



UNIVERSIDADE DE  
VIGO

ESCOLA SUPERIOR DE ENXEÑARÍA INFORMÁTICA

Memoria do Traballo de Fin de Grao que presenta

**D. Marcos Núñez Celeiro**

para a obtención do Título de Graduado en Enxeñaría Informática

**Xestor de Grupos de Investigación**

Xullo, 2017



**Traballo de Fin de Grado Nº: EI16/17-069**

**Titor/a:** Miguel Reboiro Jato

**Área de coñecemento:** Linguaxes e Sistemas Informáticos

**Departamento:** Informática

2017



# Índice

1. Introducción.....	1
2. Objetivos.....	1
3. Resumen.....	2
3.1. Solución aportada.....	2
3.2. Cumplimiento de los objetivos.....	2
3.3. Metodología de desarrollo.....	3
4. Planificación y seguimiento.....	4
4.1. Planificación inicial.....	4
4.2. Seguimiento.....	5
4.2.1. Arquitectura.....	6
4.3. Arquitectura servidor.....	7
4.3.1. Características.....	7
4.3.2. Flujo de ejecución.....	7
4.4. Arquitectura cliente.....	8
4.4.1. Características.....	8
5. Tecnologías y productos de terceros.....	9
5.1. Introducción.....	9
5.2. Tecnologías en servidor.....	9
5.3. Tecnologías en cliente.....	10
5.4. Tecnologías para el desarrollo.....	11
6. Análisis de requisitos.....	12
6.1. Identificación de actores.....	12
6.2. Diagrama de casos de uso.....	12
6.2.1. Diagrama de casos de uso de gestión de proyectos.....	13
6.2.2. Diagrama de casos de uso de gestión de congresos.....	13
6.2.3. Diagrama de casos de uso de gestión de revistas.....	14
6.2.4. Diagrama de casos de uso de gestión de investigadores.....	14
6.2.5. Diagrama de casos de uso de gestión de autores.....	15
6.2.6. Requisitos no funcionales.....	15
6.3. Descripción de casos de uso.....	16
6.3.1. Caso de uso identificarse.....	16
6.3.2. Caso de uso salir.....	16
6.3.3. Caso de uso consultar estado global.....	16
6.3.4. Caso de uso de gestión de investigadores: consultar investigadores.....	17
6.3.5. Caso de uso de gestión de investigadores: consultar investigador.....	17
6.3.6. Caso de uso de gestión de investigadores: añadir investigador.....	17
6.3.7. Caso de uso de gestión de investigadores: actualizar investigador.....	18
6.3.8. Caso de uso de gestión de investigadores: eliminar investigador.....	18
6.3.9. Caso de uso de gestión de congresos: consultar lista de congresos.....	19
6.3.10. Caso de uso de gestión de congresos: consultar congreso.....	19
6.3.11. Caso de uso de gestión de congresos: añadir congreso.....	20
6.3.12. Caso de uso de gestión de congresos: actualizar congreso.....	21
6.3.13. Caso de uso de gestión de congresos: eliminar congreso.....	21
7. Diseño del software.....	22
7.1. Diseño estático.....	22
7.1.1. Diseño estático en el servidor.....	22
7.1.1.1. Convenciones.....	22
7.1.1.2. Diagrama de clases de identificación y salida del sistema.....	23

7.1.1.3. Diagrama de clases de gestión de investigadores.....	23
7.1.1.4. Diagrama de clases de gestión de congresos.....	24
7.1.2. Diseño estático en el cliente.....	25
7.1.2.1. Diagrama de paquetes de la aplicación cliente.....	25
7.1.2.2. Diagrama de clases de los componentes de la vista.....	25
7.1.2.2.1. Componentes base.....	26
7.1.2.2.2. Componentes utilidad.....	26
7.1.2.2.3. Componentes de listas:.....	27
7.1.2.2.4. Componentes de detalle:.....	27
7.2. Diseño dinámico.....	27
7.2.1. Convenciones.....	27
7.2.1.1. Enrutamiento.....	27
7.2.1.2. Componentes de la vista.....	28
7.2.1.3. Creación de diagramas de secuencia auxiliares.....	28
7.2.1.4. Convenciones adicionales.....	28
7.2.1.5. Diagrama de secuencia de consultar estado global.....	29
7.2.1.6. Diagrama de secuencia de consultar lista de investigadores.....	30
7.2.1.7. Diagrama de secuencia de añadir investigador.....	30
7.2.1.8. Diagrama de secuencia de actualizar congreso.....	31
7.2.1.9. Diagrama de secuencia de eliminar congreso.....	32
8. Gestión de datos e información.....	32
8.1. Diagrama Entidad-Relación.....	32
8.2. Diccionario de datos.....	33
8.2.1. Entidad Usuario.....	33
8.2.2. Entidad Investigador.....	33
8.2.3. Entidad Autor.....	33
8.2.4. Entidad Proyecto.....	33
8.2.5. Entidad Revista.....	33
8.2.6. Entidad Congreso.....	34
8.3. Diagrama de tablas.....	34
9. Pruebas.....	36
9.1. Escenario de identificación.....	36
9.2. Escenario de salida del sistema.....	36
9.3. Escenario de creación y actualización de un investigador.....	36
9.4. Eliminación de un investigador.....	36
9.5. Creación/actualización de un autor.....	36
9.6. Eliminación de un autor.....	36
9.7. Inserción o modificación de un proyecto.....	36
9.8. Eliminación de un proyecto.....	37
9.9. Inserción o modificación de un congreso.....	37
9.10. Eliminación de un congreso.....	37
9.11. Inserción o modificación de una publicación en revista.....	37
9.12. Eliminación de una publicación en revista.....	37
9.13. Visualización de calendarios.....	37
10. Manuales de usuario.....	37
10.1. Manual de instalación.....	37
10.1.1. Requisitos mínimos.....	37
10.1.2. Instalación.....	38
10.2. Manual de utilización.....	38
11. Principales aportaciones.....	45

11.1. Repositorio centralizado de información.....	45
11.2. Simplicidad.....	45
11.3. Fluidez.....	45
11.4. Aplicación preparada para crecer.....	46
12. Conclusiones.....	46
12.1. Conclusiones personales.....	46
12.2. Conclusiones técnicas.....	46
13. Vías de trabajo futuro.....	47
13.1. Mejoras funcionales.....	47
13.1.1. Gestión de publicaciones en revista.....	47
13.1.2. Gestión de congresos.....	47
13.1.3. Integración con Google Calendar.....	47
13.2. Mejoras de usabilidad.....	47
13.3. Mejoras técnicas.....	48
14. Referencias.....	49



# Índice de Figuras

Figura 1: Fases del proceso unificado.....	3
Figura 2: Lista de tareas a realizar durante el desarrollo del proyecto.....	5
Figura 3: Ejecución final de la planificación.....	6
Figura 4: Arquitectura MVC de la aplicación en el servidor.....	8
Figura 5: Flujo de ejecución de la aplicación cliente.....	9
Figura 6: Diagrama de casos de uso global.....	13
Figura 7: Diagrama de casos de uso de gestión de proyectos.....	13
Figura 8: Diagrama de casos de uso de gestión de congresos.....	14
Figura 9: Diagrama de casos de uso de gestión de revistas.....	14
Figura 10: Diagrama de casos de uso de gestión de investigadores.....	15
Figura 11: Diagrama de casos de uso de gestión de autores.....	15
Figura 12: Diagrama de paquetes del backend.....	22
Figura 13: Diagrama de clases de identificación y salida del sistema.....	23
Figura 14: Diagrama de clases de gestión de investigadores.....	24
Figura 15: Diagrama de clases de gestión de congresos.....	24
Figura 16: Diagrama de paquetes simple de la aplicación cliente.....	25
Figura 17: Diagrama de clases de los componentes del cliente.....	26
Figura 18: Diagrama de secuencia de consultar estado global.....	29
Figura 19: Diagrama de secuencia de ver lista de investigadores.....	30
Figura 20: Diagrama de secuencia de añadir investigador.....	30
Figura 21: Diagrama de secuencia de actualizar congreso.....	31
Figura 22: Diagrama de secuencia de eliminar congreso.....	32
Figura 23: Diagrama entidad-relación.....	32
Figura 24: Modelo de datos.....	35
Figura 25: Pantalla de identificación.....	38
Figura 26: Pantalla de calendario de actividades.....	39
Figura 27: Pantalla de agenda.....	39
Figura 28: Pantalla de lista de investigadores.....	40
Figura 29: Pantalla de detalle de un investigador.....	40
Figura 30: Pantalla de lista de autores.....	41
Figura 31: Pantalla de detalle de un autor.....	41
Figura 32: Pantalla de lista de proyectos.....	42
Figura 33: Pantalla de detalle de un proyecto.....	42
Figura 34: Pantalla de lista de revistas.....	43
Figura 35: Pantalla de detalle de una revista.....	43
Figura 36: Pantalla de lista de congresos.....	44
Figura 37: Pantalla de detalle de un congreso.....	44
Figura 38: Pantalla de formulario con errores en la edición de una revista.....	45



## Índice de Tablas

Tabla 1: Estimación del proyecto por fases e iteraciones.....	4
Tabla 2: Descripción caso de uso "identificarse".....	16
Tabla 3: Descripción de caso de uso "salir".....	16
Tabla 4: Descripción de caso de uso "consultar estado global".....	17
Tabla 5: Descripción de caso de uso "consultar investigadores".....	17
Tabla 6: Descripción de caso de uso "consultar investigador".....	17
Tabla 7: Descripción de caso de uso "añadir investigador".....	18
Tabla 8: Descripción de caso de uso "actualizar investigador".....	18
Tabla 9: Descripción de caso de uso "eliminar investigador".....	19
Tabla 10: Descripción de caso de uso "consultar lista de congresos".....	19
Tabla 11: Descripción de caso de uso "consultar congreso" .....	20
Tabla 12: Descripción de caso de uso "añadir congreso".....	20
Tabla 13: Descripción de caso de uso "actualizar congreso".....	21
Tabla 14: Descripción de caso de uso "eliminar congreso".....	21



## 1. Introducción

Dentro de las universidades el Personal Docente e Investigador (PDI) suele organizarse en grupos de investigación, bajo los cuales se agrupan varios investigadores que trabajan en un mismo campo. Estos grupos de investigación facilitan la colaboración entre estos investigadores, al proporcionar recursos compartidos (p.ej. laboratorios, material para investigación, etc.) y facilitar la realización de distintas gestiones (p.ej. petición de proyectos, gestión de presupuestos, etc.).

Los grupos de investigación están dirigidos por uno o varios Investigadores Principales (IP), que son los responsables de coordinar el grupo, normalmente, marcando el rumbo del grupo dentro de su campo de investigación. Además, es habitual que los IP sean los encargados de realizar las principales gestiones del grupo, aunque en ciertas ocasiones estas funciones pueden estar delegadas en, por ejemplo, un secretario.

La labor de coordinación de un grupo de investigación es un proceso complejo y que puede llegar a consumir mucho tiempo. Esto se debe a que la cantidad de acciones que puede llevar a cabo un investigador, individual o grupalmente, es muy amplia. Así, por ejemplo, algunas de las acciones habituales son la solicitud y realización de proyectos de investigación, publicación de artículos en revistas y congresos, solicitud de acreditaciones, solicitud de becas de investigación, realización de trabajos fin de grado o máster, realización de estancias, etc. Además, estas acciones requieren de un seguimiento temporal, que permita saber en cada momento el estado de la misma.

Con el fin de facilitar la gestión y coordinación de los grupos de investigación se ha realizado la creación de una herramienta informática que automatiza parte del proceso. Esta herramienta sirve como un repositorio centralizado en el que se recoge toda la información de las acciones que están realizando los distintos miembros del grupo de investigación. De este modo, el coordinador puede, fácilmente, tener una visión global del estado del grupo de investigación o de un investigador en concreto. Además, permitiendo a los propios investigadores la introducción de información, es posible distribuir el trabajo de gestión entre los miembro del grupo.

La solución aportada permite a los investigadores, de manera sencilla, consultar información sobre el estado del grupo de investigación y sus actividades, así como modificar la información de sus miembros. Mediante el acceso a la aplicación, un usuario puede ver de manera simple, mediante calendarios o agendas, los eventos en los que participarán los miembros de ese grupo así como gestionar su información.

Esta herramienta, por lo tanto, puede resultar útil para el grupo de investigación debido a que permite la consulta unificada de los datos del grupo de manera sencilla y en un único lugar. Permite, además, modificar los datos almacenados en la aplicación en conjunto con otros miembros del grupo, manteniendo toda la información actualizada y sin conllevar un trabajo de gestión excesivo gracias a la colaboración entre los participantes. Todo esto convierte el proceso de gestión de y coordinación del grupo en una tarea más simple.

## 2. Objetivos

El objetivo principal de este trabajo es la creación de una herramienta para la gestión de grupos de investigación de universidades.

Para poder alcanzar este objetivo principal, ha sido necesario alcanzar los siguientes objetivos parciales:

- **Gestión de investigadores:** poder gestionar la información de los investigadores que forman parte del grupo de investigación. El usuario debe ser capaz de consultar los datos de estos tanto de forma global, viendo un listado, como en detalle. También debe poder añadir, actualizar y borrar los datos de los miembros del grupo.
- **Gestión de proyectos:** poder gestionar los proyectos del grupo de investigación. Esta gestión permite hacer un seguimiento del proyecto y de los distintos estados por los que pasa desde la convocatoria de las subvenciones hasta la justificación del mismo. El usuario debe ser capaz de consultar los datos, añadir nuevos proyectos,

actualizarlos o eliminarlos de forma sencilla. En el proceso, tanto de actualización, como de añadido de nuevos proyectos debe ser posible la gestión de los investigadores que participan en estos, pudiendo asociar los ya existentes al proyecto que se esté gestionando.

- **Gestión de publicaciones en revista:** poder gestionar las publicaciones en revista. Esta gestión permite hacer un seguimiento de la publicación y de los distintos estados por los que pasa desde la redacción del artículo hasta la publicación o rechazo del mismo. El usuario debe poder consultar los datos, añadir nuevas publicaciones, actualizarlas y eliminarlas. En las funcionalidades de añadido y edición, debe poder asociarse o quitar autores existentes a cada una de las publicaciones.
- **Gestión de publicaciones en congresos:** poder gestionar las publicaciones en congresos. Esta gestión permite hacer un seguimiento de la publicación y de los distintos estados por los que pasa desde la redacción del artículo hasta la presentación o rechazo del mismo. El usuario debe poder consultar los datos, añadir nuevas publicaciones, actualizarlas y eliminarlas. En las funcionalidades de añadido y edición, debe poder asociarse o quitar investigadores existentes de cada una de las publicaciones en congresos.
- **Visualización general del estado del grupo de investigación:** el sistema debe proporcionar dos tipos de visualización que permiten conocer el estado actual del grupo de investigación. Una visualización en forma de calendario de actividades y otra en forma de agenda.

### 3. Resumen

#### 3.1. Solución aportada

Para la gestión de los grupos de investigación se ha creado una herramienta que permite a los investigadores de un grupo consultar de forma fácil cada uno de los trabajos en los que están involucrados, así como realizar consultas sencillas del estado de cada proyecto o publicación.

Para la realización de esta herramienta se han realizado dos aplicaciones independientes entre sí. Por un lado, se presenta una Interfaz de Programación de Aplicaciones (API, Application Programming Interface) mediante la cual se podrá acceder a los datos almacenados, y por otro se presenta una interfaz de usuario (UI, User Interface) sobre la cual se accede a las distintas partes de la aplicación de forma sencilla para el usuario final.

En la solución aportada, mediante la interfaz de usuario, los investigadores del grupo pueden identificarse como miembros y acceder a las distintas secciones presentadas en los objetivos, pudiendo también modificar los datos de los proyectos, congresos o revistas de las que son autores, así como los datos de su perfil de investigador. Se dispone también de una sección para la visualización del estado del grupo de investigación, tanto en forma de calendario de actividades como de agenda. En esta sección se presentan las fechas de inicio y fin de los proyectos y congresos próximos.

#### 3.2. Cumplimiento de los objetivos

Con el desarrollo de esta aplicación se han logrado cumplir los objetivos marcados anteriormente. Sin embargo, como único punto negativo, no se ha logrado la profundidad deseada sobre la gestión de publicaciones en revista.

Aunque en esta sección se permite, al igual que en las demás, la visualización y edición de publicaciones, no se trabaja con los conceptos de JCR (Journal Citation Reports) y sus factores de impacto por año y categoría, los cuales exigían una complejidad extra tanto en el API como en la UI.

El no lograr la profundidad deseada en esta sección se ha debido a que el trabajo que se ha tenido que realizar para la comprensión y utilización de las tecnologías seleccionadas, tanto para la implementación de la vista y el código de servidor, como para la mejora de la calidad del proyecto global, ha supuesto una complejidad excesiva respecto a lo esperado, especialmente en el caso de la comprensión de las tecnologías utilizadas en el servidor.

### 3.3. Metodología de desarrollo

Para la realización del proyecto se ha utilizado el Proceso Unificado (UP, Unified Process) [1] en conjunción con el Lenguaje Unificado de Modelado (UML, Unified Modeling Language) [2].

El proceso unificado se caracteriza por estar dirigido por casos de uso, estar centrado en la arquitectura y ser iterativo e incremental. Estas características convierten a este marco de desarrollo en flexible y le permite adaptarse fácilmente a las necesidades de cualquier desarrollador o equipo.

El UP ha resultado especialmente útil en este proyecto debido a que se trata de un tipo de desarrollo individual en conjunción con el cliente. Esta metodología, al ser iterativa e incremental, aporta una gran flexibilidad permitiendo adaptarla a las necesidades de un único desarrollador mediante un enfoque más ágil para la organización de las tareas.

En la Figura 1 se puede visualizar gráficamente el desarrollo de un proyecto con UP. Como se puede observar, el proceso está dividido en cuatro fases principales:

- **Inicio:** se realiza un análisis global del sistema. Se obtienen los requisitos del proyecto y se determina si es viable así como que recursos serán necesarios. En esta fase pueden desarrollarse ya algunos casos de uso globales así como un pequeño principio de la arquitectura e implementación.
- **Elaboración:** en esta fase se describen en detalle todos los casos de uso del sistema y se diseña gran parte de la arquitectura. Aparte de esto, se siguen refinando el trabajo de la fase de inicio.
- **Construcción:** la principal tarea de esta fase es la implementación del sistema. Se siguen refinando requisitos y arquitectura así como analizando riesgos y realizando las demás tareas de las fases anteriores, aunque en menor medida.
- **Transición:** se refina el trabajo de las fases anteriores y se termina la realización de todos los artefactos. La meta es que el producto satisfaga a los interesados. Esta última fase se inicia normalmente con una beta de la aplicación y se finaliza con la puesta en producción del sistema final.

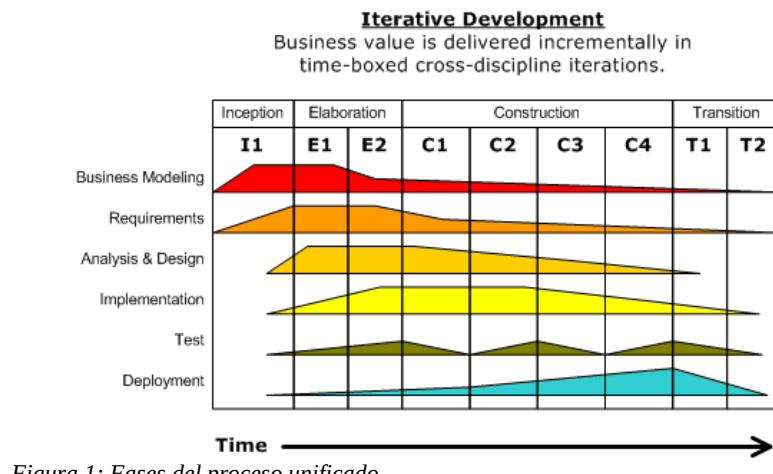


Figura 1: Fases del proceso unificado.

