	-	link Revista
Issue	Revista 3	v1	1	link Issue
Edición	Congreso Internacional de Informática	4	-	link Edición
Edición	Congreso Internacional de Informática	3	-	link Edición
Edición	Congreso Internacional de Informática	2	-	link Edición
Edición	Congreso 2	4	-	link Edición

Tópicos Similares

- Modelado Geométrico
- Realidad virtual y realidad aumentada
- Visualización expresiva
- Visualización fotorrealista

Añadir tópico Eliminar selecc.

Figura 8.5: Captura de GesCORE, consultar tópico

Esta asociación entre elementos es una constante que se repite prácticamente en todas las vistas del proyecto. También el buscador rápido nos ayuda a hacer factible este requisito. Como podemos ver en la siguiente captura, si escribimos “barcelo” en el buscador rápido, éste nos devuelve también las ediciones de congresos que se han realizado en Barcelona.

barcelo

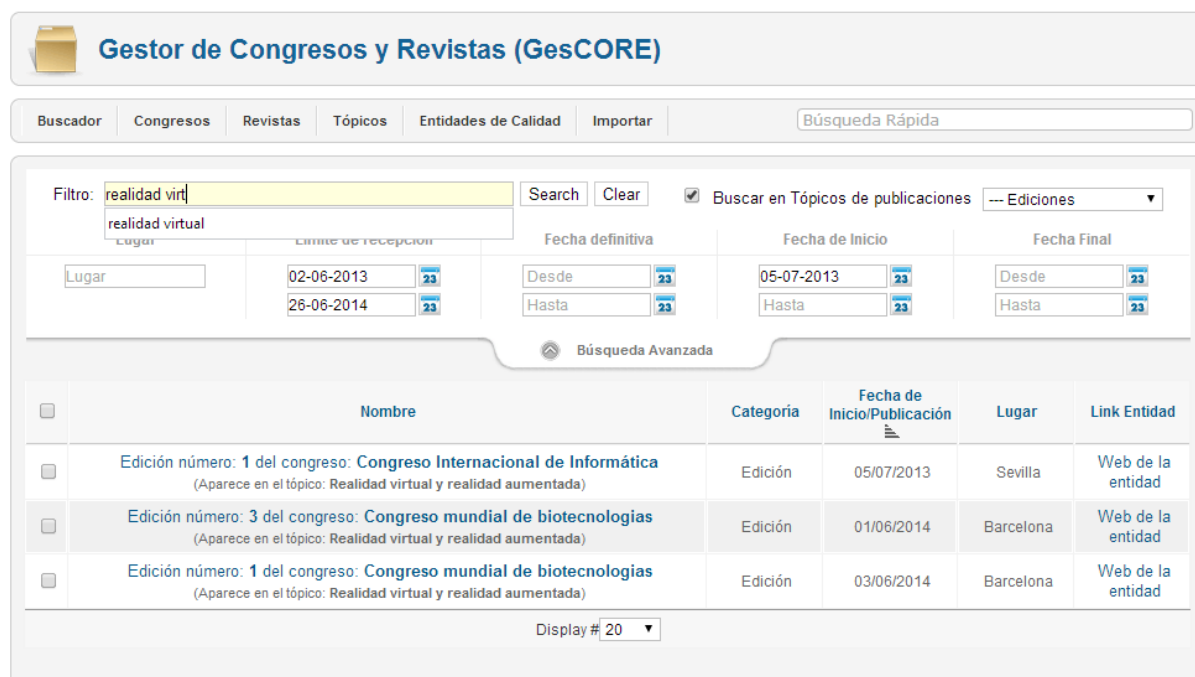
Edición

- Congreso mundial de biotecnologías - edición 1 en Barcelona
- Congreso mundial de biotecnologías - edición 3 en Barcelona
- Congreso mundial de biotecnologías - edición 4 en Barcelona
- Congreso Internacional de Informática - edición 3 en Barcelona
- Congreso Internacional de Informática - edición 4 en Barcelona

Figura 8.6: Captura de GesCORE, buscador rápido

Por supuesto, la búsqueda de elementos es importante en el proyecto debido a que la información que se maneja es muy amplia, es por ello que se invirtió mucho esfuerzo para que sea lo más potente posible.