En esta aplicación ha habido una gran cantidad de dificultades iniciales con la puesta a punto de las tecnologías relacionadas, tanto con la propia aplicación, como la mejora de calidad del código. Este tipo de dificultades han provocado una fase de inicio más amplia de lo habitual.

## 4. Planificación y seguimiento

### 4.1. Planificación inicial

El desarrollo del proyecto ha sido planificado para llevarlo a cabo de lunes a domingo, con una dedicación esperada de 4 horas de trabajo al día incluyendo los fines de semana. Esto resulta en un total de 75 días empezando el día 1 de Marzo de 2017 y finalizando el 2 de Junio de 2017.

El punto más crítico a tener en cuenta en esta aplicación es el gran número de tecnologías que se utilizan, *frameworks* complejos en la parte del servidor y una gran cantidad de librerías en la parte del cliente. Esto aumenta especialmente la posibilidad de desvíos en las estimaciones.

La estimación realizada aquí deja un espacio de tiempo considerable para que la posibilidad que existe de que estos desvíos se produzcan no provoque retrasos en la entrega final.

En la 1 se muestra un desglose general de este trabajo dividido por fases. En la Figura 2 se muestra una tabla mucho más detallada y que contiene todas las tareas a desarrollar.

Dedicación semanal prevista	28 horas por semana
Fase	Estimación temporal (en días)
Inicio	11
Elaboración	11
Construcción	44
• Primera iteración	13
• Segunda iteración	17
• Tercera iteración	14
Transición	9
<b>TOTAL DEL PROYECTO</b>	<b>75</b>

Tabla 1: Estimación del proyecto por fases e iteraciones

<b>« Gestor de grupos de investigación</b>	<b>75 días</b>	<b>01/03/17</b>	<b>02/06/17</b>
<b>« Inicio</b>	<b>11 días</b>	<b>01/03/17</b>	<b>14/03/17</b>
Estudio del alcance	2 días	01/03/17	03/03/17
Análisis y priorización de requisitos	2 días	03/03/17	05/03/17
Primeros diagramas	1 día	06/03/17	07/03/17
Estudio de tecnologías	4 días	07/03/17	12/03/17
Implementación inicial de la arquitectura	1 día	12/03/17	13/03/17
Pruebas	1 día	13/03/17	14/03/17
<b>« Elaboración</b>	<b>11 días</b>	<b>14/03/17</b>	<b>28/03/17</b>
Refinamiento de los requisitos	2 días	14/03/17	17/03/17
Diseño	5 días	17/03/17	23/03/17
Implementación	2 días	23/03/17	25/03/17
Pruebas	1 día	26/03/17	27/03/17
Refactorización	1 día	27/03/17	28/03/17
<b>« Construcción</b>	<b>44 días</b>	<b>28/03/17</b>	<b>22/05/17</b>
<b>« Primera iteración</b>	<b>13 días</b>	<b>28/03/17</b>	<b>13/04/17</b>
Drefinamiento de requisitos	1 día	28/03/17	29/03/17
Mejora del diseño	1 día	29/03/17	30/03/17
Implementación	8 días	31/03/17	09/04/17
Pruebas	1 día	10/04/17	11/04/17
Refactorización	2 días	11/04/17	13/04/17
<b>« Segunda iteración</b>	<b>17 días</b>	<b>13/04/17</b>	<b>04/05/17</b>
Refinamiento de los requisitos	2 días	13/04/17	16/04/17
Mejora del diseño	2 días	16/04/17	18/04/17
Implementación	10 días	18/04/17	01/05/17
Pruebas	1 día	01/05/17	02/05/17
Refactorización	2 días	02/05/17	04/05/17
<b>« Tercera iteración</b>	<b>14 días</b>	<b>05/05/17</b>	<b>22/05/17</b>
Refinamiento de los requisitos	1 día	05/05/17	06/05/17
Mejora del diseño	2 días	06/05/17	08/05/17
Implementación	8 días	08/05/17	18/05/17
Pruebas	2 días	18/05/17	21/05/17
Refactorización	1 día	21/05/17	22/05/17
<b>« Transición</b>	<b>9 días</b>	<b>22/05/17</b>	<b>02/06/17</b>
Refactorización	2 días	22/05/17	24/05/17
Refinamiento de las pruebas	2 días	25/05/17	27/05/17
Actualización de la documentación	3 días	27/05/17	31/05/17
Manual de usuario	2 días	31/05/17	02/06/17

Figura 2: Lista de tareas a realizar durante el desarrollo del proyecto

## 4.2. Seguimiento

El desarrollo de la aplicación ha llevado más tiempo de lo que cabría esperar en un inicio. Debido al desconocimiento de una gran cantidad de tecnologías utilizadas el tiempo en la fase de inicio ha aumentado por necesidad de estudiarlas más detenidamente.

Además, en las fases de implementación también se han encontrado ciertos problemas o ha habido que investigar el uso o aplicación de ciertas tecnologías, lo cuál ha provocado que el tiempo de desarrollo del proyecto se desfase varios días.

En la Figura 3 se muestra de nuevo el desglose de tareas una vez aplicados estos desvíos.

<b>▲ Gestor de grupos de investigación</b>	<b>85 días</b>	<b>01/03/17</b>	<b>15/06/17</b>
<b>▲ Inicio</b>	<b>15 días</b>	<b>01/03/17</b>	<b>19/03/17</b>
Estudio del alcance	2 días	01/03/17	03/03/17
Análisis y priorización de requisitos	2 días	03/03/17	05/03/17
Primeros diagramas	1 día	06/03/17	07/03/17
Estudio de tecnologías	8 días	07/03/17	17/03/17
Implementación inicial de la arquitectura	1 día	17/03/17	18/03/17
Pruebas	1 día	18/03/17	19/03/17
<b>▲ Elaboración</b>	<b>11 días</b>	<b>19/03/17</b>	<b>02/04/17</b>
Refinamiento de los requisitos	2 días	19/03/17	22/03/17
Diseño	5 días	22/03/17	28/03/17
Implementación	2 días	28/03/17	30/03/17
Pruebas	1 día	31/03/17	01/04/17
Refactorización	1 día	01/04/17	02/04/17
<b>▲ Construcción</b>	<b>50 días</b>	<b>02/04/17</b>	<b>03/06/17</b>
<b>▲ Primera iteración</b>	<b>15 días</b>	<b>02/04/17</b>	<b>21/04/17</b>
Drefinamiento de requisitos	1 día	02/04/17	03/04/17
Mejora del diseño	1 día	03/04/17	04/04/17
Implementación	10 días	05/04/17	17/04/17
Pruebas	1 día	17/04/17	18/04/17
Refactorización	2 días	18/04/17	21/04/17
<b>▲ Segunda iteración</b>	<b>21 días</b>	<b>21/04/17</b>	<b>17/05/17</b>
Refinamiento de los requisitos	2 días	21/04/17	23/04/17
Mejora del diseño	2 días	23/04/17	26/04/17
Implementación	14 días	26/04/17	13/05/17
Pruebas	1 día	13/05/17	14/05/17
Refactorización	2 días	15/05/17	17/05/17
<b>▲ Tercera iteración</b>	<b>14 días</b>	<b>17/05/17</b>	<b>03/06/17</b>
Refinamiento de los requisitos	1 día	17/05/17	18/05/17
Mejora del diseño	2 días	18/05/17	21/05/17
Implementación	8 días	21/05/17	31/05/17
Pruebas	2 días	31/05/17	02/06/17
Refactorización	1 día	02/06/17	03/06/17
<b>▲ Transición</b>	<b>9 días</b>	<b>04/06/17</b>	<b>15/06/17</b>
Refactorización	2 días	04/06/17	06/06/17
Refinamiento de las pruebas	2 días	06/06/17	08/06/17
Actualización de la documentación	3 días	09/06/17	12/06/17
Manual de usuario	2 días	12/06/17	15/06/17

Figura 3: Ejecución final de la planificación

Si se desea, se pueden consultar los diagramas de Gantt de la planificación y la ejecución real del proyecto en el Anexo I: Diagramas de Gantt

#### 4.2.1. Arquitectura

La aplicación que se ha desarrollado en este trabajo es una aplicación web. Por lo tanto, a nivel de diseño físico utiliza una arquitectura cliente-servidor [3], donde los primeros actúan como demandantes realizando peticiones al servidor, mientras que este último acepta un número determinado de conexiones de clientes y se encarga de responder las peticiones que soliciten.

La aplicación web se divide en dos partes. Por un lado, se ha implementado la parte del servidor, presentando los datos en forma de servicio web, y, por otro lado, la del cliente, la cual presentará los datos al usuario final y permitirá a este interactuar con el sistema.

## 4.3. Arquitectura servidor

A nivel arquitectónico la aplicación de lado del servidor sigue el patrón Modelo-Vista-Controlador (MVC) [4]. Este modelo se caracteriza por la división de la aplicación en tres capas, cada una con un objetivo.

### 4.3.1. Características

A continuación explicamos la función de cada capa en esta aplicación:

- Modelo: representa cada una de las entidades del mundo real, en este caso, como *case classes* en Scala. Esta capa es la responsable de la persistencia de la información del sistema en forma de modelo.
- Vista: representa las distintas presentaciones de los datos de la aplicación. En este caso se hace en forma de API REST.
- Controlador: se encarga de la lógica de la aplicación. Los controladores actúan como intermediarios entre la vista y el modelo. Dentro de la aplicación, en esta capa, se realizan también conversiones de las entidades reales del modelo a entidades amigables para la vista y viceversa.

### 4.3.2. Flujo de ejecución

El patrón MVC también define un flujo de ejecución, el cual se puede consultar en la Figura 4. Se compone de los siguientes pasos:

1. El usuario realiza una petición al servidor.
2. El enrutador del *framework* (Play) decide cuál será el controlador encargado de manejarla.
3. En el controlador correspondiente se realiza una validación antes de la ejecución del bloque, que determinará si el usuario tiene autorizado el acceso.
4. El controlador se encarga, en caso de la peticiones POST, de la transformación de los datos a clases Scala [5] que actúan como objetos de vista (*view objects*), realizando, además, una serie de validaciones implícitas. Esto se hace utilizando las utilidades *Form* [6] de Scala.
5. Una vez hechas las validaciones (en caso de ser necesarias), se realizarán las operaciones de lógica de acuerdo a la petición realizada y se actualiza el modelo.
6. El controlador envía los datos a la vista.
7. Por último, se envía la respuesta HTTP al cliente con el JSON correspondiente.

En nuestra aplicación, utilizamos objetos de vista para mostrar u ocultar al usuario final los atributos necesarios en cada instante. Desde el propio controlador se transforman los objetos de la vista en las instancias que representan a las entidades reales. Las propias entidades se ocupan de validar sus datos.

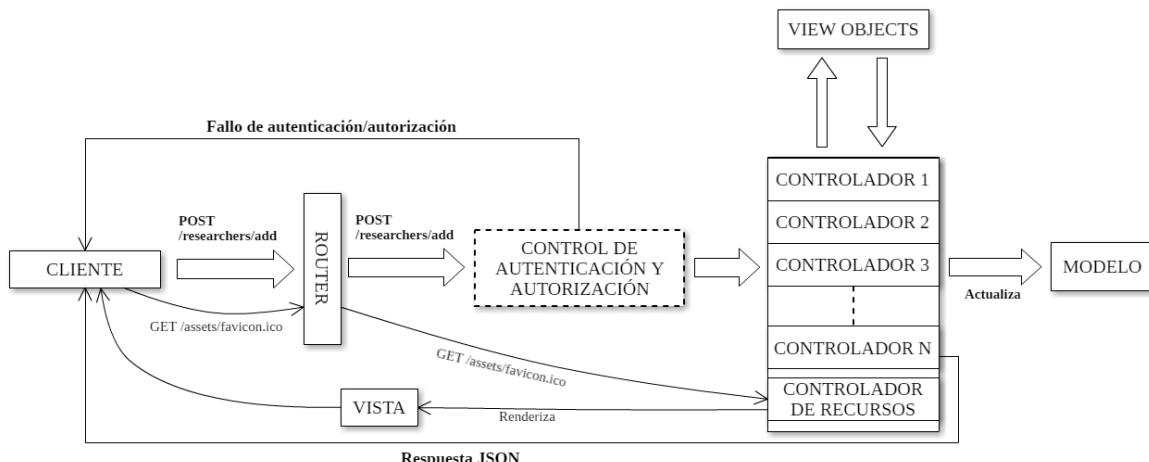


Figura 4: Arquitectura MVC de la aplicación en el servidor

## 4.4. Arquitectura cliente

### 4.4.1. Características

En la parte cliente de la aplicación se utiliza Redux [7], tecnología y estilo arquitectónico para construir aplicaciones con ReactJS [8] y que está basada en el patrón arquitectónico Flux [9] empleado por Facebook. Redux sigue las recomendaciones de Flux aunque tiene ciertas características diferentes en cuanto al diseño arquitectónico.

En la Figura 5 se puede ver el flujo de ejecución de la aplicación web cliente. La responsabilidad de las distintas partes que forman parte de Redux y que se pueden ver dentro del flujo de ejecución mencionado son las siguientes:

- **Componentes (components):** estos elementos son los que forman la vista. Los componentes escuchan las interacciones del cliente y las traducen en acciones.
- **Acciones (actions):** representan un evento que ocurre dentro del contexto de la aplicación. Cuando son ejecutadas devuelven un objeto al cuál deberá asociarse al menos un “tipo”, que servirá para saber donde debe ser recogida la acción. Las acciones pueden además tener asignados otros datos adicionales.
- **Creadores de acciones (action creators):** en algunas ocasiones, cuando no se dispone en el mismo instante de los datos que queremos asociar a una acción, se utilizan los creadores de acciones. Estos elementos actúan como punto intermedio entre el componente y el envío de la propia acción. Se utilizan, por ejemplo, cuando se requiere realizar llamadas asíncronas a un API para la obtención de los datos.
- **Reductores (reducers):** la responsabilidad de los reductores es recoger las acciones enviadas y devolver un objeto nuevo con el siguiente estado de la aplicación. Se ha implementado un reductor por cada una de las entidades de la aplicación.
- **Almacén (store):** el almacén guarda el estado (los datos) de la aplicación, permite el acceso a este desde otras partes de la vista y su actualización mediante la ejecución de acciones en base a un objeto *dispatcher* proporcionado por la librería.

En resumen, las características principales de Redux son las siguientes:

- Flujo de datos unidireccional.
- Flujo de datos asíncrono, donde se envían acciones desde los componentes de la vista y se recogen desde los reductores.
- Dispone de un almacén de datos donde se guarda todo el estado de la aplicación. Esta es la gran diferencia con Flux, patrón arquitectónico en el que está basado Redux, donde existen varios almacenes de datos para cada una de las partes.

- Los cambios en el estado de la aplicación se realizan mediante funciones puras. Esto es, una vez los reductores reciben una acción se devuelve un objeto nuevo con el nuevo estado pero no se modifica el anterior.

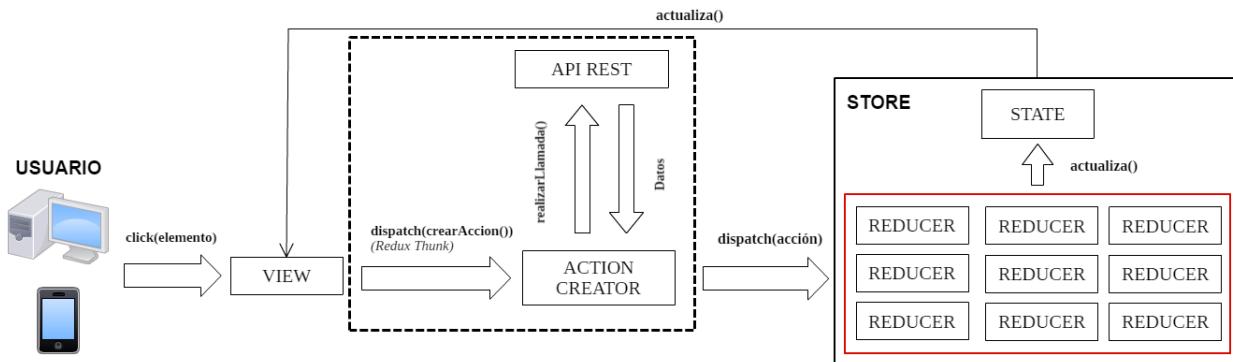


Figura 5: Flujo de ejecución de la aplicación cliente

En esta parte, se utiliza además una librería adicional para gestionar el enruteado en la vista. Esta utilidad, que será explicada de forma más extensa en el siguiente apartado, permite dirigir las direcciones URL introducidas en el navegador a los componentes que correspondan de la vista.

## 5. Tecnologías y productos de terceros

### 5.1. Introducción

En la aplicación se han utilizado multitud de tecnologías, tanto en servidor como en cliente, que facilitan el desarrollo y mejoran la reutilización de código. También se han utilizado herramientas para la automatización de tareas y de control sobre la calidad del código.

Las tecnologías utilizadas en el servidor se han basado en el uso de *frameworks* que proporcionan funcionalidades más completas y que suponen un control de este más complejo para el desarrollador. En cliente, en cambio, se han utilizado múltiples librerías JavaScript más pequeñas, que se han ido ampliando con otras adicionales disponibles en proyectos libres de terceros para solucionar problemas comunes como la: gestión de formularios, control de algunos componentes web, enrutamiento, etc.

Todas estas tecnologías se encuentran explicadas y referenciadas en los siguientes apartados.

### 5.2. Tecnologías en servidor

Las tecnologías utilizadas en el servidor son los siguientes:

- Scala [5]: lenguaje de programación tanto imperativo como funcional, orientado a objetos, de tipado fuerte y que se ejecuta en la máquina virtual de Java (JVM). Es compatible con la mayoría de librerías existentes en Java.
- Play Framework [10]: *framework* para la realización de aplicaciones web en Java y Scala que sigue la arquitectura MVC. Play sigue el paradigma software de convención sobre configuración, permite el muestreo de errores en el navegador, carga de código en caliente y otras funcionalidades diversas (gestión de plantillas, gestión de dependencias, etc.).

- Slick [11]: *framework* de acceso a base de datos para Scala cuyo modelo de acceso (FRM, Functional Relational Mapping) está inspirado en el proyecto LINQ [12] de Microsoft. Slick permite el acceso a los elementos de la base de datos como si se tratase de colecciones normales de Scala.  
Desde Slick se pueden realizar operaciones de base de datos asíncronas. Utilizando este framework se pueden realizar consultas de manera que el objeto devuelto sea de tipo *Future*, esto es, un objeto que representa una operación que se va a realizar y que dará lugar al objeto final. Esto se utiliza en la aplicación para evitar bloqueos.
- Scala Bcrypt [13]: envoltorio de la librería jBCrypt para realizar *hash* sobre contraseñas.
- AuthentikatJWT [14]: librería que implementa el estándar JWT (JSON Web Token) [15], el cual define una manera de transmitir información de forma segura entre dos partes mediante el envío de una firma cifrada (*token*).
- ScalaTest [16]: herramienta para la realización de pruebas para lenguaje Scala o Java. Ofrece buena integración con Eclipse y SBT, así como otras muchas tecnologías que no se utilizan para este proyecto.
- SBT (Scala Build Tool) [17]: herramienta de código abierto para la gestión de dependencias en proyectos realizados en lenguaje Java o Scala. Soporta repositorios Maven y permite automatizar tareas de compilación, pruebas y despliegue. Soporta diferentes *plugins* y utiliza un lenguaje específico del dominio (DSL, Domain Specific Language) [18] para construir las descripciones y definir las tareas.

### 5.3. Tecnologías en cliente

Las tecnologías utilizadas en el cliente son los siguientes:

- HTML (HyperText Markup Language) [19]: lenguaje de etiquetado para desarrollo de páginas web que sirve para definir la estructura de las vistas.
- CSS (Cascade Stylesheet) [20]: lenguaje que permite definir la presentación de un documento HTML, ofreciendo control sobre el estilo y formato de los documentos.
- Javascript [21]: lenguaje de programación dinámico y basado en prototipos comúnmente usado en navegadores para la ejecución de *scripts* en el ordenador cliente. Se utiliza siguiendo la última versiones del estándar ECMAScript, lo que hace que su uso sea mucho más sencillo y el código más legible.
- ReactJS [8]: librería JavaScript declarativa y basada en componentes para crear interfaces de usuario. En una aplicación, ReactJS se utiliza mediante la creación de clases que heredan de `React.Component`, las cuales permiten desgranar la aplicación en pequeños trozos que pueden ser utilizados desde distintas partes.
- React Router [22]: colección de componentes de navegación para React que se establecen de forma declarativa en la aplicación. Mediante la declaración de rutas con esta librería se navega en la aplicación cliente como una SPA [23] (Single Page Application) mejorando así la experiencia del usuario. Esta librería permite recoger llamadas realizadas por el usuario desde el navegador y mapear la URL a un componente de React sin llegar a enviar la acción al servidor. Esto solo ocurre en las rutas que se deseé definir para la aplicación.
- React Select [24]: etiqueta de HTML “Select” redefinida como un componente en React. Añade una serie de funciones como *callbacks*, multiselección, autocompletado y soporte para llamadas asíncronas.
- React Big Calendar [25]: calendario de eventos construido para React en navegadores modernos. Está inspirado en la librería JavaScript “Full Calendar” y permite diversas vistas de eventos: por mes, semana, día y agenda. Debe utilizarse en conjunción con otras librerías que permitan la manipulación de fechas.
- React Datepicker [26]: componente de entrada de texto diseñado para React que permite desplegar un seleccionable de fechas. Al igual que React Big Calendar, debe utilizarse en conjunción con alguna librería de manipulación de este tipo de datos.
- Redux [7]: Redux es un contenedor de estado para aplicaciones JavaScript que se basa en el patrón arquitectónico Flux utilizado por Facebook, pero introduciendo pequeños cambios que consiguen hacerlo más sencillo. Redux se basa en tres principios básicos: *i*) el estado de la aplicación es de solo lectura, *ii*) los cambios se hacen siempre mediante funciones puras y *iii*) solo hay una fuente de verdad, es decir, un único almacén de datos donde se guardan todos los datos de la aplicación (el estado).

- Redux Thunk [27]: tecnología que se utiliza para aplazar el lanzamiento de una acción en Redux. Permite utilizar el despachador de acciones disponible en Redux para devolver una función en vez de un dato. Así, es posible realizar peticiones asíncronas a, por ejemplo, un servicio web y devolver una acción con los datos obtenidos una vez esta se resuelve.
- Redux Form [28]: tecnología para manejo de formularios utilizando Redux. Dispone de un reductor que escucha las acciones despachadas por los formularios así como un componente para el diseño de formularios en la vista. Los cambios que se realizan en estos componentes son enviados y almacenados en el estado.
- MomentJS [29]: librería para procesar, manipular y mostrar fechas en JavaScript.
- SASS (Syntactically Awesome Style Sheets) [30]: librería extensión de CSS que sirve para añadir funcionalidades al lenguaje. Permite el uso de variables, importación de otros ficheros, reglas concatenadas y herencia, siendo compatible totalmente con CSS. SASS ayuda a que los ficheros CSS se mantengan bien organizados.
- Jquery [31]: librería que simplifica las funciones JavaScript para definir el comportamiento dinámico de las páginas.
- Twitter Bootstrap [32]: es una librería con una serie de elementos predefinidos que simplifica el desarrollo web en cliente volviéndolo más sencillo y vistoso mediante el uso de HTML, CSS y JavaScript.
- Font Awesome [33]: permite la inserción y edición simple de iconos para la web mediante CSS.
- NPM [34]: gestor de paquetes mediante el cual podemos manejar dependencias (en el caso de esta aplicación, en la parte cliente) de forma sencilla.
- Webpack [35]: empaquetador de módulos eficiente. Permite generar un archivo único en base a todos los que se disponen en la aplicación. Permite, además, generar solo fragmentos que necesite la página, importar y empaquetar recursos y añadir *plugins*. En esta aplicación se utilizan *plugins* para “minificar” código, recargar el navegador de forma automática con los cambios, compilar código ECMAScript versión 6 a 5 e importar y compilar SASS.
- Babel [36]: compilador que se encarga de transformar el código de las versiones de las últimas especificaciones JavaScript a la versión 5, soportada ampliamente hoy en día por todos los navegadores.
- BrowserSync [37]: herramienta que permite actualizar la página del navegador con cada cambio en el código, así como sincronizar el *scroll* y las pulsaciones en los navegadores conectados. Esto facilita las labores de desarrollo y pruebas.
- ESLint [38]: herramienta de código abierto para JavaScript que realiza un análisis de código para buscar problemas y enviar avisos sin tener que llegar a realizar la compilación. El uso de esta herramienta permite mejorar la calidad del código, evitando errores de programación habituales tales como: variables no utilizadas, código mal tabulado, utilización de variables fuera de rango, llaves o paréntesis sin cerrar, etc.

## 5.4. Tecnologías para el desarrollo

Las herramientas utilizadas para llevar a cabo el desarrollo de la aplicación son las siguientes:

- Windows 10 y Ubuntu 17.04: sistemas operativos en los cuales se ha llevado a cabo la aplicación. Se ha desarrollado tanto desde un ordenador de sobremesa con Windows 10 como en un portátil con Ubuntu.
- Google Chrome: navegador web utilizado tanto para la búsqueda de información como para el desarrollo de la aplicación.
- Mozilla Firefox: navegador web para realizar las pruebas de la aplicación.
- LibreOffice Writer: procesador de textos utilizado para la realización de la documentación.
- Visual Paradigm 13.2: aplicación de escritorio para realizar gran parte de los diagramas presentados en esta documentación.
- Cacoo: aplicación web utilizada para realizar algunos de los diagramas que se muestran en la documentación.
- Git: sistema de control de versiones.
- GitHub [39]: plataforma de desarrollo colaborativo de software que permite alojar proyectos que utilizan Git como sistema de control de versiones.

- Eclipse: entorno de desarrollo multiplataforma en el que se ha realizado, tanto la implementación de la automatización de tareas en cliente y servidor como el desarrollo del código en servidor utilizando lenguaje Scala.
- Scala IDE: plugin para Eclipse que simplifica al usuario la programación en Scala.
- Egit: plugin para integrar git en eclipse.
- Atom: Editor de textos multiplataforma y con gran facilidad de personalización, mediante *plugins*, para desarrollar código HTML, CSS y JavaScript. Se ha utilizado para implementar todo el código del cliente.
- Atom-beautify: *plugin* para Atom que permite embellecer el código estableciendo al configuración de determinados estándares como: tipo de indentación, cantidad de tabulaciones, líneas vacías, etc.
- File-icons: *plugin* para Atom que añade iconos a los ficheros del árbol de directorios del proyecto, diferenciándolos por su extensión.
- Linter-eslint: *plugin* para Atom que detecta, según el usuario escribe, los diversos problemas que pueda tener el código según la configuración establecida en la herramienta ESLint.
- React: *plugin* para Atom con soporte para indentación, autocompletado, formateo y sugerencias relacionadas con la librería ReactJS.

## 6. Análisis de requisitos

En esta sección se explicarán los roles de los usuarios y los requisitos, tanto funcionales como no funcionales, que tiene la aplicación web completa.

### 6.1. Identificación de actores

En este sistema se han identificado dos tipos de actores:

- El **administrador** puede ver y editar todos los investigadores, autores, congresos, proyectos y revistas así como consultar el calendario de actividades.
- El **investigador** puede consultar lo mismo que el usuario administrador, pero solo tiene posibilidad de editar sus propios datos como investigador y autor así como los congresos, proyectos y revistas a los que está asociado.

### 6.2. Diagrama de casos de uso

En la Figura 6 se presenta el diagrama de casos de uso global del sistema. Se divide en los casos de uso de identificación y salida del usuario del sistema, la consulta del estado global del grupo y en cuatro paquetes que representan la gestión de las distintas entidades existentes.

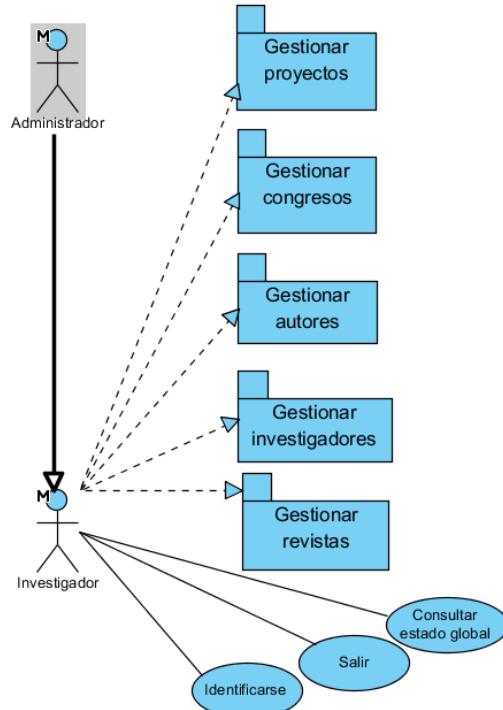


Figura 6: Diagrama de casos de uso global

### 6.2.1. Diagrama de casos de uso de gestión de proyectos

En el diagrama presentado en la Figura 7 se puede observar que, dentro de la gestión de proyectos, el administrador puede realizar todas las operaciones típicas de la gestión, mientras que el investigador no podrá crear proyectos y solo tendrá la posibilidad de actualizar y eliminar aquellos de los cuales sea autor.

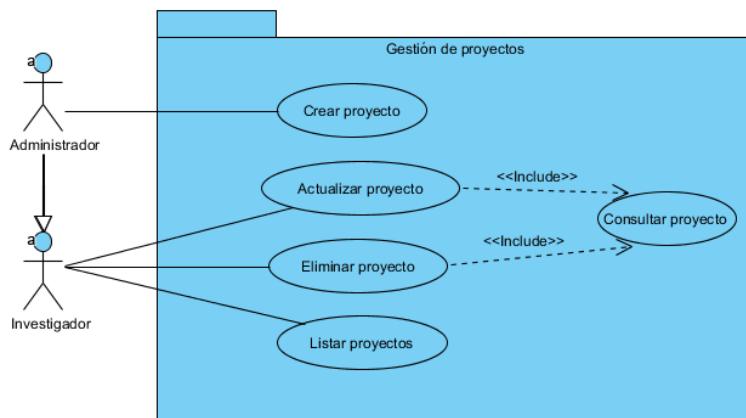


Figura 7: Diagrama de casos de uso de gestión de proyectos

### 6.2.2. Diagrama de casos de uso de gestión de congresos

En el diagrama presentado en la Figura 8 se puede observar que, dentro de la gestión de congresos, el administrador puede realizar todas las operaciones típicas de la gestión, mientras que el investigador solo puede listar congresos y eliminar y actualizar aquellos de los cuales sea autor.

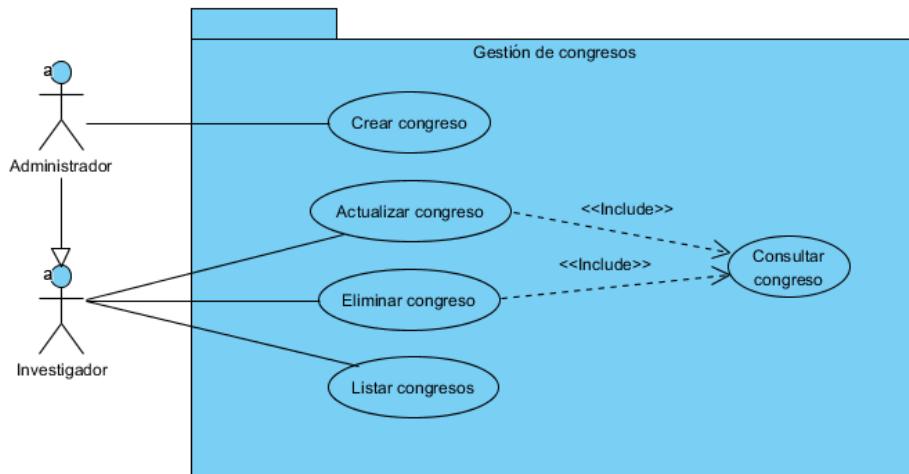


Figura 8: Diagrama de casos de uso de gestión de congresos

### 6.2.3. Diagrama de casos de uso de gestión de revistas

En el diagrama presentado en la Figura 9 se puede observar que, dentro de la gestión de revistas, el administrador puede realizar todas las operaciones típicas de la gestión, mientras que el investigador solo podrá listar revistas, actualizarlas y eliminarlas si está asignado a ellas como autor.

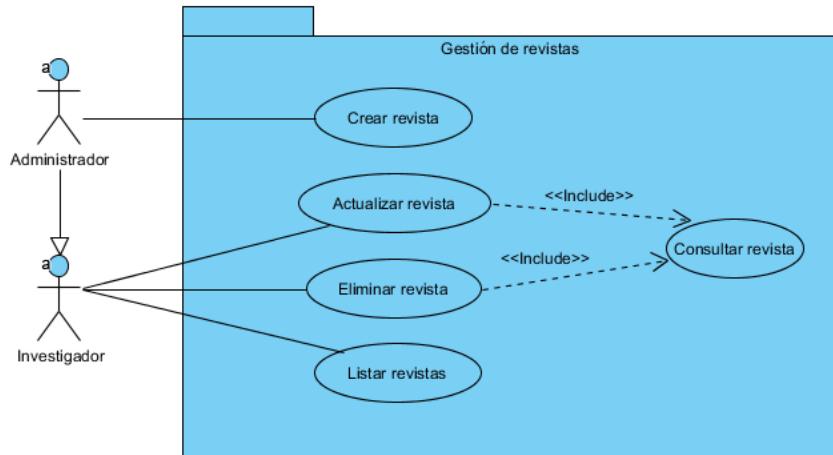


Figura 9: Diagrama de casos de uso de gestión de revistas

### 6.2.4. Diagrama de casos de uso de gestión de investigadores

En el diagrama presentado en la Figura 10 se puede observar que, dentro de la gestión de investigadores, el administrador puede realizar todas las operaciones típicas de la gestión, mientras que el investigador solamente podrá ver el listado de investigadores existentes, así como editar su perfil.

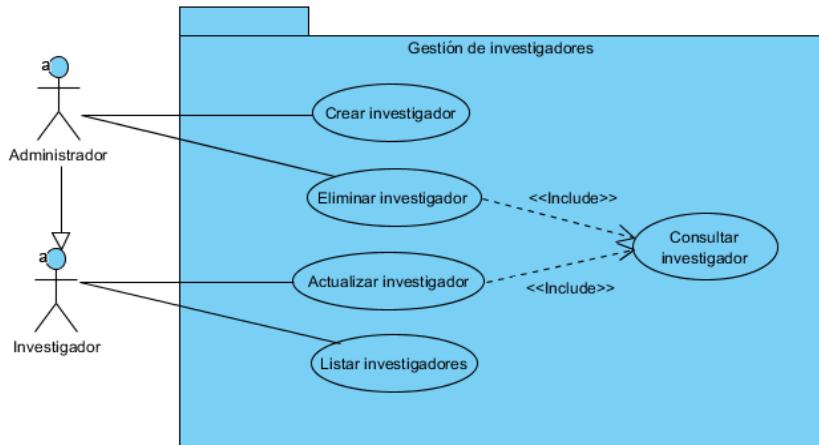


Figura 10: Diagrama de casos de uso de gestión de investigadores

### 6.2.5. Diagrama de casos de uso de gestión de autores

En el diagrama presentado en la Figura 11 se puede observar que, dentro de la gestión de autores, el administrador puede realizar todas las operaciones típicas de la gestión, mientras que el investigador solo podrá actualizar y eliminar su perfil de autor, así como listar todos los autores.

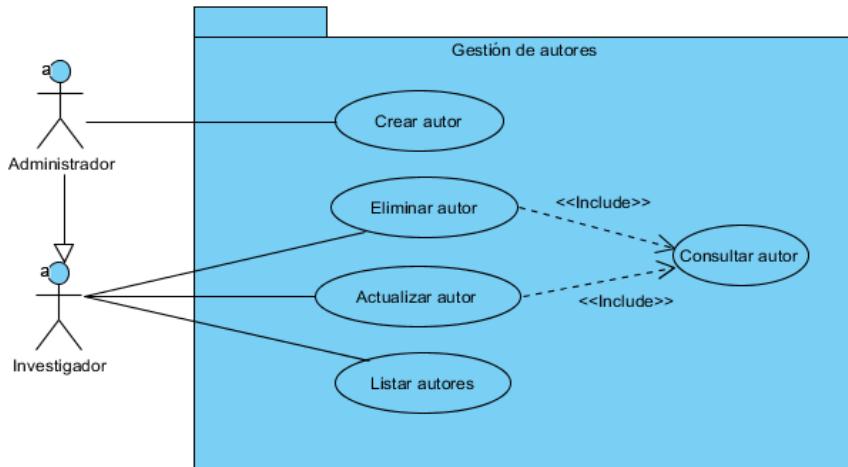


Figura 11: Diagrama de casos de uso de gestión de autores

### 6.2.6. Requisitos no funcionales

El sistema debe ser robusto y disponer de buena fluidez en la navegación por las diferentes pantallas de la interfaz. Además, los tiempos de carga también deben ser rápidos. Todo esto se intenta conseguir mediante la creación de la aplicación cliente como una aplicación de una sola página (SPA, Single Page Application). Este tipo de aplicación reduce los tiempos de carga y aumenta la fluidez, debido a que, mientras el usuario navega, solo es necesaria la carga de pequeñas partes de la aplicación.

El sistema debe ser intuitivo y sencillo de utilizar para el usuario final tanto en su uso desde una pantalla de gran tamaño (p.ej. la de un ordenador personal), como desde una pantalla pequeña (p.ej. la de un teléfono móvil). La aplicación cliente debe adaptarse al tamaño del dispositivo haciendo la interfaz más agradable para el usuario final.

Al existir diversas posibilidades de gestión de entidades, los errores de los formularios deben presentarse de forma clara y precisa para el usuario, de manera que puede corregir e intuir fácilmente los posibles problemas en la inserción o edición de datos en cualquier parte del sistema.

### 6.3. Descripción de casos de uso

En este sistema, como se puede apreciar en el diagrama de casos de uso representado mediante herencia entre los dos actores, un administrador puede realizar cualquier tarea que un investigador también pueda hacer. Por lo tanto, siempre que en la descripción de casos de uso se presente a un investigador como actor, debe tenerse en cuenta que un administrador también es un actor de ese caso de uso.

A continuación, se presentan las descripciones de los casos de uso que se consideran como los más importantes. El resto de casos de uso, muy similares a estos, se pueden consultar en el Anexo II: Descripciones de casos de uso detallados.

#### 6.3.1. Caso de uso identificarse

<b>NOMBRE</b>	Gestión de usuarios – Identificarse	
<b>ACTORES</b>	Investigador	
<b>OBJETIVO</b>	El usuario no identificado puede acceder al sistema mediante el uso de su correo y contraseña.	
<b>PRECONDICIONES</b>	El usuario no puede estar identificado en la aplicación.	
<b>FLUJO PRINCIPAL</b>		
ACTOR	SISTEMA	
1. El usuario introduce los datos.	2. Comprueba si existe el usuario introducido y si la contraseña es válida. 3. Comprueba si el usuario tiene permisos de acceso al sistema. En caso de no tenerlo <<Usuario sin acceso>>. 4. Se identifica al usuario y se redirige a la página principal.	
<b>FLUJO ALTERNATIVO: USUARIO SIN ACCESO</b>		
		4. Se muestra el mensaje de error explicando el motivo.