Mediante el uso de la API antes descrita, pudimos crear un buscador que está integrado dentro de GesCORE. La siguiente captura muestra como el usuario intenta buscar dentro del sistema las ediciones que traten sobre realidad virtual y que se encuentren en un rango de fechas de recepción entre el 02/06/2013 y el 26/06/2014, también quiere que se muestren sólo aquellas con una fecha de inicio del congreso superior al 05/07/2013.



Gestor de Congresos y Revistas (GesCORE)

Buscador | Congresos | Revistas | Tópicos | Entidades de Calidad | Importar | Búsqueda Rápida

Filtro: ☒ Buscar en Tópicos de publicaciones

Lugar:

Fecha definitiva: Desde Hasta

Fecha de Inicio: Desde Hasta

Fecha Final: Desde Hasta

Búsqueda Avanzada

<input type="checkbox"/>	Nombre	Categoría	Fecha de Inicio/Publicación	Lugar	Link Entidad
<input type="checkbox"/>	Edición número: 1 del congreso: Congreso Internacional de Informática (Aparece en el tópico: Realidad virtual y realidad aumentada)	Edición	05/07/2013	Sevilla	Web de la entidad
<input type="checkbox"/>	Edición número: 3 del congreso: Congreso mundial de biotecnologías (Aparece en el tópico: Realidad virtual y realidad aumentada)	Edición	01/06/2014	Barcelona	Web de la entidad
<input type="checkbox"/>	Edición número: 1 del congreso: Congreso mundial de biotecnologías (Aparece en el tópico: Realidad virtual y realidad aumentada)	Edición	03/06/2014	Barcelona	Web de la entidad

Display # 20

Figura 8.7: Captura de GesCORE, buscador

En el proyecto no sólo es importante poder disponer de toda la información que se requiera, también es importante disponer de las estadísticas y análisis de los elementos.

En cuanto a estos conceptos, el sistema integra las “Entidades de Índices de Valoraciones”, estas entidades, como su nombre indica, se encargan de calificar las revistas y los congresos mediante sus publicaciones.

De este modo, al consultar una Entidad de Índice de Valoración era importante poder analizar gráficamente de qué forma realiza sus valoraciones esta Entidad.

En la siguiente captura podemos ver todas las valoraciones de las ediciones e issues que ha realizado la “Entidad Calidad 1” y ver gráficamente como se distribuye sus valoraciones.