Tabla 2: Descripción caso de uso "identificarse"

#### 6.3.2. Caso de uso salir

<b>NOMBRE</b>	Gestión de usuarios – Salir	
<b>ACTORES</b>	Investigador	
<b>OBJETIVO</b>	El usuario identificado puede salir del sistema.	
<b>PRECONDICIONES</b>	El usuario debe estar previamente identificado en el sistema.	
<b>FLUJO PRINCIPAL</b>		
ACTOR	SISTEMA	
1. El usuario pulsa el botón de Salir.	2. El sistema redirige a la página de inicio de sesión.	

Tabla 3: Descripción de caso de uso "salir"

#### 6.3.3. Caso de uso consultar estado global

<b>NOMBRE</b>	Gestión de calendarios – Consultar estado global
---------------	--

<b>ACTORES</b>	Investigador
<b>OBJETIVO</b>	El usuario puede consultar los eventos del grupo como un calendario o una agenda.
<b>PRECONDICIONES</b>	El usuario debe estar identificado en el sistema.
<b>FLUJO PRINCIPAL</b>	
ACTOR	SISTEMA
1. Usuario solicita acceso a la sección de calendarios.	
	2. El sistema muestra la página de calendarios desde el mes actual.

Tabla 4: Descripción de caso de uso "consultar estado global"

#### 6.3.4. Caso de uso de gestión de investigadores: consultar investigadores

<b>NOMBRE</b>	Gestión de investigadores – Consultar listado investigadores
<b>ACTORES</b>	Investigador
<b>OBJETIVO</b>	El usuario puede consultar la lista de investigadores registrados en el sistema.
<b>PRECONDICIONES</b>	El usuario debe estar identificado en el sistema.
<b>FLUJO PRINCIPAL</b>	
ACTOR	SISTEMA
1. Usuario solicita acceso a la sección “Investigadores”.	
	2. El sistema busca la lista de investigadores en la base de datos.
	3. El sistema devuelve una tabla con la lista de todos los investigadores.

Tabla 5: Descripción de caso de uso "consultar investigadores"

#### 6.3.5. Caso de uso de gestión de investigadores: consultar investigador

<b>NOMBRE</b>	Gestión de investigadores – Consultar investigador
<b>ACTORES</b>	Usuario investigador
<b>OBJETIVO</b>	El usuario puede consultar los datos de un investigador del sistema.
<b>PRECONDICIONES</b>	El usuario debe estar identificado en el sistema. Deben existir investigadores registrados.
<b>FLUJO PRINCIPAL</b>	
ACTOR	SISTEMA
1. El usuario pulsa en el botón “Detalle” de un investigador de la tabla.	
	2. El sistema busca los datos del investigador buscado en la base de datos.
	3. El sistema muestra una pantalla con los datos del investigador buscado.

Tabla 6: Descripción de caso de uso "consultar investigador"

#### 6.3.6. Caso de uso de gestión de investigadores: añadir investigador

<b>NOMBRE</b>	Gestión de investigadores – Añadir investigador
---------------	---

<b>ACTORES</b>	Administrador
<b>OBJETIVO</b>	El usuario añade un nuevo investigador en el sistema.
<b>PRECONDICIONES</b>	El usuario debe estar identificado en el sistema.
<b>FLUJO PRINCIPAL</b>	
ACTOR	SISTEMA
1. Usuario pulsa en el botón “añadir investigador”.	
	2. El sistema devuelve un formulario con los datos a introducir.
3. El usuario introduce los datos del investigador.	
	4. Se validan los datos del investigador introducido. En caso de datos incorrectos <<datos incorrectos>>
	5. Se guarda el investigador en la base de datos.
	6. El usuario es redirigido a la lista de investigadores mostrando un mensaje de éxito.
<b>FLUJO ALTERNATIVO: DATOS INCORRECTOS</b>	
	5. Se muestran mensajes de error debajo de los campos mal introducidos explicando el problema.

Tabla 7: Descripción de caso de uso "añadir investigador"

### 6.3.7. Caso de uso de gestión de investigadores: actualizar investigador

<b>NOMBRE</b>	Gestión de investigadores – Actualizar investigador
<b>ACTORES</b>	Investigador
<b>OBJETIVO</b>	El usuario puede actualizar los datos de un investigador en el sistema.
<b>PRECONDICIONES</b>	Se parte del caso de uso “consultar investigador”. El usuario debe estar identificado en el sistema. El usuario que se edita debe tener permisos de administrador o bien el investigador que se edite debe estar asociado al usuario identificado.
<b>FLUJO PRINCIPAL</b>	
ACTOR	SISTEMA
1. El usuario realiza las modificaciones que considere oportunas sobre el investigador y pulsa sobre “Actualizar investigador”.	
	2. Se validan los datos del investigador. En caso de datos incorrectos <<datos incorrectos>>
	3. Se actualiza el investigador en la base de datos.
	4. El usuario es redirigido a la lista de investigadores mostrando un mensaje de éxito.
<b>FLUJO ALTERNATIVO: DATOS INCORRECTOS</b>	
	3. Se muestran mensajes de error debajo de los campos mal introducidos explicando el problema.

Tabla 8: Descripción de caso de uso "actualizar investigador"

### 6.3.8. Caso de uso de gestión de investigadores: eliminar investigador

<b>NOMBRE</b>	Gestión de investigadores – Eliminar investigador
---------------	---

<b>ACTORES</b>	Administrador
<b>OBJETIVO</b>	Debe haber investigadores registrados en el sistema.
<b>PRECONDICIONES</b>	Se parte del caso de uso “consultar investigador”. El usuario debe estar identificado en el sistema.
<b>FLUJO PRINCIPAL</b>	
ACTOR	SISTEMA
1. El usuario pulsa en el botón “Eliminar investigador”.	
	2. El investigador se elimina de la base de datos.
	3. El sistema redirige al usuario al listado de investigadores mostrando un mensaje de éxito.

Tabla 9: Descripción de caso de uso "eliminar investigador"

### 6.3.9. Caso de uso de gestión de congresos: consultar lista de congresos

<b>NOMBRE</b>	Gestión de congresos – Consultar listado congresos
<b>ACTORES</b>	Investigador
<b>OBJETIVO</b>	El usuario puede consultar la lista de congresos.
<b>PRECONDICIONES</b>	El usuario debe estar identificado en el sistema.
<b>FLUJO PRINCIPAL</b>	
ACTOR	SISTEMA
1. Usuario solicita acceso a la sección “congresos”.	
	2. El sistema busca la lista de congresos en la base de datos.
	3. El sistema devuelve una tabla con la lista de todos los congresos.

Tabla 10: Descripción de caso de uso "consultar lista de congresos"

### 6.3.10. Caso de uso de gestión de congresos: consultar congreso

<b>NOMBRE</b>	Gestión de congresos – Consultar congreso
<b>ACTORES</b>	Usuario investigador
<b>OBJETIVO</b>	El usuario puede consultar los datos de un congreso del sistema.
<b>PRECONDICIONES</b>	El usuario debe estar identificado en el sistema. Deben existir congresos almacenados en el sistema.
<b>FLUJO PRINCIPAL</b>	
ACTOR	SISTEMA
1. El usuario pulsa en el botón “Detalle” de un congreso de la pantalla de lista de congresos.	
	2. El sistema busca los datos del congreso en la base de datos.
	3. El sistema busca los estados de publicación y los tipos de congreso asociados a este.

	4. El sistema muestra una pantalla con los datos del congreso.
--	--

Tabla 11: Descripción de caso de uso "consultar congreso"

### 6.3.11. Caso de uso de gestión de congresos: añadir congreso

<b>NOMBRE</b>	Gestión de congresos – Añadir congreso
<b>TIPO</b>	Principal
<b>ACTORES</b>	Administrador
<b>OBJETIVO</b>	El usuario añade un nuevo congreso en el sistema.
<b>PRECONDICIONES</b>	El usuario debe estar identificado en el sistema.
<b>FLUJO PRINCIPAL</b>	
ACTOR	SISTEMA
1. Usuario pulsa en el botón “añadir congreso”.	
	2. El sistema obtiene de la base de datos la lista de tipos de congreso, autores totales y estados de publicaciones para que el usuario pueda seleccionarlas en el formulario.
	3. El sistema devuelve un formulario con los datos a introducir.
4. El usuario introduce los datos del congreso así como los autores participantes.	
	5. Se validan los datos del congreso introducido. En caso de datos incorrectos <<datos incorrectos>>.
	6. Se guarda el congreso en la base de datos.
	7. El usuario es redirigido a la lista de congresos mostrando un mensaje de éxito.
<b>FLUJO ALTERNATIVO: DATOS INCORRECTOS</b>	
	6. Se muestran mensajes de error debajo de los campos mal introducidos explicando el problema.

Tabla 12: Descripción de caso de uso "añadir congreso"

### 6.3.12. Caso de uso de gestión de congresos: actualizar congreso

<b>NOMBRE</b>	Gestión de congresos – Actualizar congreso	
<b>ACTORES</b>	Investigador	
<b>OBJETIVO</b>	El usuario actualiza el congreso en el sistema.	
<b>PRECONDICIONES</b>	<ul style="list-style-type: none"> <li>• Se parte del caso de uso “consultar congreso”.</li> <li>• El usuario debe estar identificado y el congreso debe existir en el sistema.</li> <li>• El usuario debe ser administrador o bien investigador, siendo en este último caso participante del congreso que se quiere actualizar.</li> </ul>	
<b>FLUJO PRINCIPAL</b>		
ACTOR	SISTEMA	
1. El usuario edita los datos del congreso.	2. Se validan los datos de congreso introducidos. En caso de datos incorrectos <<datos incorrectos>>. 3. Se guarda el congreso en la base de datos. 4. El usuario es redirigido a la lista de congresos mostrando un mensaje de éxito.	
<b>FLUJO ALTERNATIVO: DATOS INCORRECTOS</b>		
	7. Se muestran mensajes de error debajo de los campos mal introducidos explicando el problema.	

Tabla 13: Descripción de caso de uso "actualizar congreso"

### 6.3.13. Caso de uso de gestión de congresos: eliminar congreso

<b>NOMBRE</b>	Gestión de congresos – Eliminar congreso	
<b>ACTORES</b>	Investigador	
<b>OBJETIVO</b>	El congreso seleccionado se elimina del sistema.	
<b>PRECONDICIÓN</b>	<ul style="list-style-type: none"> <li>• Debe haber congresos almacenados en el sistema.</li> <li>• El usuario debe estar identificado en el sistema. El usuario puede ser:               <ul style="list-style-type: none"> <li>◦ <u>Administrador</u>: puede eliminar cualquier congreso.</li> <li>◦ <u>Investigador</u>: puede eliminar los congresos en los que participe.</li> </ul> </li> </ul>	
<b>FLUJO PRINCIPAL</b>		
ACTOR	SISTEMA	
1. En la lista de congresos, el usuario pulsa en el botón “Ver” sobre una de las filas.	2. El sistema obtiene los posibles estados de publicaciones, tipos de congreso, la lista de autores completa y los datos del congreso seleccionado con sus autores participantes. 3. El sistema devuelve un formulario con los datos del congreso llenados y la lista de autores participantes en el mismo.	
4. El usuario pulsa en el botón “Eliminar congreso”.	5. El congreso se elimina de la base de datos. 6. El sistema redirige al usuario al listado de congresos mostrando un mensaje de éxito.	

Tabla 14: Descripción de caso de uso "eliminar congreso"

## 7. Diseño del software

### 7.1. Diseño estático

#### 7.1.1. Diseño estático en el servidor

En la Figura 12 se presenta un diagrama de paquetes en el que se pueden ver los distintos paquetes que componen el sistema. Estos paquetes son los siguientes:

- Entidades (entities): representan las entidades reales que se utilizan en la aplicación.
- Objetos de la vista (vos): representan las clases que se mostrarán en la vista. Estas clases están adaptadas para utilizar en la interfaz de usuario. Los VOs contienen métodos que permiten transformar de VO a entidad y viceversa.
- Controladores (controllers): recogen las llamadas al servidor y las interpretan. Las entidades de los controladores suelen tener una relación de asociación con las de los repositorios y utilizan los VOs para transformar entre objetos que se enviarán a la vista y entidades reales.
- Repositorios (repositories): se encargan de realizar las consultas, inserciones, actualizaciones y borrados de base de datos.
- Utilidad (utils): paquete que recoge todas las librerías que se pueden usar por todas las demás clases. En este caso solo se han implementado librerías de utilidad para la encriptación y autenticación, por lo que solo se usan desde el paquete de controladores y desde un VO.

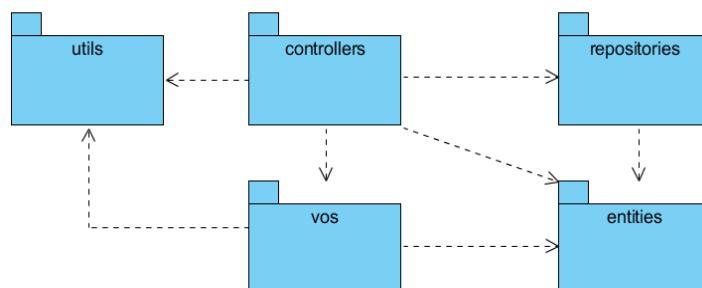


Figura 12: Diagrama de paquetes del backend

#### 7.1.1.1. Convenciones

El diseño estático del sistema se presentará mediante varias secciones, las cuales tendrán cada uno un diagrama de clases asociado. En cada sección, se presentarán algunas clases que ya se explican en secciones anteriores, por lo tanto no se explicarán de nuevo.

Los objetos de tipo *Table* se muestran en todos los diagramas. Estos objetos siempre tienen asociada a ellos la entidad que describe su nombre. Por ejemplo, la clase *ResearcherTable* siempre estará asociada a la clase *Researcher*. Muchas de estas relaciones entre las tablas y su entidad no se mostrarán en los diagramas para facilitar la visibilidad y no tener demasiadas clases que no tengan que ver con lo que se explica en cada uno de esos apartados.

La clase *SecuredAuthenticator* contiene referencias a casi todos los repositorios del sistema. Esta clase tiene una relación de asociación con el repositorio de usuarios y con todos los repositorios que tienen relación con los autores debido a que es la encargada de gestionar la autorización. Como esta clase aparece en casi todos los diagramas, no se mostrará esta relación de asociación con los repositorios.

### 7.1.1.2. Diagrama de clases de identificación y salida del sistema

En la Figura 13 se presenta el diagrama de clases para la entrada y salida del sistema por un usuario. A continuación se explican las responsabilidades de las diferentes clases:

- Password: se encarga de encriptar y desencriptar las contraseñas. Se utiliza desde el controlador de sesiones.
- SessionController: su función es recoger las llamadas del cliente y emitir una respuesta.
- UserRepository: clase encargada de interaccionar con la base de datos de usuarios.
- UserTable: representa a un usuario en la base de datos.
- User: representa a un usuario en el sistema.

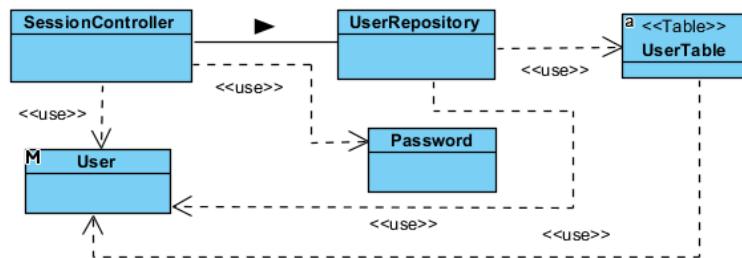


Figura 13: Diagrama de clases de identificación y salida del sistema

### 7.1.1.3. Diagrama de clases de gestión de investigadores

En la Figura 10 se presentan las clases implicadas en la gestión de investigadores. Desde la clase *ResearcherController* se recibe la petición y se interactúa con el repositorio para gestionar los datos almacenados en la base de datos de investigadores. A continuación se explica la responsabilidad de cada una de las clases presentadas en este diagrama:

- ResearcherController: clase encargada de recibir las peticiones y generar una respuesta.
- ResearcherRepository: clase encargada de acceder a los datos almacenados relacionados con los investigadores.
- SecuredAuthenticator: clase encargada de la autenticación y autorización.
- JWTUtils: clase utilidad con la responsabilidad de encriptar y desencriptar el *token* de acceso.
- Researcher: representa a un investigador en el sistema.
- ResearcherVO: representa a un investigador en la vista y contiene también datos de su usuario asociado. Esta clase tiene como responsabilidad, además, realizar las transformaciones entre investigadores normales y investigadores de la vista.
- ResearcherTable: representa a la tabla de investigadores en la base de datos.

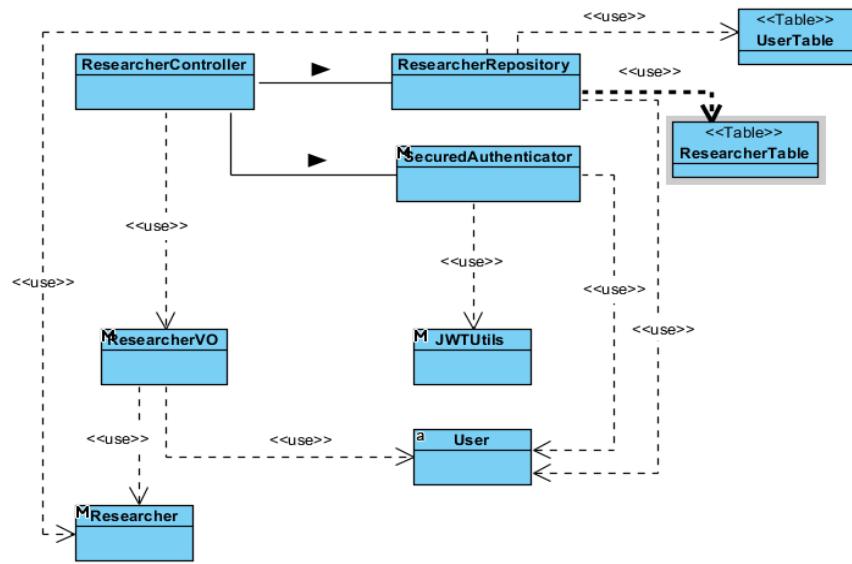


Figura 14: Diagrama de clases de gestión de investigadores

#### 7.1.1.4. Diagrama de clases de gestión de congresos

En la Figura 15 se presenta el diagrama de clases de la gestión de congresos. A continuación se explican las distintas responsabilidades de cada clase:

- **CongressController:** clase encargada de recibir las peticiones relacionadas con los congresos y generar una respuesta.
- **CongressRepository:** clase encargada de acceder a los datos almacenados relacionados con los congresos.
- **AuthorCongressRepository:** clase encargada de realizar operaciones de base de datos que se relacionen con ambas entidades, congresos y autores.
- **Congress:** representa a un proyecto en el sistema.
- **CongressVO:** representa a un congreso en la vista. Se encarga de realizar las transformaciones entre los congresos de la vista y las entidades congreso del sistema.
- **AuthorOfCongressVO:** Representa un autor que además tiene atributos relacionados con el congreso.

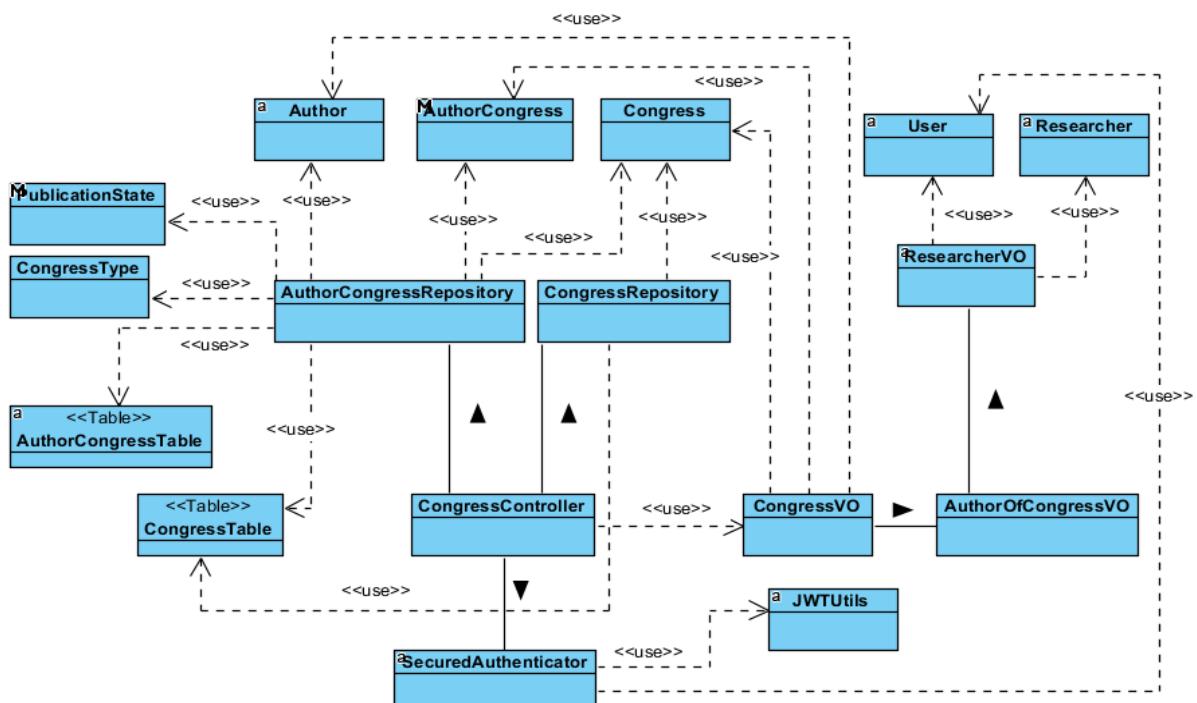


Figura 15: Diagrama de clases de gestión de congresos

El diseño estático relacionado con la gestión de revistas y proyectos es muy similar al de gestión de congresos y el diseño que involucra a la gestión de autores es también similar al de investigadores. Estos diagramas, por lo tanto, no se muestran aquí, pero pueden consultarse en el Anexo III: Diseño estático en el servidor.

### 7.1.2. Diseño estático en el cliente

La aplicación cliente, tal y como se ha explicado en la subsección 4.4 Arquitectura cliente, se divide en tres partes fundamentales: los componentes de la vista, las acciones y los reductores.

La parte de la aplicación relacionada con la vista utiliza componentes, los cuales son clases que extienden de *React.Component*. Esto permite que se pueda hacer una representación en diagramas de clases y paquetes del funcionamiento general de la aplicación.

#### 7.1.2.1. Diagrama de paquetes de la aplicación cliente

En el código de la aplicación cliente no se trabaja directamente con el concepto de paquetes, sino que los distintos ficheros se agrupan en un árbol de directorios. Aún así, en la aplicación existen tres directorios fundamentales que se pueden considerar como tal y que agrupan casi todos los ficheros relacionados con la aplicación y toda la lógica de esta. En la Figura 16 se pueden consultar las partes principales en las que se agrupa la aplicación.

El proceso es el mismo que se ha comentado en la subsección 4.4 Arquitectura cliente, con los componentes enviando acciones que son recibidas por los reductores. Estos actualizan el estado de la aplicación y esta actualización provoca una renderización en el cliente.

En el cliente, mediante la librería *Redux*, se realiza una conexión entre el estado de la aplicación, inicializado en los distintos reductores, con los componentes de la vista.

El diseño de los artefactos correspondientes a los ficheros de acción, reductores y utilidades se ha explicado de forma general tanto aquí como el apartado de 4.4 Arquitectura cliente. En el Anexo IV: Detalle de diseño estático en el cliente se muestran esquemas y explicaciones más detalladas.

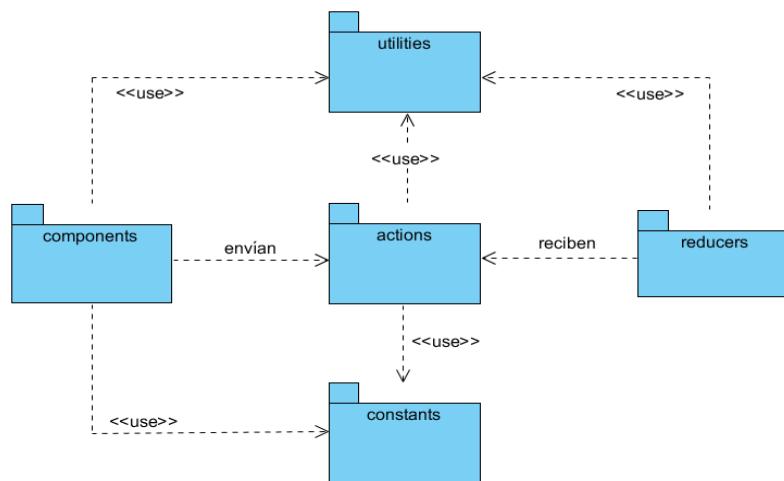


Figura 16: Diagrama de paquetes simple de la aplicación cliente

#### 7.1.2.2. Diagrama de clases de los componentes de la vista

Los componentes de la vista se presentan como clases, mientras que las acciones y los reductores son funciones. En la Figura 17 se puede ver el diagrama de clases de la vista dividiéndolo en componentes de utilidad, listas y pantallas de detalle. También se utilizan paquetes con constantes y utilidades.

Los componentes de la vista se pueden dividir en cuatro tipos en base a su uso en la aplicación. Por un lado, existen unos componentes base, en los que se contienen las demás pantallas de la aplicación. Los componentes de listas y de detalle corresponden con las pantallas de listados de elementos y de detalle de estos. Por último, los componentes utilidad representan pequeños elementos que son utilizados por los demás componentes de la aplicación.

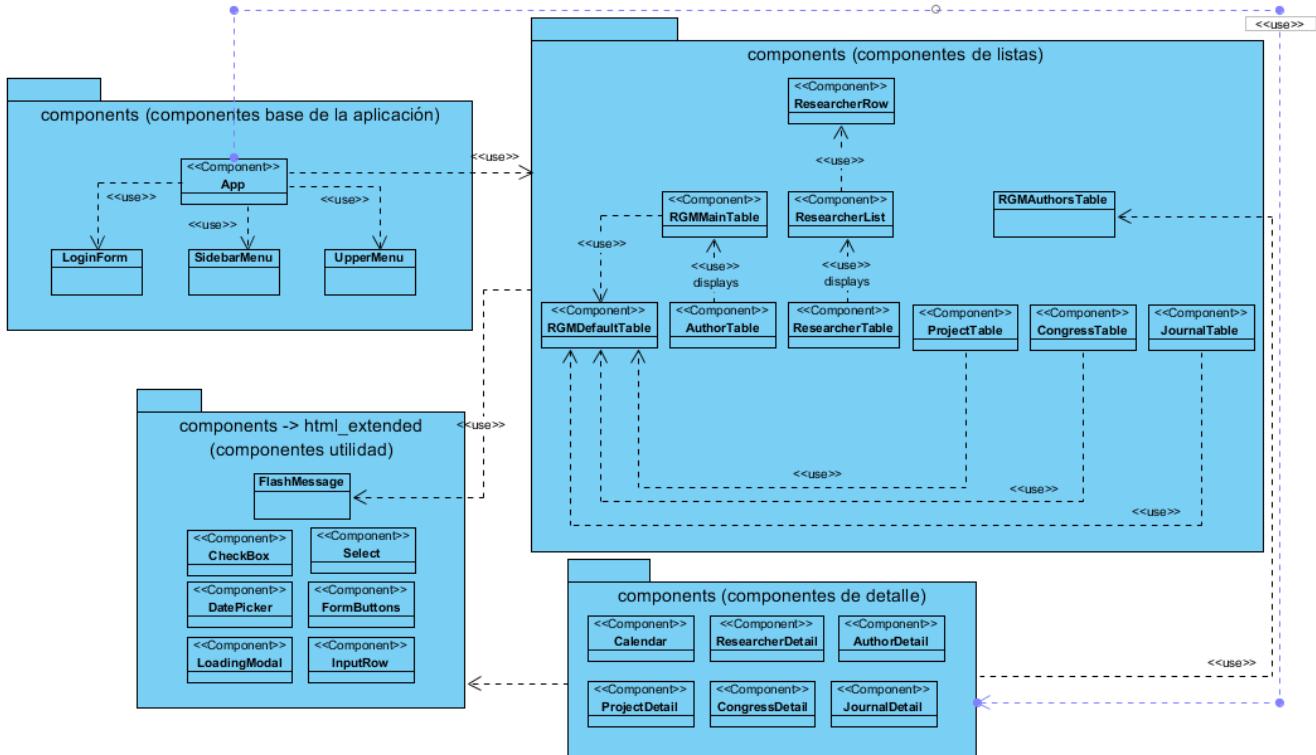


Figura 17: Diagrama de clases de los componentes del cliente

### 7.1.2.2.1. Componentes base

El paquete de componentes base contiene los que representan a la página base de la aplicación.

- App: se ocupa de renderizar todos los demás componentes de la aplicación. Este componente muestra, si el usuario no está identificado, el componente LoginForm y en caso contrario siempre renderiza los componentes SidebarMenu y UpperMenu.
- LoginForm: renderiza el formulario de identificación del usuario en la aplicación.
- UpperMenu: renderiza la barra superior.
- SidebarMenu: renderiza la barra lateral.

### 7.1.2.2.2. Componentes utilidad

Estos componentes son componentes que se reutilizan continuamente en otros de la vista. A continuación se explica la responsabilidad de cada uno de ellos:

- CheckBox: abstracción de un *checkbox* de HTML para adaptarlo a la aplicación cliente.
- Select: abstracción de un *select* de HTML para adaptarlo a la aplicación cliente.
- DatePicker: abstracción de un *input* HTML para adaptarlo a la selección de fechas.
- InputRow: abstracción de un *input* HTML para adaptarlo a la introducción de datos en formularios.
- LoadingModal: *popup* que muestra un mensaje de carga.
- FormButtons: botones de añadir, cancelar y eliminar adaptados a la aplicación.
- FlashMessage: mensaje flash personalizado que se muestra en los listados cada vez que se ejecuta una operación de inserción, actualización o borrado con éxito.

### 7.1.2.2.3. Componentes de listas:

- ResearcherTable: renderiza la pantalla de listado de investigadores.
- ResearcherList: renderiza el conjunto de filas del listado de investigadores.
- ResearcherRow: renderiza la fila de un investigador concreto.
- AuthorTable: renderiza la tabla de autores.
- RGMMainTable: componente reutilizable para renderizar tablas y utilizado por *AuthorTable*.
- RGMDefaultTable: componente reutilizable para renderizar tablas.
- ProjectTable: renderiza la pantalla de proyectos apoyándose en el componente *RGMDefaultTable*.
- CongressTable: renderiza la pantalla de congresos apoyándose en el componente *RGMDefaultTable*.
- JournalTable: renderiza la pantalla de publicaciones en revista apoyándose en el componente *RGMDefaultTable*.
- RGMAuthorsTable: renderiza una tabla con los datos de los autores. Se utiliza en varias pantallas de detalle.

### 7.1.2.2.4. Componentes de detalle:

- Calendar: renderiza un calendario o agenda.
- ResearcherDetail: renderiza el detalle de un investigador.
- AuthorDetail: renderiza el detalle de un autor y también su investigador asociado, si procede.
- ProjectDetail: renderiza el detalle de un proyecto y también sus autores asociados mediante el componente *RGMAuthorsTable*.
- CongressDetail: renderiza el detalle de un congreso y también sus autores asociados mediante el componente *RGMAuthorsTable*.
- JournalDetail: renderiza el detalle de una publicación en revista y también sus autores asociados mediante el componente *RGMAuthorsTable*.

## 7.2. Diseño dinámico

En esta parte se mostrarán únicamente los diagramas de secuencia más importantes de los casos de uso del sistema. Si se desea profundizar más en el diseño dinámico, se pueden consultar el resto de diagramas en el Anexo V: Diagramas de secuencia detallados.

### 7.2.1. Convenciones

Con el fin de facilitar la correcta comprensión de los diagramas de secuencia presentados en esta sección, se establecen una serie de convenciones relativas a las librerías y *frameworks* utilizados.

#### 7.2.1.1. Enrutamiento

En la aplicación, tal y como se ha comentado en la sección 4.2.1. Arquitectura, se utilizan dos librerías de enrutado, una en el cliente, con la librería *React-router* y otra en el servidor mediante la utilizad de enrutado que proporciona *Play Framework*.

Ambas partes proporcionan una forma de definir rutas. La librería cliente mediante un componente `<Router />` y el *framework* utilizado en el servidor definiéndolas en un fichero. Estas rutas son controladas por las propias tecnologías y dirigen las llamadas a la clase o componente correspondiente, el cual se ocupa de gestionarlas.

Debido a que el enrutado se lleva a cabo de forma interna mediante las tecnologías utilizadas en la aplicación, estos enruteadores no se presentan en los diagramas de secuencia, en los cuáles se muestran directamente las llamadas dirigidas al controlador o componente que debe gestionarlas.

Cuando, en los diagramas, se muestren flechas que representen una llamada realizada desde el navegador, solo se mostrará el método HTTP (GET, POST o DELETE) cuando esta llamada llegue realmente al servidor. En caso de ser interceptada por el *router* cliente se mostrará solo la URL utilizada.

### **7.2.1.2. Componentes de la vista**

En algunos componentes de la vista se utilizarán estereotipos para representarlos. Estos estereotipos son <<component>>, representando a los componentes de la vista y <<reducer>>, que representa a los reductores, elementos que reciben acciones y actualizan el estado global de la aplicación.

Tal y como se ha comentado en la sección 4.2.1 Arquitectura, cuando un reductor recibe una acción, este puede provocar un cambio en el estado de la aplicación. Cada vez que el estado cambia, la vista se renderiza de nuevo para mostrar esas modificaciones.

### **7.2.1.3. Creación de diagramas de secuencia auxiliares**

Algunos de los casos de uso realizan operaciones que se repiten en múltiples lugares. Esto, unido con el grosor de la arquitectura tanto en el servidor como en el cliente hacen que varios diagramas de secuencia sean difíciles de representar de forma clara y concisa y se puedan ver de forma sencilla. Por esta razón, se ha decidido crear algunos diagramas adicionales que se utilizan en los diagramas de secuencia principales. Si se desea, se pueden consultar de forma detallada estos diagramas en el Anexo V: Diagramas de secuencia detallados.

### **7.2.1.4. Convenciones adicionales**

Los elementos de la parte del cliente se definen en color azul, mientras que los de la parte del servidor se mostrarán en color verde.