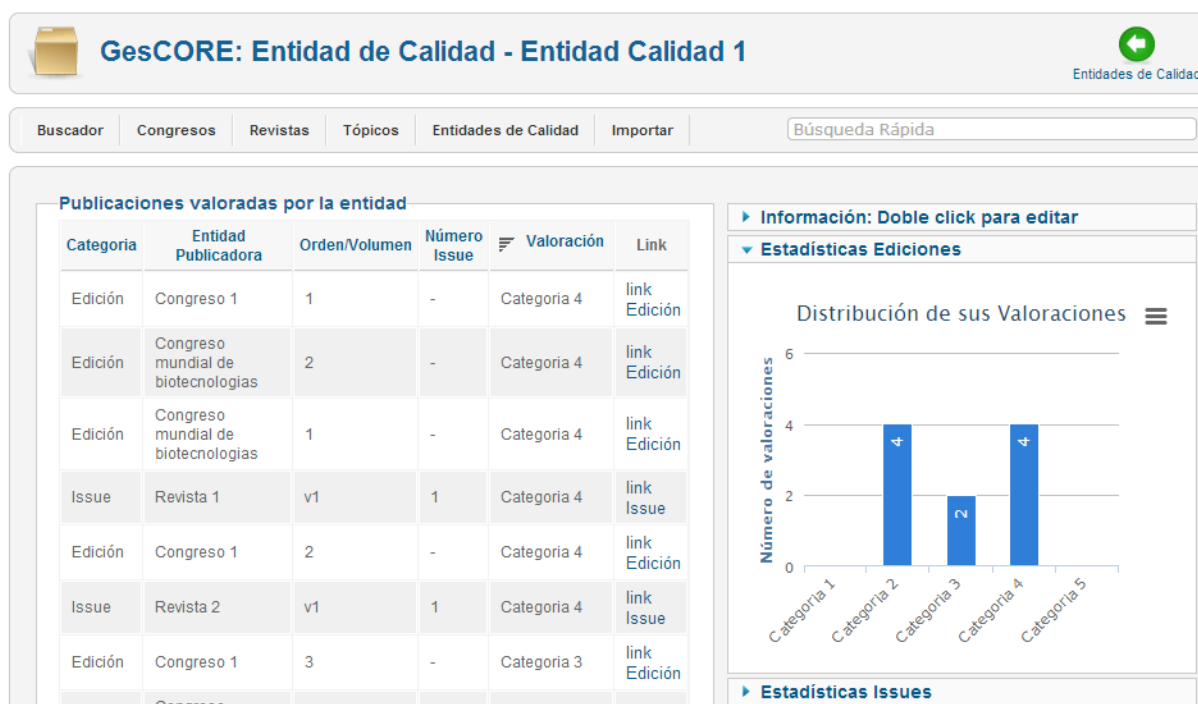


Figura 8.8: Captura de GesCORE, consultar entidad de índice de calidad

Por supuesto al ser GesCORE un gestor de publicaciones, permite dar de alta/eliminar/editar Congresos, Revistas, Ediciones, Issues, Tópicos, Entidades de Calidad, etc.

GesCORE: Nueva Issue

Guardar Cancelar

Buscador Congresos Revistas Tópicos Entidades de Calidad Importar Búsqueda Rápida

Información general Tópicos Valoraciones

Datos de la Issue

Revista*:

Tipo de edición*: ☒ Edición Simple ☐ Edición Especial

Volumen del issue*:

Número del issue* (Número más reciente del volumen: -):

Fechas de la Issue

Fecha límite de recepción:

Fecha definitiva de recepción:

Fecha publicación:

Figura 8.9: Captura de GesCORE, nueva issue

Finalmente, no queríamos que GesCORE fuese un sistema aislado, es por ello que quisimos integrarlo con Scopus, el cual dispone millones de publicaciones en su base de datos.

En la siguiente captura hemos importado de Scopus dos publicaciones que tenían alguna referencia con “robótica” y que se hayan publicado entre los años 2011 y 2014.

Se ha importado correctamente 2 publicaciones

- Force control and exponential stabilisation of one-link flexible arm (Link: [click aquí](#))
- Comparative analysis of a new 3×PPRS parallel kinematic mechanism (Link: [click aquí](#))

Filtro: robótica Search Journal - Tipo de documento - Engineering

Select All | Buscar en:

- ☒ Título - Entidad
- ☒ Palabra clave
- ☒ Autor
- ☒ ISSN
- ☒ Título - Publicación
- ☒ Referencias
- ☒ Editor
- ☒ ISBN
- ☒ Extracto
- ☒ Afiliación
- ☒ Info de la conferencia
- ☒ DOI

Búsqueda por fecha:

Tipo de búsqueda: ☐ Mes/Año ☒ Intervalo

2011 2014

-Se ha encontrado 2610 resultados-

Ordenar por: Fecha de Publicación DES 25

<input type="checkbox"/>	Título - Entidad Publicadora	Source Type	Título - Publicación	ISSN	Volumen	Nro Issue	Fecha Publicación	Primer Autor	Link Scopus
<input type="checkbox"/>	Robotics and Computer-Integrated Manufacturing	Journal	Robust nonlinear control of a planar 2-DOF parallel manipulator with redundant actuation	07365845	30	6	2014-12-12	Shang, W.	link Scopus
<input type="checkbox"/>	Robotics and Computer-Integrated Manufacturing	Journal	Adaptive control of a 3-DOF parallel manipulator considering payload handling and relevant parameter models	07365845	30	5	2014-10-01	Cazalilla, J.	link Scopus
<input checked="" type="checkbox"/>	International Journal of Control	Journal	Force control and exponential stabilisation of one-link flexible arm	13665820	87	9	2014-09-01	Endo, T.	link Scopus

Figura 8.10: Captura de GesCORE, importar desde Scopus

8.4 Método de desarrollo y seguimiento del proyecto

La metodología utilizada para el desarrollo del proyecto se basó en Scrum.

8.4.1 Scrum

Es un modelo de desarrollo ágil dividido en etapas de menor tamaño (Sprints) caracterizado por:

- Adoptar una estrategia de desarrollo incremental, en lugar de la planificación y ejecución completa del producto.
- Basar la calidad del resultado más en el conocimiento tácito de las personas en equipos autoorganizados, que en la calidad de los procesos empleados.
- Solapamiento de las diferentes fases del desarrollo, en lugar de realizar una tras otra en un ciclo secuencial o de cascada.

Roles

- Product Owner: Representa la voz del cliente. Se asegura de que el equipo Scrum trabaje de forma adecuada desde la perspectiva del negocio.
- ScrumMaster: Sirve de enlace entre el product Owner y el equipo de desarrollo, se asegura de que el proceso Scrum se utiliza como es debido.
- Equipo de desarrollo: El equipo tiene la responsabilidad de entregar el producto y gestionarlo

En cuanto a los artefactos, los más importantes son el Product Backlog y el Sprint Backlog, se tratan del documento en alto nivel para todo el proyecto y el de cualquier Sprint respectivamente.

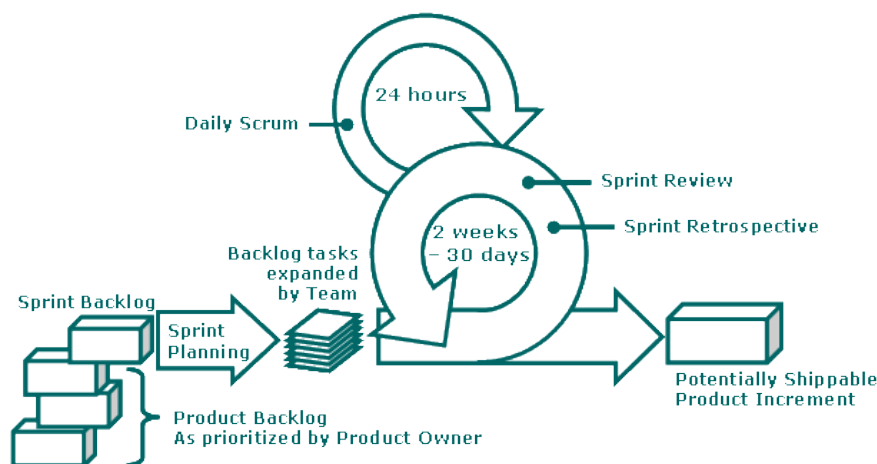


Figura 8.11: Desarrollo del proyecto utilizando Scrum

8.4.2 El desarrollo del Proyecto mediante Scrum

Identificando a los roles de Scrum en el proyecto tenemos:

- **Product Owner:** Enrique Mayol, quien es una de los integrantes del grupo de investigación Sushitos
- **ScrumMaster y equipo de desarrollo:** En este caso, el equipo de desarrollo lo conforma una sola persona, por lo que ambos roles se trata del desarrollador del proyecto.

El proyecto se ha compuesto por **18 Sprints**, y cada Sprint ha durado **2 semanas** como mucho.

El seguimiento de Scrum se realizó mediante la herramienta de gestión Trello. El gráfico que vemos a continuación se trata de uno de los Sprints del proyecto. En este gráfico podemos identificar el **Product Backlog** y el **Sprint Backlog**, así también como las tareas ya realizadas.