### 7.2.1.5. Diagrama de secuencia de consultar estado global

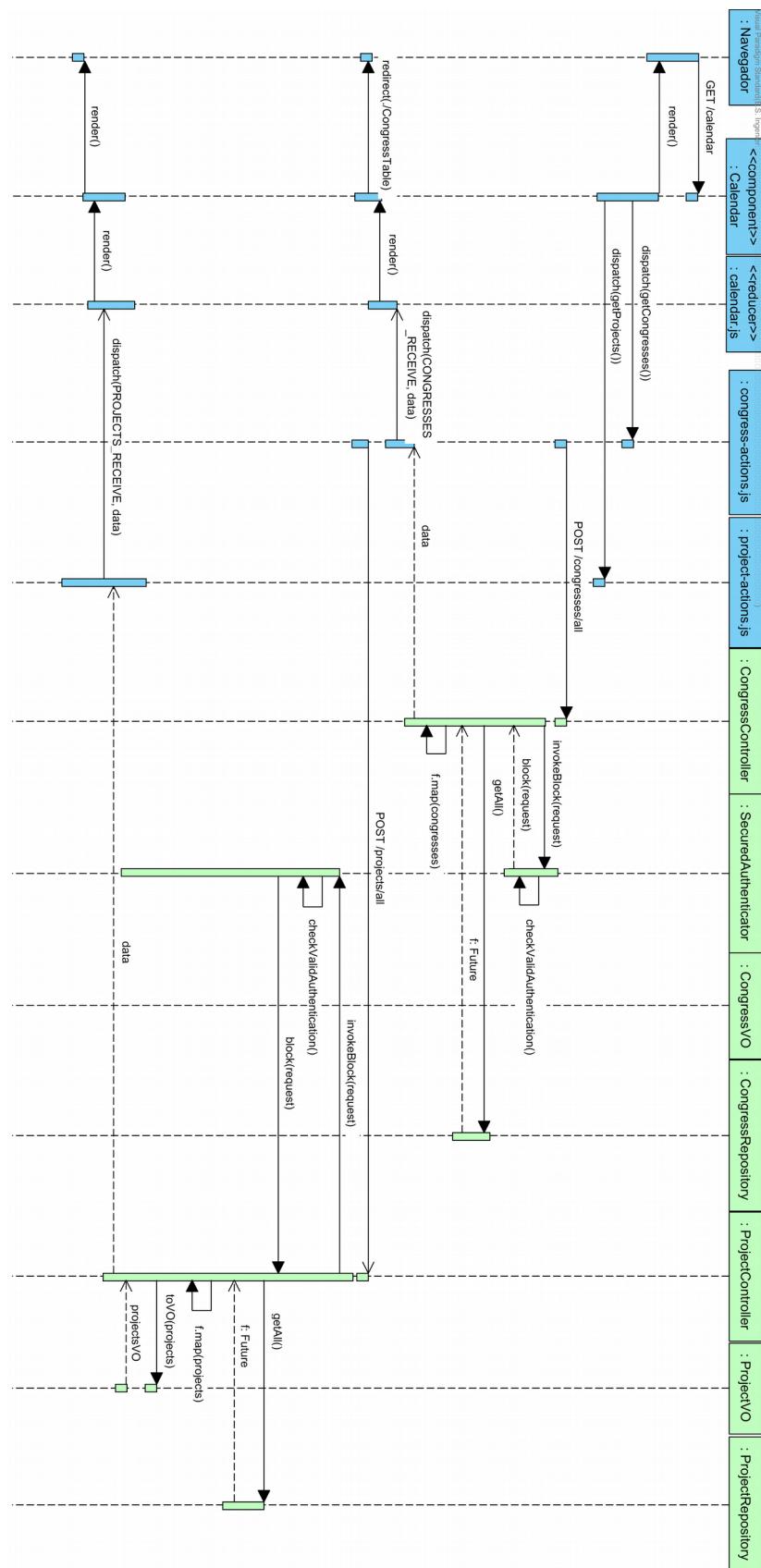


Figura 18: Diagrama de secuencia de consultar estado global

### 7.2.1.6. Diagrama de secuencia de consultar lista de investigadores

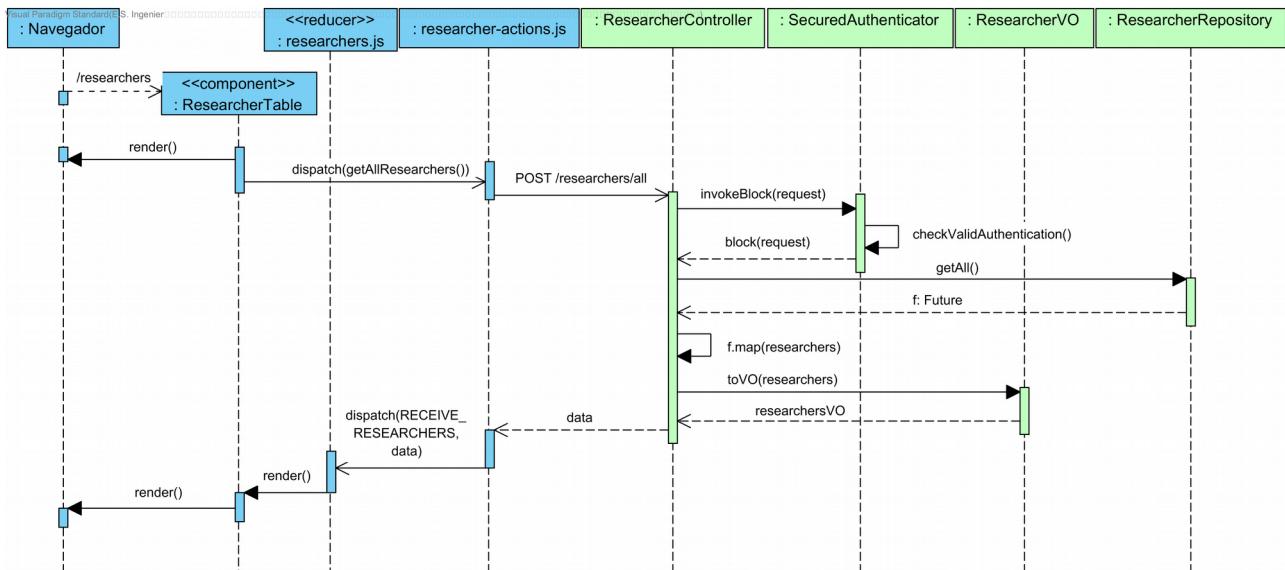


Figura 19: Diagrama de secuencia de ver lista de investigadores

### 7.2.1.7. Diagrama de secuencia de añadir investigador

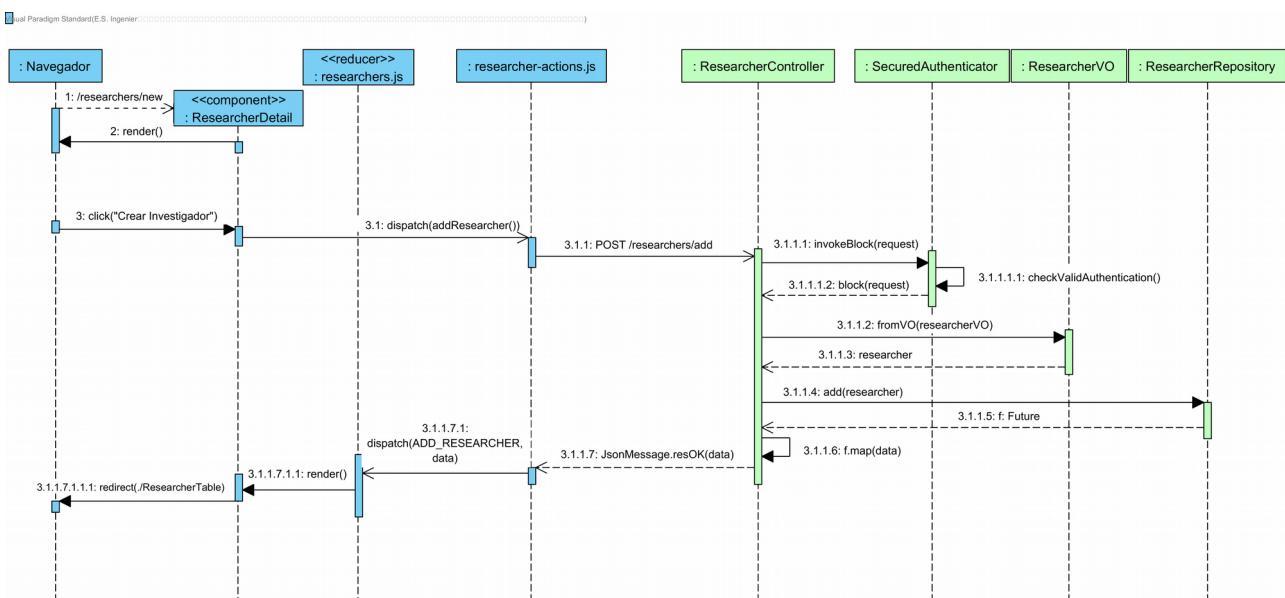


Figura 20: Diagrama de secuencia de añadir investigador

### 7.2.1.8. Diagrama de secuencia de actualizar congreso

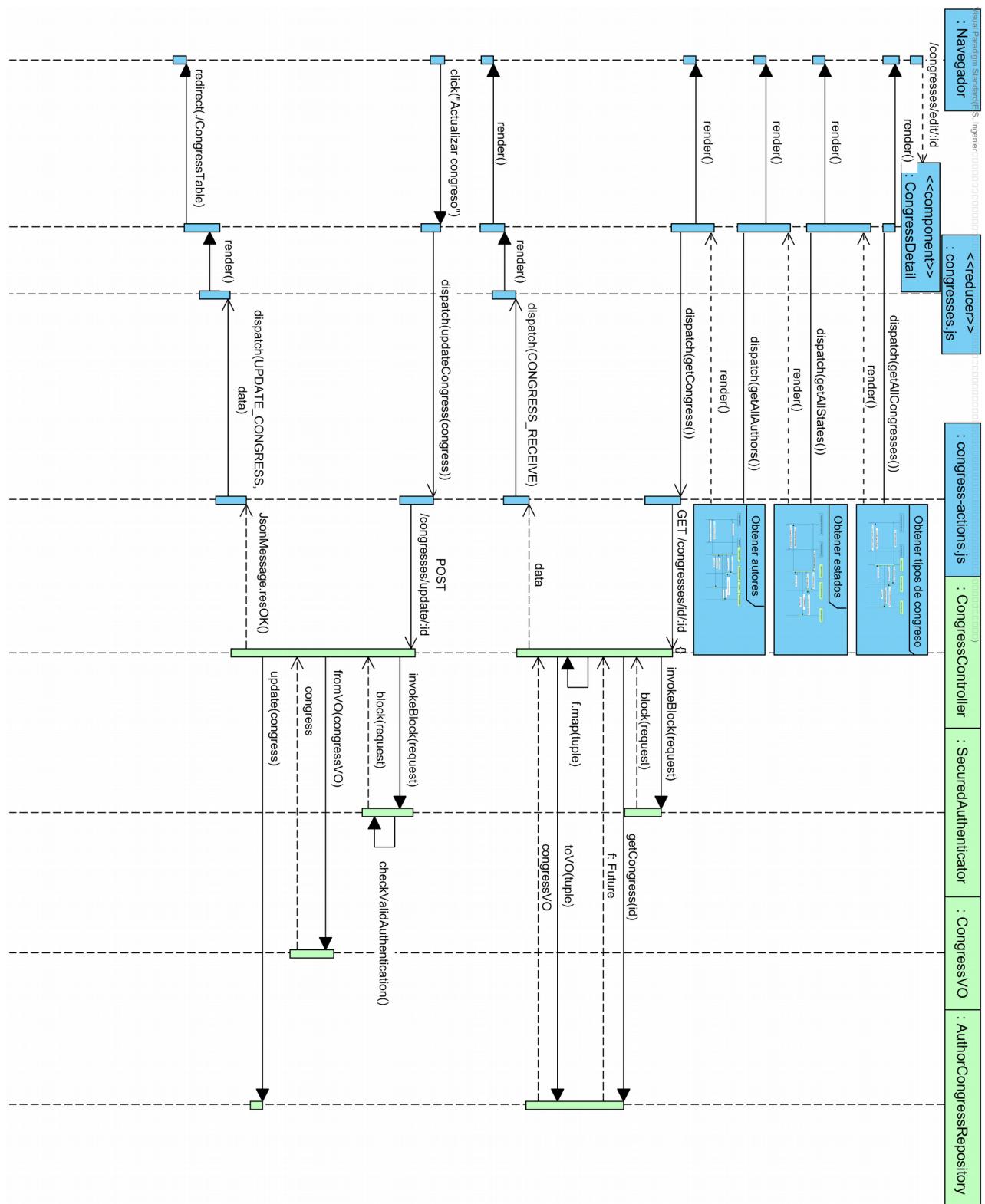


Figura 21: Diagrama de secuencia de actualizar congreso

### 7.2.1.9. Diagrama de secuencia de eliminar congreso

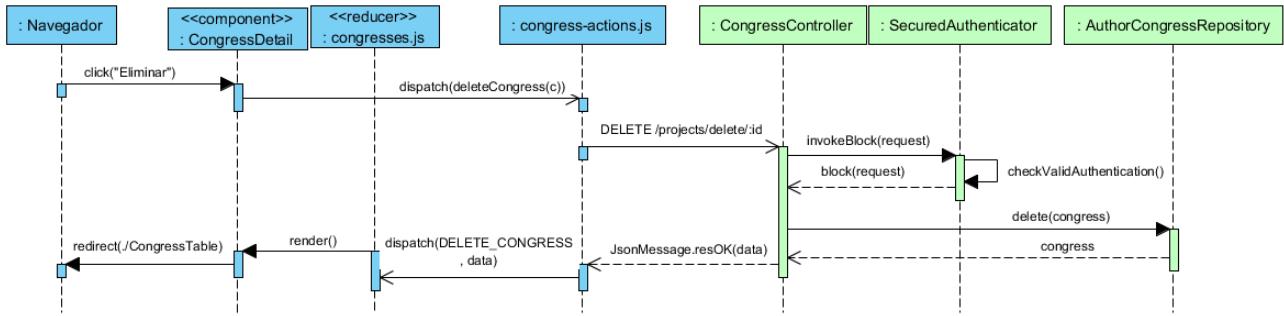


Figura 22: Diagrama de secuencia de eliminar congreso

## 8. Gestión de datos e información

### 8.1. Diagrama Entidad-Relación

En esta sección se recoge la información extraída en el análisis de datos del sistema. En este análisis se presentan las entidades necesarias para su realización así como los atributos de cada una de estas entidades.

En la Figura 23 se puede ver el diagrama entidad-relación obtenido con la realización de este proceso:

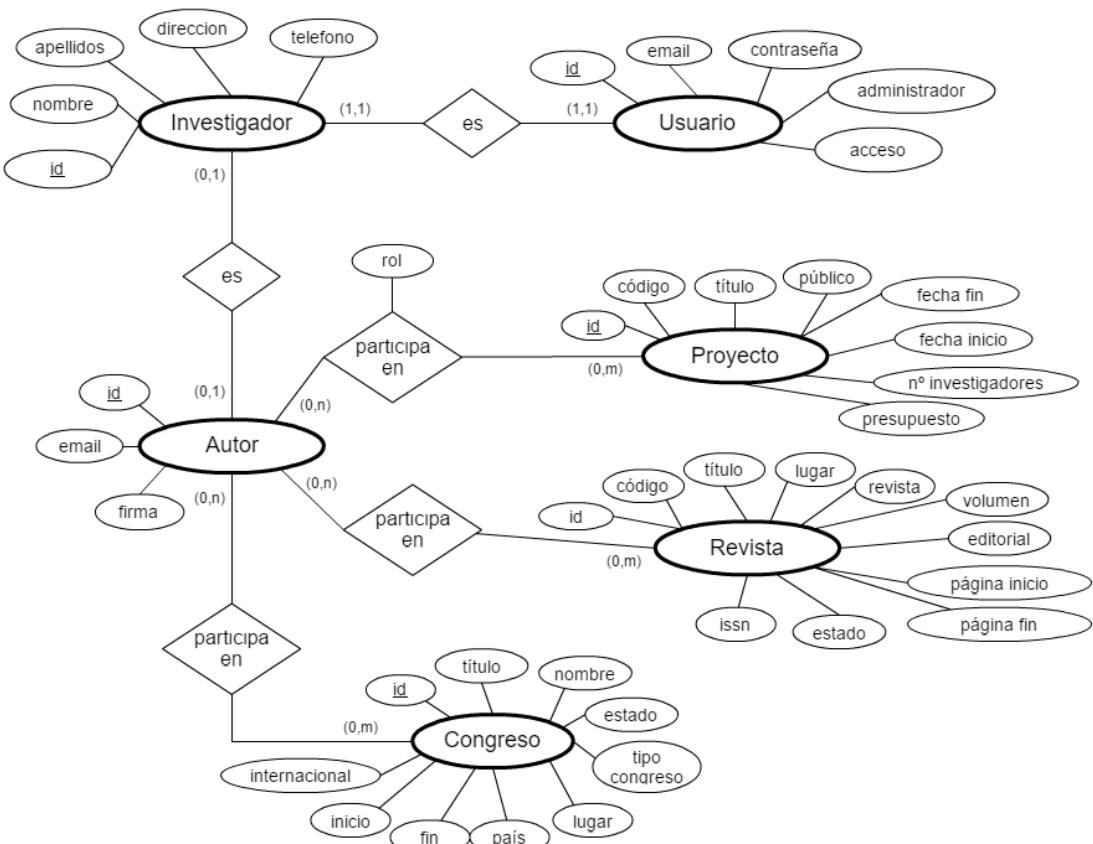


Figura 23: Diagrama entidad-relación

## 8.2. Diccionario de datos

En esta subsección se describen las distintas entidades del modelo entidad-relación y sus atributos.

### 8.2.1. Entidad Usuario

La entidad Usuario representa a un usuario de la aplicación y tiene una cantidad muy pequeña de datos, siendo identificados por su id o su email, el cuál no puede repetirse.

- Email: este atributo no puede repetirse. El email es utilizado por el usuario para identificarse en la aplicación.
- Contraseña: contraseña para identificarse en la aplicación. Mediante el email y este atributo el usuario puede acceder.
- Administrador: toma valores cero o uno. En caso de tener valor uno el usuario se presentará con permisos de administrador, en caso contrario tendrá permisos de investigador.
- Acceso: Sus valores están acotados entre cero y uno. Representa el permiso de acceso del usuario a la aplicación. Si se desea restringir el permiso a diversos investigadores a la aplicación este atributo debe marcarse con el valor cero.

### 8.2.2. Entidad Investigador

Esta entidad tiene relación directa con la tabla Usuario, de manera que cuando un investigador nuevo se registre en la aplicación, se le asociará un usuario con sus datos correspondientes. Los atributos de esta entidad son los siguientes:

- Nombre: nombre del investigador.
- Apellidos: apellidos del investigador.
- Dirección: dirección en la que vive el investigador.
- Teléfono: teléfono del investigador.

### 8.2.3. Entidad Autor

Esta entidad representa un autor de alguna publicación, congreso o proyecto. Un autor puede ir asociado con los datos de un investigador concreto o ser independiente. Los atributos de un autor son los siguientes:

- Email: correo electrónico del autor.
- Firma: texto con el que firma el autor.

### 8.2.4. Entidad Proyecto

Esta entidad representa un proyecto del mundo real en el que pueden participar una serie de autores. Los atributos de un proyecto son los siguientes:

- Código: código identificativo de la aplicación.
- Lugar: lugar donde se desarrolla el proyecto.
- Público: determina si el proyecto es público o privado. El valor está acotado entre cero para privado y uno para público.
- Fecha inicio: fecha de inicio del proyecto.
- Fecha fin: fecha límite de entrega del proyecto.
- N.º investigadores: número de investigadores que participan en el proyecto.
- Presupuesto: presupuesto final del proyecto.

### 8.2.5. Entidad Revista

Esta entidad representa una publicación en una revista. Una revista puede tener cualquier número de autores asociados. A continuación se presentan los atributos de esta entidad:

- Código: código de la publicación.
- Título: título de la publicación.
- Lugar: lugar de la publicación.
- Revista: revista en la que se ha publicado.
- Volumen: volumen de la revista donde se ha publicado.
- Editorial: editorial de la revista.
- Página inicio: página de inicio de la publicación en la revista.
- Página fin: página final de la publicación.
- Estado: estado de la publicación. Puede tomar los valores de justificado, aceptado o enviado.
- ISSN: código internacional asociado a la publicación.

### 8.2.6. Entidad Congreso

Esta entidad representa una ponencia o un póster en un congreso. Un congreso puede tener una serie de autores asociados. A continuación se presentan los atributos de esta entidad:

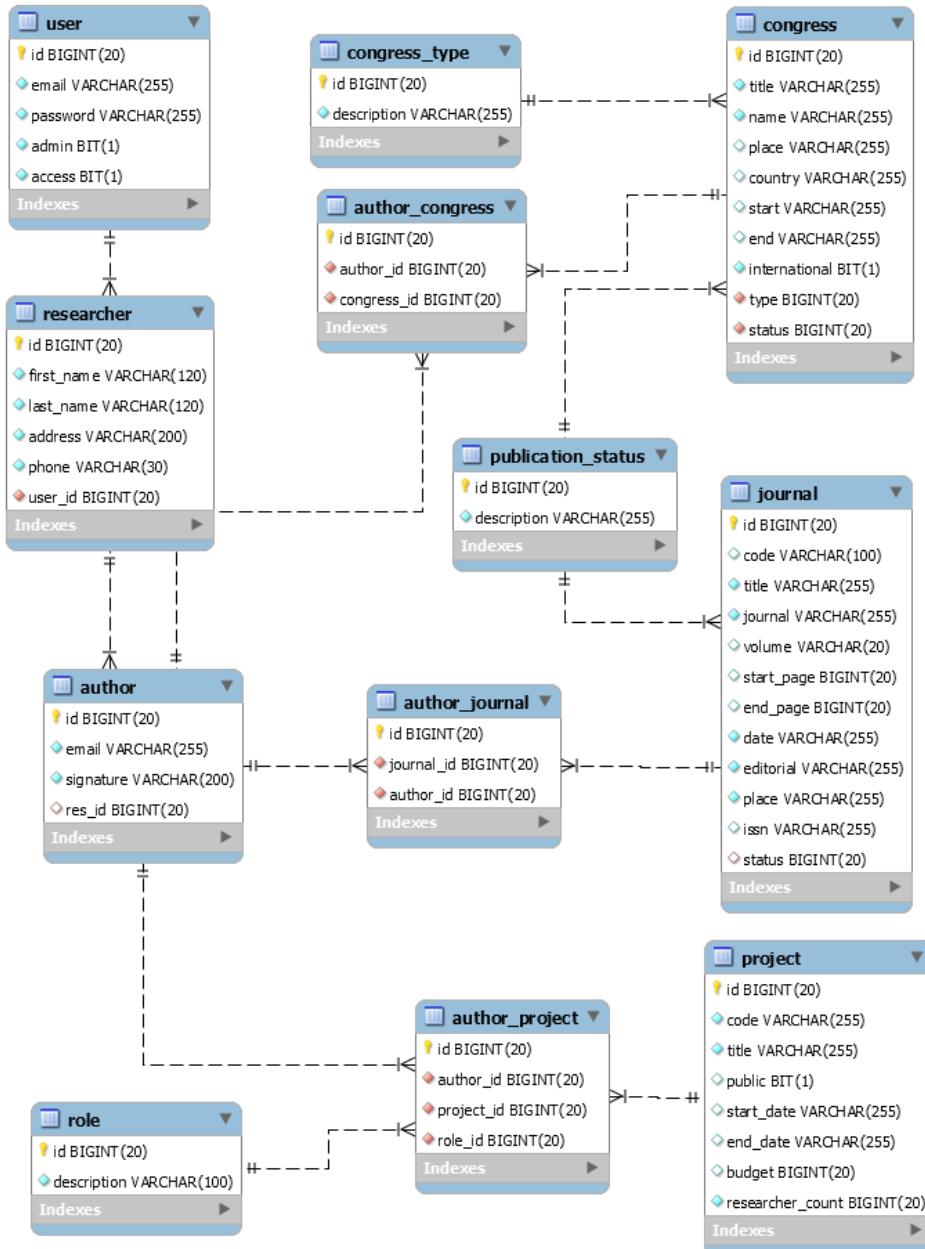
- Título: título de la publicación.
- Nombre: nombre del congreso.
- Estado: estado del congreso. Puede tomar los valores de justificado, aceptado o enviado.
- Tipo congreso: tipo de congreso. Puede tomar los valores de ponencia o póster.
- Lugar: lugar donde se realiza el congreso.
- País: país donde se realiza el congreso.
- Inicio: fecha de inicio del congreso.
- Fin: fecha de fin del congreso.
- Internacional: Indica si un congreso es o no internacional. En caso de ser internacional toma el valor uno, en caso contrario el cero.