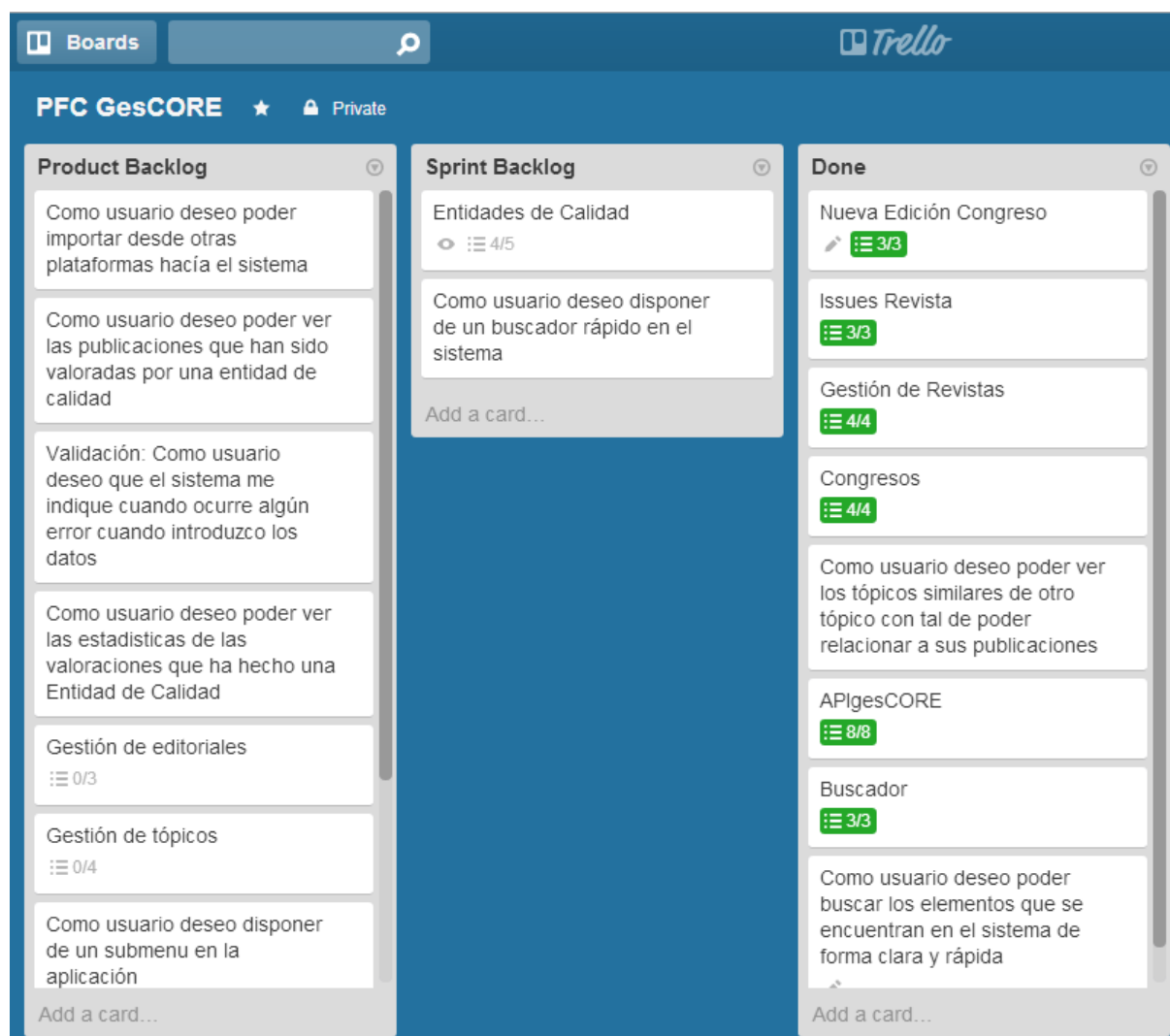


Figura 8.12: Captura de un sprint a lo largo del proyecto

9

Conclusiones

9.1 Objetivos Cumplidos	88
9.2 Futuras ampliaciones del Proyecto	89
9.3 Conclusiones Personales	89

Finalizado el proyecto, podemos llegar a las siguientes conclusiones:

9.1 Objetivos Cumplidos

GesCORE nacía como una solución a un problema, el cual venía precedido por las necesidades del grupo de investigación *Sushitos*. Estas necesidades se centraban en poder disponer de un sistema que les ayudara con la gestión de Call For Papers.