### 8.3. Diagrama de tablas

En esta subsección se explica el paso a tablas del análisis de datos inicial que ha dado lugar al modelo entidad-relación.

Como se puede apreciar en la Figura 24, el número de tablas es bastante superior al de las entidades del modelo inicial. Esto se debe tanto a la existencia de diversas relaciones múltiples (n a n), como a la de atributos cuyo valor está acotado a varios valores predefinidos.

Las relación de un autor con los congresos, revistas o proyectos es que los primeros pueden asistir o participar en tantos de los segundos como se decida, y en estos últimos puede participar un número cualquiera de autores. Esto exige tablas intermedias en cada una de las relaciones.



*Figura 24: Modelo de datos*

La existencia de los “tipos de congreso” y “estados de una publicación” están también restringidos a varios predefinidos, es decir, no se permite al usuario insertar el tipo u estado que desea, sino que se le obliga a elegir entre unos dados. Para convertir estos tipos y estados en seleccionables en la interfaz de usuario se opta por la separación en tablas y consulta por parte de la aplicación para permitir al usuario la selección correspondiente.

## 9. Pruebas

Aquí se presentan una serie de descripciones de las pruebas de aceptación que se han realizado para comprobar el correcto funcionamiento del sistema. Si se desea, en el Anexo VI: Pruebas detalladas se pueden consultar estas mismas pruebas de forma más detallada.

### 9.1. Escenario de identificación

Se contempla la identificación de un usuario en el sistema de forma correcta. Para ello se introducen las credenciales de manera correcta y comprueba que el usuario es redirigido a la página de calendarios. Se hace el mismo proceso con credenciales erróneas, con las que se muestra un mensaje de error.

### 9.2. Escenario de salida del sistema

Se comprueba que el usuario identificado pulse en el botón de “salir”. En este caso, debe ser redirigido a la página de identificación.

### 9.3. Escenario de creación y actualización de un investigador

Se realizan pruebas de añadido o edición de un investigador, para ello el usuario debe estar identificado y tener permisos de administrador. Se realiza una prueba cubriendo todos los datos del formulario de forma correcta y otra utilizando contraseñas diferentes.

En la primera prueba el usuario es redirigido a la página de listado de investigadores, mientras que en la segunda se muestra un mensaje de error.

### 9.4. Eliminación de un investigador

Un usuario con permisos de administrador accederá a un investigador concreto que no sea el principal de la aplicación. Se pulsa en el botón eliminar y el usuario es redirigido a la página de listado de investigadores mostrando un mensaje de éxito.

### 9.5. Creación/actualización de un autor

Esta prueba se realizará pulsando en la creación o edición de un autor. Se completan los campos con un email válido y un texto válido en la firma y se pulsa en “añadir”. El usuario es redirigido a la lista de autores con un mensaje de éxito.

### 9.6. Eliminación de un autor

Un usuario administrador selecciona un autor y, una vez en la pantalla de detalle pulsa en el botón eliminar. El usuario es redirigido a la lista de autores mostrando un mensaje de éxito.

### 9.7. Inserción o modificación de un proyecto

Un usuario selecciona un proyecto sobre el que tenga permisos de edición o, en caso de ser administrador, crea un nuevo proyecto. Ser rellenan todos los datos correctamente y se pulsa en el botón de crear. El usuario es redirigido a la lista de proyectos con un mensaje de éxito.

## 9.8. Eliminación de un proyecto

El usuario selecciona un proyecto sobre el que tenga permisos de edición y pulsa en el botón “eliminar”. El usuario es redirigido a la lista de proyectos con un mensaje de éxito.

## 9.9. Inserción o modificación de un congreso

El usuario edita o crea un nuevo congreso y completa todos los campos correctamente. Una vez hecho esto se pulsa en el botón de creación o actualización del congreso. El usuario es redirigido a la lista de congresos con un mensaje de éxito.

Se realiza la misma prueba de creación o modificación dejando todos los campos vacíos y pulsando en crear. Se muestran una serie de errores en el formulario.

## 9.10. Eliminación de un congreso

El usuario selecciona un congreso sobre el que tenga permisos de edición y pulsa en “eliminar”. El usuario es redirigido a la lista de congresos con un mensaje de éxito.

## 9.11. Inserción o modificación de una publicación en revista

Un usuario administrador accede a la pantalla de creación de una publicación en revista. Se rellenan todos los datos y se pulsa en aceptar. El usuario debe ser redirigido a la lista de revistas con un mensaje de éxito.

Se realizará la misma prueba completando un número de página de inicio superior al de la página de fin. Una vez hecho esto el formulario muestra el error de forma descriptiva.

## 9.12. Eliminación de una publicación en revista

Un usuario con permisos de administrador accede a una publicación en revista y pulsa en “eliminar”. El usuario es redirigido a la pantalla de lista de publicaciones en revista con un mensaje de éxito.

## 9.13. Visualización de calendarios

El usuario accede a la sección “calendario” del menú lateral. En caso de existir proyectos y congresos con fechas en el sistema se mostrará un calendario del mes actual con los eventos asociados a su día.

# 10. Manuales de usuario

## 10.1. Manual de instalación

### 10.1.1. Requisitos mínimos

Para el despliegue de la aplicación deben disponerse, como mínimo, de las siguientes utilidades:

- Sistema operativo Ubuntu 16.04 o superior.
- MySQL 5.6 o superior.
- Disponer del JRE de Java en su versión 8 o superior y añadirlo al PATH.

### 10.1.2. Instalación

Para la instalación de la aplicación web en el servidor es necesario situarse en la carpeta “instalación” disponible en el CD y realizar las siguientes acciones:

Ejecutar el fichero “database.sql” mediante el siguiente comando:

```
mysql -u<user> -p < database.sql
```

- Instalar el paquete en el sistema operativo. Para esto debe ejecutarse el siguiente comando:

```
sudo dpkg -i GestorGrupoInvestigacion.deb
```

- Una vez instalada la aplicación, se puede acceder a esta mediante la URL <http://localhost:8008>. Una vez aquí y si todo es correcto se presentará la pantalla de la Figura 25, que sirve para identificarse como usuario. Las credenciales por defecto son las siguientes:
  - Usuario: admin@admin.es
  - Contraseña: 1234

### 10.2. Manual de utilización

En esta subsección se explica, mediante una navegación por las distintas pantallas de la aplicación, el uso global de esta.

Al acceder por primera vez a la aplicación, se muestra la pantalla de identificación presentada en la Figura 25, en la cuál deben introducirse el correo y contraseña del usuario.

En caso de detectarse algún error al intentar acceder se indicará con un mensaje en rojo, por el contrario, si la identificación tiene éxito se redirigirá a la pantalla principal de la aplicación, la cuál se corresponde con el calendario de actividades.

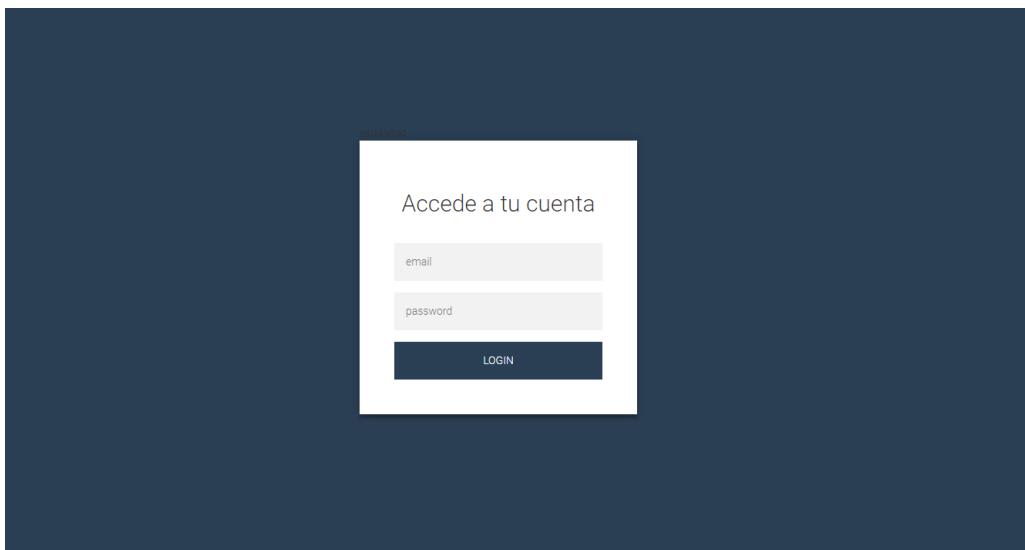


Figura 25: Pantalla de identificación

En la Figura 26 se presenta una imagen de la pantalla principal de la aplicación.

En esta pantalla se muestran dos secciones que estarán siempre presentes a lo largo de la aplicación, son las siguientes:

- Menú superior: se puede salir de la aplicación, la cual redirigirá al usuario de nuevo a la pantalla de identificación, o bien acceder al perfil del usuario.

- **Menú lateral:** desde aquí es posible acceder a todas las secciones de la aplicación.

En el centro de la pantalla siempre se mostrará el contenido de las distintas secciones. En este caso ese contenido son las distintas actividades de la aplicación, ordenadas por fecha y representadas en un calendario.

Figura 26: Pantalla de calendario de actividades

En la Figura 27 se pueden consultar las actividades de nuevo, pero ahora en forma de agenda (de esta forma no se ven los días vacíos). Para acceder a la agenda solo hay que pulsar en la parte superior derecha sobre el botón correspondiente.

Fecha	Evento
lun. jun. 26	Inicio title2
dom. jul. 02	Inicio title1
mié. jul. 05	TITULO CONGRESO 2
mar. jul. 11	Inicio TITULO CONGRESO 3
mié. jul. 12	Fin TITULO CONGRESO 3
mar. jul. 18	TITULO CONGRESO 1

Figura 27: Pantalla de agenda

Para acceder a la pantalla que se muestra en la Figura 28 debe pulsarse en la sección investigadores del menú lateral. Desde esta pantalla se pueden ver los principales datos de todos los investigadores y, a su vez, usuarios de la aplicación.

También, si se dispone de permisos de administrador, se puede acceder a la pantalla de creación de un nuevo investigador y usuario mediante el botón de Nuevo investigador así como ver en detalle los ya existentes.

The screenshot shows the RGM application's interface. On the left is a sidebar with a dark blue background containing user information ('Bienvenido, mnceleiro'), navigation links ('General', 'Calendario', 'Investigadores', 'Autores', 'Proyectos', 'Publicaciones' with 'Revistas' and 'Congresos' sub-options), and a 'Nuevo investigador' button. The main area displays a table of investigators with columns: Nombre, Apellidos, Dirección, Teléfono, and Correo. Each row has a 'Detalle' button. The data in the table is as follows:

Nombre	Apellidos	Dirección	Teléfono	Correo	Detalle
Marcos	Núñez Celeiro	Calle Empanada de Zorza nº5	9825312121	mnceleiro@esei.uvigo.es	<button>Detalle</button>
Oscar	Mixwell Cafielas	Calle Alcantara nº12	9825312122	ocanellas@gmail.com	<button>Detalle</button>
Dennis	MacAlistair Ritchie	Avenida Curros Enríquez 7	9825312123	dmritchie@yahoo.es	<button>Detalle</button>
Linus	Benedict Torvalds	Plaza Pontevedra 122	9825312124	ltorval@esei.uvigo.es	<button>Detalle</button>
Miguel	Reboiro Jato	Avenida de la Coruña 17	9825312125	mrjato@esei.uvigo.es	<button>Detalle</button>

Figura 28: Pantalla de lista de investigadores

En la Figura 29 se muestra la pantalla tanto de añadido como de edición y detalle de un investigador. En ella, si se dispone de permisos sobre ese usuario, es posible editar los datos del investigador en cuestión y, si el usuario identificado es administrador y se está realizando una edición es posible eliminarlo mediante el botón de la parte inferior derecha.

The screenshot shows the RGM application's detail view for an investigator. The sidebar is identical to Figure 28. The main area is titled 'Datos personales' and contains form fields for personal information: Email (mnceleiro@esei.uvigo.es), Contraseña, Confirmar contraseña, Nombre (Marcos), Apellidos (Núñez Celeiro), Dirección (Calle aleatoria nº5 2º), and Teléfono (9825312121). Below the form are two checkboxes: 'Acceso usuario' (checked) and 'Administrador' (checked). At the bottom are 'Guardar' and 'Volver' buttons, and a 'Eliminar investigador' button on the right.

Figura 29: Pantalla de detalle de un investigador

En el botón autores del menú lateral se dispone toda la gestión de estos. La pantalla principal de esta sección es la lista de autores, que se puede consultar en la Figura 30

Email	Firma	Investigador asociado	
mnceleiro@esei.uvigo.es	Marcos Núñez-Celeiro	Marcos Núñez Celeiro	<a href="#">Detalle</a>
ocanellas@gmail.com	Oscar Mixwell Cafielas	Oscar Mixwell Cafielas	<a href="#">Detalle</a>
dmitchie@yahoo.es	Dennis MacAlistair-Ritchie	Dennis MacAlistair Ritchie	<a href="#">Detalle</a>
torval@esei.uvigo.es	Linus Benedict-Torvalds	Linus Benedict Torvalds	<a href="#">Detalle</a>
mrjato@esei.uvigo.es	Miguel Reboiro-Jato	Miguel Reboiro Jato	<a href="#">Detalle</a>
mfowler@gmail.com	Martin Fowler		<a href="#">Detalle</a>

Figura 30: Pantalla de lista de autores

En la Figura 31 se puede ver la pantalla de detalle de un autor. En esta pantalla, además mostrarse los campos del propio autor, se permite asociarlo con un investigador concreto existente en la aplicación. Los datos del investigador se mostrarán debajo a modo orientativo pero sin permitir su edición.

Figura 31: Pantalla de detalle de un autor

Bajo el botón de acceso a la sección de autores, se muestra el de proyectos. Al pulsarlo se accede a la pantalla de la lista de proyectos, la cual se puede ver en la Figura 32. Esta pantalla es muy similar a los listados explicados anteriormente. En este caso la pantalla se presenta como ejemplo con un mensaje adicional en color verde, el cual indica que se procede de la pantalla de detalle, desde la cual se ha eliminado un proyecto de forma exitosa.

## TFG EI16/17-069

The screenshot shows the RGM application interface. On the left is a sidebar with icons for General, Calendario, Investigadores, Autores, Proyectos, Publicaciones (highlighted in blue), Revistas, and Congresos. The main area has a header with a user icon, 'mnceleiro', and 'Salir'. A green banner at the top says 'El proyecto ha sido eliminado correctamente.' Below it is a table with columns: Código, Título, Fecha de Inicio, Fecha de fin, Presupuesto, Investigadores, and Detalle (button). The table contains three rows: 2323 (title1), 2324 (title2), and 1212 (Proyecto de prueba).

Figura 32: Pantalla de lista de proyectos

La siguiente pantalla a comentar es la de detalle de un proyecto. Esta pantalla muestra los datos del proyecto así como sus autores asociados. Desde ella no se podrán crear nuevos autores o eliminar existentes, pero si se podrán asociar o quitar del proyecto autores que ya estén en la aplicación. En la Figura 33 podemos ver como es la pantalla de consulta y edición de un proyecto y como se presentan sus autores asociados.

The screenshot shows the 'Datos del proyecto' (Project Details) screen. It has a sidebar with the same menu as Figure 32. The main area has a header with a user icon, 'mnceleiro', and 'Salir'. The 'Datos del proyecto' section contains fields for Código (2323), Título (title1), Fecha de inicio (02/07/2017), Fecha de fin (03/10/2017), and Presupuesto (3000). Below this is a 'Autores asociados' (Associated Authors) section with two dropdown menus labeled 'Select...' and a 'Añadir' (Add) button. A table lists authors with columns: Email, Autor, rol, Investigador asociado, and Eliminar (Delete). The table shows two entries: 'mnceleiro@esei.uvigo.es' (Marcos Nunez-Celeiro, IP, mnceleiro@esei.uvigo.es) and 'ocanellas@gmail.com' (Oscar Maxwell Cañellas, IP, ocanellas@gmail.com). At the bottom are 'Guardar' (Save), 'Cancelar' (Cancel), and 'Eliminar proyecto' (Delete project) buttons.

Figura 33: Pantalla de detalle de un proyecto

En el listado de revistas, representado en la Figura 34, se puede ver el total de revistas almacenadas en el sistema así como acceder a las pantallas de añadido y detalle de estas.

Código	Título	Revista	Volumen	Editorial	Lugar
39887L	SOFTWARE FOR BIOIMAGING	JOURNAL OF SOFTWARE FOR BIOIMAGING	3:12	Intech	Ourense
A222324	Publicación de prueba	Revista de prueba	2:11	Editorial de ejemplo	Ourense

Figura 34: Pantalla de lista de revistas

La siguiente pantalla a explicar es la de consulta de una publicación en revista. En esta se muestran los datos de la misma manera que en un proyecto y se pueden, igualmente, añadir o quitar autores del mismo.

En la Figura 35 se puede ver el contenido de la pantalla. En este caso, se ha decidido mostrarla para un usuario sin permisos de edición, es decir, un investigador que, como autor, no está asociado a esa publicación y que no puede, por tanto, ni editarla ni eliminarla.

Email	Autor	Investigador asociado
mnceleiro@esei.uvigo.es	Marcos Nunez-Celeiro	mnceleiro@esei.uvigo.es

Figura 35: Pantalla de detalle de una revista

En la Figura 36 se presenta el listado de congresos. Esta vez no aparece el botón para añadir uno nuevo debido a que el usuario que está identificado no tiene permisos de administrador.

## TFG EI16/17-069

The screenshot shows the RGM application's interface. On the left is a sidebar with icons for general navigation: Bienvenido, General, Calendario, Investigadores, Autores, Proyectos, Publicaciones, Revistas, and Congresos. The main area displays a table titled 'Congreso' with columns: Título, Lugar, País, Inicio, and Fin. There are three rows of data: 'TITULO CONGRESO 1' (Lugar: OURENSE, País: ESPAÑA, Inicio: 18/07/2017, Fin: 18/07/2017), 'TITULO CONGRESO 2' (Lugar: OURENSE, País: ESPAÑA, Inicio: 05/07/2017, Fin: 05/07/2017), and 'TITULO CONGRESO 3' (Lugar: OURENSE, País: ESPAÑA, Inicio: 11/07/2017, Fin: 12/07/2017). Each row has a 'Detalles' button.

Figura 36: Pantalla de lista de congresos

En la Figura 37 se muestra la pantalla de consulta de un congreso concreto con un usuario sin permisos de administrador y que no está asociado al mismo. Esta pantalla es muy similar a la de proyectos y revistas y en ella también se pueden añadir y eliminar asociaciones entre los diferentes autores de la aplicación y el congreso que se está consultando.

The screenshot shows the RGM application's interface for viewing a specific congress. The sidebar is identical to Figure 36. The main area is divided into two sections: 'Datos del congreso' and 'Autores asociados'. The 'Datos del congreso' section contains fields for Titulo (TITULO CONGRESO 1), Nombre (NOMBRE CONGRESO 1), Lugar (OURENSE), País (ESPAÑA), Fecha de inicio (18/07/2017), Fecha de fin (18/07/2017), Estado (JUSTIFICADO), and Tipo (PONENCIA). A checkbox for 'Internacional' is checked. The 'Autores asociados' section lists two authors with their emails and associated investigator emails:

Email	Autor	Investigador asociado
mncelero@esei.uvigo.es	Marcos Nunez-Celeiro	mncelero@esei.uvigo.es
ocanellas@gmail.com	Oscar Mixwell Cañellas	ocanellas@gmail.com

Figura 37: Pantalla de detalle de un congreso

Por último, se ha decidido mostrar un formulario con todos los datos mal introducidos. En la Figura 38 se puede ver la edición de una revista una vez se ha pasado con el cursor por todos los campos.