Así mismo, era muy importante para la organización, que el gestor no fuera sólo una herramienta aislada sino que pueda comunicarse con el exterior, de forma que se pueda importar publicaciones desde otras plataformas. Este objetivo se cumplió mediante la herramienta de importación desde Scopus.

Con esto pretendíamos cubrir el objetivo principal, que era informatizar su sistema de gestión de Call For Paper.

Como objetivo secundario, era deseable por la organización poder disponer de la información almacenada en GesCORE en el lado frontend de su web. Decidimos ampliar este objetivo para que cualquier persona pueda disponer del contenido de GesCORE en su web, blog, o en el sitio que desee. Este objetivo lo hemos cumplido con la creación de la API y la adaptación del servidor como un web service con arquitectura RESTful.

9.2 Futuras ampliaciones del Proyecto

Creemos que el Gestor de Congresos y Revistas sólo es la punta del Iceberg, al tener un web service con arquitectura RESTful nos invita a pensar que el proyecto puede crecer ampliando los clientes. En principio la API está pensada para un cliente web, pero en un futuro estos clientes pueden ampliarse a los dispositivos móviles.

Es decir, aprovechando la arquitectura, podríamos disponer de los elementos almacenados en GesCORE en nuestro móvil, también podríamos crear alertas para que nos avisen cuando se acerca la fecha límite de aceptación (deadline) de publicaciones de una revista científica, podríamos crear un mapa interactivo que nos muestre las ubicaciones de las ediciones de los congresos que se están celebrando actualmente, y muchas más posibilidades.

En cuanto al Gestor, este también puede ampliarse añadiendo la posibilidad de importar desde otras plataformas parecidas a Scopus, como DBLP (Digital Bibliography & Library Project), el cual es un sitio web que posee un enorme repositorio bibliográfico de artículos relacionados con ciencias de la computación.

9.3 Conclusiones Personales

Este proyecto ha resultado un reto para mí desde el punto de vista de desarrollador, ya que si bien es cierto que actualmente estoy trabajando en desarrollo de aplicaciones web, hasta ahora no había desarrollado un proyecto de esta magnitud en solitario. A su vez también ha sido una motivación personal el poder conocer nuevas tecnologías y nuevas plataformas en el desarrollo de aplicaciones web. Entre otras cosas, me ha permitido conocer un sistema de gestión de contenidos robusto y con una amplia comunidad de desarrolladores como es Joomla.

Como desarrollador también me ha permitido ampliar mis conocimientos sobre tecnologías que ya conocía como desarrollo utilizando MVC, PHP, jQuery, arquitecturas REST, AJAX, etc.

En lo personal, estoy contento de haber escogido este proyecto, de diseñarlo, desarrollarlo y de ver como se materializaban el un proyecto real de software las ideas que tenía en la mente.

A

Instalación de GesCORE

Antes de nada, es requisito necesario e importante disponer de una web realizada sobre Joomla, la versión recomendada es la 1.6.

GesCORE es un componente de Joomla, por lo que su instalación es recomendable hacerlo por la propia interfaz de Joomla.

Para instalar GesCORE en un sitio web se ha de seguir lo siguiente:

1. Asegurarse que no existan las siguientes tablas en el servidor:

- [xxx_]valoracionIssue
- [xxx_]valoracionEdicion
- [xxx_]entidadCategoria
- [xxx_]topicoSimilar
- [xxx_]issueTopico
- [xxx_]revistaTopico
- [xxx_]edicionTopico
- [xxx_]congresoTopico
- [xxx_]categoria
- [xxx_]entidadIndiceCalidad
- [xxx_]issue
- [xxx_]revista
- [xxx_]topico

- [xxx_]edicion
- [xxx_]congreso
- [xxx_]editorial

[xxx_] = prefijo que asigna el desarrollador a Joomla en el momento de su instalación. Joomla creará todas sus tablas anteponiendo este prefijo a sus nombres.

2. Conectarse al sitio web como administrador, para esto, se tiene que dirigir al administrador de Joomla. Generalmente: <http://mysite.com/administrator>
3. Abrimos el gestor de extensiones de Joomla ubicada en: “Extensions” -> “Extension Manager”.

Extension Manager: Install

Install | Update | Manage | Discover | Warnings

Upload Package File

Package File com_gescore.zip

Install from Directory

Install Directory

Install from URL

Install URL

4. Clickar sobre “Seleccionar archivo” y escoger el archivo “.zip” donde se encuentra el proyecto.
5. (Alternativa) Mover com_gescore.zip a una carpeta e indicarle la carpeta al “Gestor de Extensiones”
6. Clickar el botón “Upload & Install” y esperar a que Joomla instale el nuevo componente.
7. Comprobar que se ha instalado el componente correctamente y que se han creado las tablas ya mencionadas. Si todo ha ido correctamente se tiene que haber creado un nuevo elemento en el menú “Componentes” de nombre “Gescore”.

Lista de Código Fuente

Código 8.1: Ejemplo de APIgesCORE	69
Código 8.2: Formato JSONP de la respuesta del servidor al cliente	78
Código 8.3: Respuesta del servidor al cliente en el ejemplo	78
Código 8.4 : formato de la petición al servidor	79
Código 8.5 : Activación del debugger de la API, lado javascript	79
Código 8.6 : Activación del debugger de la API, lado html.....	79
Código 8.7: Ejemplo de APIgesCORE utilizando el Debugger	80
Código 8.8 : Formato de salida del debugger de la API.....	81

Referencias

- [1] Sushitos. (accedido el día 16 de mayo del 2014). http://www.essi.upc.edu/recerca-es/grupos-de-investigacion-1/sushitos?set_language=es
- [2] Joomla (accedido el día 17 de mayo del 2014)
<http://en.wikipedia.org/wiki/Joomla>
- [3] Diseño lógico y físico de la BBDD: (accedido el día 07 de junio de 2014):
<http://irfeyal.wordpress.com/base-de-datos/modelamiento-de-bdd/>
- [4] Modelo de acceso en BBDD: (accedido el día 07 de junio de 2014)
[http://www.fing.edu.uy/inco/cursos/tagsi/TAGSI-DisBD\(v2007\).pdf](http://www.fing.edu.uy/inco/cursos/tagsi/TAGSI-DisBD(v2007).pdf)
- [5] Patrones de diseño: (accedido el día 02 de junio de 2014)
http://es.wikipedia.org/wiki/Patr%C3%B3n_de_dise%C3%B1o
- [6] JSONP: (accedido el día 09 de junio de 2014) <http://es.wikipedia.org/wiki/JSONP>
- [7] REST y RESTful (accedido el día 06 de junio de 2014)
http://en.wikipedia.org/wiki/Representational_state_transfer
- [8] Cross-Doman en Ajax (accedido el día 06 de junio de 2014) <http://jquery-howto.blogspot.com.es/2013/09/jquery-cross-domain-ajax-request.html>
- [9] Scrum (accedido el día 10 de junio de 2014)
[http://en.wikipedia.org/wiki/Scrum_\(software_development\)](http://en.wikipedia.org/wiki/Scrum_(software_development))
- [10] Integración de un Web Service en Android (accedido el día 11 de junio de 2014)
<http://amatellanes.wordpress.com/2014/01/02/android-integrando-un-restful-web-service-en-android/>

Herramientas Utilizadas

Las herramientas utilizadas para la realización de este documento son:

[1] Diagramas de secuencia: <https://www.websequencediagrams.com>

[2] Diagramas de caso de uso: <https://www.draw.io>

[3] Diagramas de arquitectura: <https://gliffy.com>

[4] Mockups: <https://webdemo.balsamiq.com>

[5] Edición de imágenes: Adobe Photoshop

[6] Diagramas de diseño de la Base de datos: Visual Paradigm