En rojo se presentan los mensajes de error, de forma explicativa para el usuario, con los campos que es necesario editar antes de poder guardar el elemento. Estas validaciones se realizan en todos los formularios existentes en la aplicación.

Datos de la publicación (Revista)

Código:	Código Por favor, introduzca un código.	Título:	Título Por favor, introduzca un título.		
Revista:	Revista Por favor, introduzca el nombre de la revista	Página de inicio:	155 La página de inicio no puede ser mayor que la de fin.	Página de fin:	120 La página de inicio no puede ser mayor que la de fin.
Volumen:	Volumen	Fecha:	Fecha Por favor, introduzca la fecha de la revista	Editorial:	qq Este campo excede el número de caracteres permitido.
Lugar:	Lugar Por favor, introduzca un lugar	issn:	issn	Estado:	JUSTIFICADO

Autores asociados

Email	Autor	Investigador asociado	Eliminar
<input type="text" value="Select..."/> <input type="button" value="Añadir"/> <input type="button" value="Guardar"/> <input type="button" value="Cancelar"/>			

Figura 38: Pantalla de formulario con errores en la edición de una revista

## 11. Principales aportaciones

### 11.1. Repositorio centralizado de información

La aplicación desarrollada sirve como repositorio de información para que los miembros de un grupo de investigación puedan realizar cualquier consulta sobre los datos de los proyectos o publicaciones en los que están involucrados, tanto ellos como otros miembros del grupo de investigación.

Cada miembro, además, puede colaborar en cada uno de los proyectos o publicaciones en los que está implicado actualizando la información de estos elementos.

### 11.2. Simplicidad

La aplicación se ha diseñado de manera que sea sencilla e intuitiva para el usuario final. La interfaz de usuario es uniforme, todas las pantallas siguen el mismo patrón para las operaciones de gestión y los formularios presentan un control de errores claro y sencillo de entender.

### 11.3. Fluidez

Al acceder a la página por primera vez, se descargan una serie de elementos que se mantienen y utilizan durante toda la navegación del usuario en la página web. La aplicación está implementada como una SPA, esto permite que los elementos que se han descargado y las operaciones que se han realizado inicialmente no se hagan de nuevo cuando el usuario sigue navegando de forma normal por la página.

La consecución de esto es una sensación de fluidez para el usuario en la interacción con la aplicación.

## 11.4. Aplicación preparada para crecer

La aplicación del lado del servidor utilizan como base dos *frameworks*, Play para la gestión global de la aplicación y Slick para la comunicación con la base de datos. Estos *frameworks* son, en su campo, los más populares utilizados en lenguaje Scala y tienen un desarrollo muy activo. Play, además, no es solo una herramienta para gestionar llamadas al servidor, sino que proporciona utilidades para: transformar objetos a formato JSON, controlar la autenticación, utilizar filtros HTTP, controlar la evolución de la base de datos, etc.

En la aplicación del lado del cliente, por su parte, se utiliza como lenguaje JavaScript en su especificación de 2015, la cuál aún no implementan la mayoría de los navegadores y que se compila en la aplicación mediante Babel. También se utilizan librerías populares como son React o Redux así como una serie de herramientas que ayudan a asegurar una cierta calidad en el código.

La utilización de todos estos elementos favorece que la aplicación pueda ampliarse fácilmente en el futuro, ya que se usan tecnologías populares que favorecen la reutilización de código y cuyo desarrollo se mantiene muy activo. Además, el código del servidor y el cliente es independiente, lo cuál es útil en este aspecto ya que cualquier cambio que se realice en una parte no tiene por qué implicar cambios en la otra.

# 12. Conclusiones

## 12.1. Conclusiones personales

El trabajo realizado ha sido muy interesante de cara a mi aprendizaje y mejora profesional. En este, se han utilizando una gran cantidad de tecnologías antes desconocidas para mí pero muy populares actualmente e interesantes para su uso a nivel profesional.

He podido aprender a automatizar ciertas tareas y a asegurar una cierta calidad de código utilizando tecnologías para gestión de dependencias, bloqueo de subidas a control de versiones o automatización de tareas de compilación. He aprendido mucho de programación funcional, que antes casi no utilizaba y he también un lenguaje que me ha parecido muy bueno, que es Scala.

La realización del proyecto ha sido muy didáctica, entretenida, a veces divertida y otras un poco frustrante. He utilizado una gran cantidad utilidades entre librerías y *frameworks*, y todas estas tecnologías fueron totalmente nuevas para mí. Considero también que he infravalorado la curva de aprendizaje de algunas, como han sido Play y Slick.

La realización del proyecto a nivel personal ha sido muy satisfactoria, ya que he podido aprender cosas que no había visto ni estudiando ni trabajando pero que son muy interesantes tanto a nivel didáctico como a nivel profesional.

## 12.2. Conclusiones técnicas

Considero que las tecnologías utilizadas están bien escogidas de cara a la mejora futura del proyecto. Tanto Play como Slick son *frameworks* con un desarrollo muy activo, de hecho son los más populares relacionados con Scala, y cuya inicial complejidad ha sido enorme, pero que aportan grandes posibilidades de cara a la mejora futura del proyecto pudiendo así encarar cualquier necesidad que pueda surgir en una ampliación de este.

En el cliente se ha decidido utilizar React, la cuál, al contrario que las tecnologías elegidas en el servidor o que otras muy populares existentes en cliente como podría ser AngularJS, tiene una curva de aprendizaje sencilla. React es una librería que según las necesidades del proyecto se puede ampliar con muchas otras y hay que decidir cuál es mejor en cada caso.

Las ampliaciones sobre esta librería se hacen normalmente en pequeñas partes y por tanto la toma de decisiones puede aplazarse e irse realizando a lo largo del desarrollo. Esto ha facilitado que a lo largo de las iteraciones y según el tiempo restante para la realización del trabajo se puedan seguir incluyendo tecnologías que sean útiles para el proyecto sin tener que decidir todo al inicio, como se ha hecho en la parte del servidor.

## 13. Vías de trabajo futuro

Este trabajo se ha realizado de manera que la aplicación pueda crecer ampliamente en un futuro y con gran facilidad.

Se han utilizado en servidor *frameworks* de amplia complejidad, cuyo desarrollo se mantiene muy activo y que son, por tanto, muy completos. Esto permite que muchos requisitos, tanto funcionales como no funcionales, sean más sencillos y posibles de cumplir en un futuro. Por otro lado, en el cliente, se utilizan librerías que mejoran la encapsulación y que ayudan a mantener cierta calidad en el código a lo largo del tiempo.

Todo esto permite que el proyecto pueda crecer de manera útil y el código sea sencillo de refactorizar y mejorar según se van añadiendo nuevas funcionalidades.

### 13.1. Mejoras funcionales

#### 13.1.1. Gestión de publicaciones en revista

La parte que se considera más importante de implementar en un futuro es la gestión de publicaciones en revista. Esta sección, aunque dispone de un funcionamiento básico de consulta, añadido, actualización y borrado de estas, no tiene actualmente la posibilidad de seleccionar categorías y manejar los factores de impacto.

#### 13.1.1.2. Gestión de congresos

Por otro lado sería interesante, aprovechando la gestión de publicaciones en revista, realizar una refactorización parcial del código relacionado con estas para añadir a la aplicación la posibilidad de gestionar también congresos. Esta funcionalidad sería muy similar a la de gestión de publicaciones en revista y sería, por lo tanto, sencilla de implementar.

#### 13.1.1.3. Integración con Google Calendar

Se plantea la opción para el usuario de integrar el calendario de actividades disponible en la aplicación con *Google Calendar*. Esto podría hacerse mediante un botón disponible en la propia sección de “calendarios”.

Se debería investigar la posibilidad de mantener el calendario sincronizado de manera que cada cambio que se realice en la aplicación se modifique en *Calendar*. También debería investigarse la posibilidad mediante el API de *Calendar* de mantener calendarios individuales que solo muestren los eventos que impliquen al usuario identificado en la aplicación.

## 13.2. Mejoras de usabilidad

A nivel de diseño de la interfaz de usuario se considera importante mejorar la usabilidad de la aplicación mediante la separación o diferenciación de las entidades relacionadas con el usuario identificado de las del resto del sistema.

De esta manera, se consigue que un usuario identificado pueda ver de forma rápida las publicaciones y eventos en los que participa. Esto podría hacerse, o bien separando los listados en dos tablas en las cuales una tendría que ver con el usuario de la aplicación, o bien identificando estos listados con diferentes colores o tonos, de manera que el usuario pueda diferenciar a simple vista las publicaciones o eventos que implican al usuario directamente de las del resto de la aplicación.

### 13.3. Mejoras técnicas

A nivel técnico es interesante refactorizar el código realizado en Slick. En este *framework* no es posible realizar las operaciones y reutilización de código de la manera en que se hace en un *ORM*.

En la aplicación, la calidad del código de acceso a datos es peor que la del resto de la aplicación, debido a que dificultad de reutilizar código es muy alta en comparación con las demás tecnologías utilizadas. En esta librería muchas consultas deben hacerse con *joins* y muchas de las entidades consultadas de esta manera en base de datos tiene características muy similares. Por ejemplo, tanto congresos, como revistas y proyectos, contienen una tabla adicional para asociarse con sus autores.

Sería interesante estudiar en mayor profundidad el *framework* y refactorizar el código actual de manera que sea posible reutilizar una mayor cantidad en las operaciones de acceso a datos.

## 14. Referencias

### Bibliografía

- [1] Proceso Unificado De Desarrollo De Software, [ingsoftware072301.obolog.es/up-proceso-unificado-2010775](http://ingsoftware072301.obolog.es/up-proceso-unificado-2010775). Última consulta: 2 de Mayo de 2017.
- [2] Introduction to UML, UML. <http://www.uml.org/what-is-uml.htm>. Última consulta: de Junio de 2017.
- [3] Cliente-Servidor, Wikipedia. <https://es.wikipedia.org/wiki/Cliente-servidor>. Última consulta: 5 de Mayo de 2017.
- [4] Modelo-vista-controlador, Wikipedia. <https://es.wikipedia.org/wiki/Modelo%E2%80%93vista%E2%80%93controlador>. Última consulta: 24 de Abril de 2017.
- [5] What is Scala, Martin Odersky. <https://www.scala-lang.org/what-is-scala.html>. Última consulta: 5 de Mayo de 2017.
- [6] Handling form submission, PlayFramework. <https://www.playframework.com/documentation/2.5.x/ScalaForms>. Última consulta: 2 de Junio de 2017.
- [7] Redux: An architectural Style Inspired by Flux, Abel Avram. <https://www.infoq.com/news/2015/11/redux>. Última consulta: 3 de Junio de 2017.
- [8] What is React, James Padolsey. . Última consulta: 3 de Junio de 2013.
- [9] Flux, application architecture for building user interfaces, Facebook Inc.. <https://facebook.github.io/flux/>. Última consulta: 3 de Junio de 2017.
- [10] Play Framework, Wikipedia. [https://en.wikipedia.org/wiki/Play\\_Framework](https://en.wikipedia.org/wiki/Play_Framework). Última consulta: 5 de Junio de 2017.
- [11] Slick, Functional Relational Mapping for Scala, Slick. <http://slick.lightbend.com/>. Última consulta: 5 de Junio de 2017.
- [12] , Wikipedia. [https://es.wikipedia.org/wiki/Language\\_Integrated\\_Query](https://es.wikipedia.org/wiki/Language_Integrated_Query). Última consulta: 5 de Junio de 2017.
- [13] Scala Bcrypt, Yaroslav Klymko. <https://github.com/t3hnar/scala-bcrypt>. Última consulta: 5 de Junio de 2017.
- [14] Authentikat JWT, Jason Goodwin. . Última consulta: 5 de Junio de 2017.
- [15] Introduction to JSON Web Tokens, Auth0. . Última consulta: 5 de Junio de 2017.
- [16] Bill Venners, Artima Inc, . <http://www.scalatest.org/>. Última consulta: 5 de Junio de 2017.
- [17] SBT, Wikipedia. [https://en.wikipedia.org/wiki/Sbt\\_\(software\)](https://en.wikipedia.org/wiki/Sbt_(software)). Última consulta: 5 de Junio de 2017.
- [18] Creating DSLs in Java, Part 1: What is a domain-specific language?, Venkat Subramaniam. <http://www.javaworld.com/article/2077865/core-java/core-java-creating-dsls-in-java-part-1-what-is-a-domain-specific-language.html>. Última consulta: 12 de Junio de 2017.
- [19] HTML, Wikipedia. . Última consulta: 26 de Mayo de 2017.
- [20] Hoja de estilos en cascada, Wikipedia. . Última consulta: 24 de Mayo de 2017.
- [21] JavaScript, Wikipedia. <https://es.wikipedia.org/wiki/JavaScript>. Última consulta: 16 de Mayo de 2017.
- [22] React Router, React Training. <https://reacttraining.com/react-router/>. Última consulta: 5 de Junio de 2017.

- [23] SPA, un paradigma de Arquitectura de Aplicaciones Web en auge, Javier Tejero. <https://cink.es/blog/2013/10/07/spa-un-paradigma-de-arquitectura-de-aplicaciones-web-en-auge/>. Última consulta: 5 de Junio de 2017.
- [24] React Select, Jed Watson. <http://jedwatson.github.io/react-select/>. Última consulta: 5 de Junio de 2017.
- [25] React Big Calendar, International Justice Mission . <http://intljusticemission.github.io/react-big-calendar/>. Última consulta: 5 de Junio de 2017.
- [26] React Date Picker, HackerOne. <https://github.com/Hacker0x01/react-datepicker>. Última consulta: 5 de Junio de 2017.
- [27] Redux Thunk, Dan Abramov. <https://github.com/gaearon/redux-thunk>. Última consulta: 12 de Junio de 2017.
- [28] Getting Started with Redux-Form, Erik Rasmussen. <http://redux-form.com/6.7.0/docs/GettingStarted.md/>. Última consulta: 12 de Junio de 2017.
- [29] Moment, Iskren Ivov Chernev. <https://github.com/moment/moment>. Última consulta: 12 de Junio de 2017.
- [30] Sass (Syntactically Awesome StyleSheets), Hampton Catlin. [http://sass-lang.com/documentation/file.SASS\\_REFERENCE.html](http://sass-lang.com/documentation/file.SASS_REFERENCE.html). Última consulta: de de 2016.
- [31] jQuery, Wikipedia. <https://es.wikipedia.org/wiki/JQuery>. Última consulta: 5 de Junio de 2017.
- [32] What is bootstrap?, Brad Markle. <http://www.inmotionhosting.com/support/edu/joomla-3/using-bootstrap/what-is-bootstrap>. Última consulta: 5 de Junio de 2017.
- [33] Font Awesome, Dave Gandy. <https://github.com/FortAwesome/Font-Awesome>. Última consulta: 5 de Junio de 2017.
- [34] ¿Qué es npm?, Alexander Guevara Benites. <https://devcode.la/blog/que-es-npm/>. Última consulta: 5 de Junio de 2017.
- [35] Webpack Introduction, Webpack. <https://webpack.js.org/concepts/>. Última consulta: 2 de Junio de 2017.
- [36] What is Babel, and how will it help you write JavaScript?, Nicholas Johnson. <http://nicholasjohnson.com/blog/what-is-babel/>. Última consulta: 12 de Junio de 2017.
- [37] Actualiza el navegador al guardar cambios con Browser Sync, Iván Abascal Lozano. <https://browsersync.io/>. Última consulta: 15 de Junio de 2017.
- [38] About ESLint, JS Foundation. <http://eslint.org/docs/about/>. Última consulta: 6 de Junio de 2017.
- [39] Git: Introducción, Luciano Castillo. <http://conociendogithub.readthedocs.io/en/latest/data/introduccion/>. Última consulta: 13 de Junio de 2017.

# **Anexo I: Diagramas de Gantt**



## **Índice de Figuras**

Figura 1: Diagrama de Gantt de la planificación del desarrollo.....	1
Figura 2: Diagrama de Gantt del desarrollo real del proyecto.....	2



## Gestor de grupos de investigación

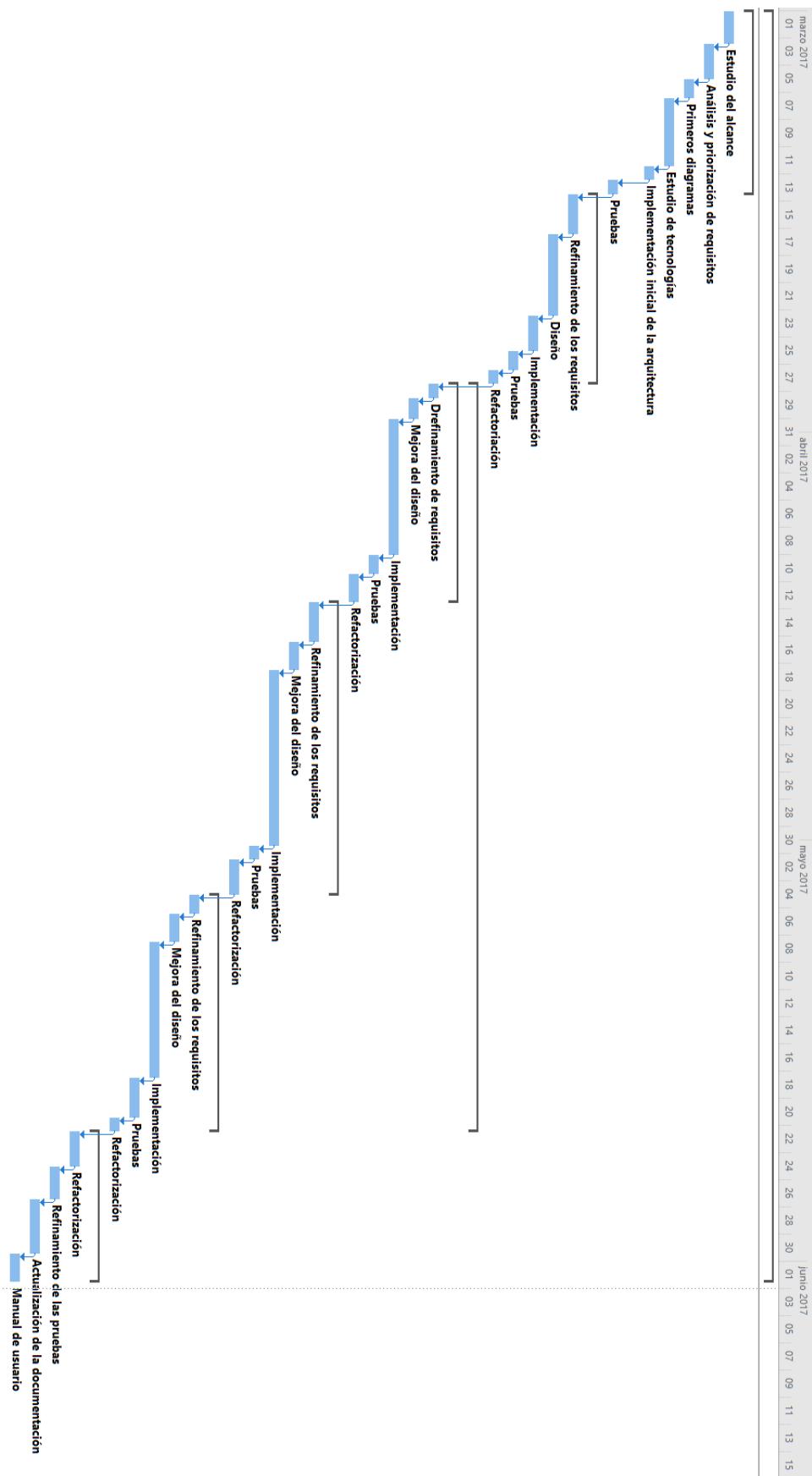


Figura 1: Diagrama de Gantt de la planificación del desarrollo

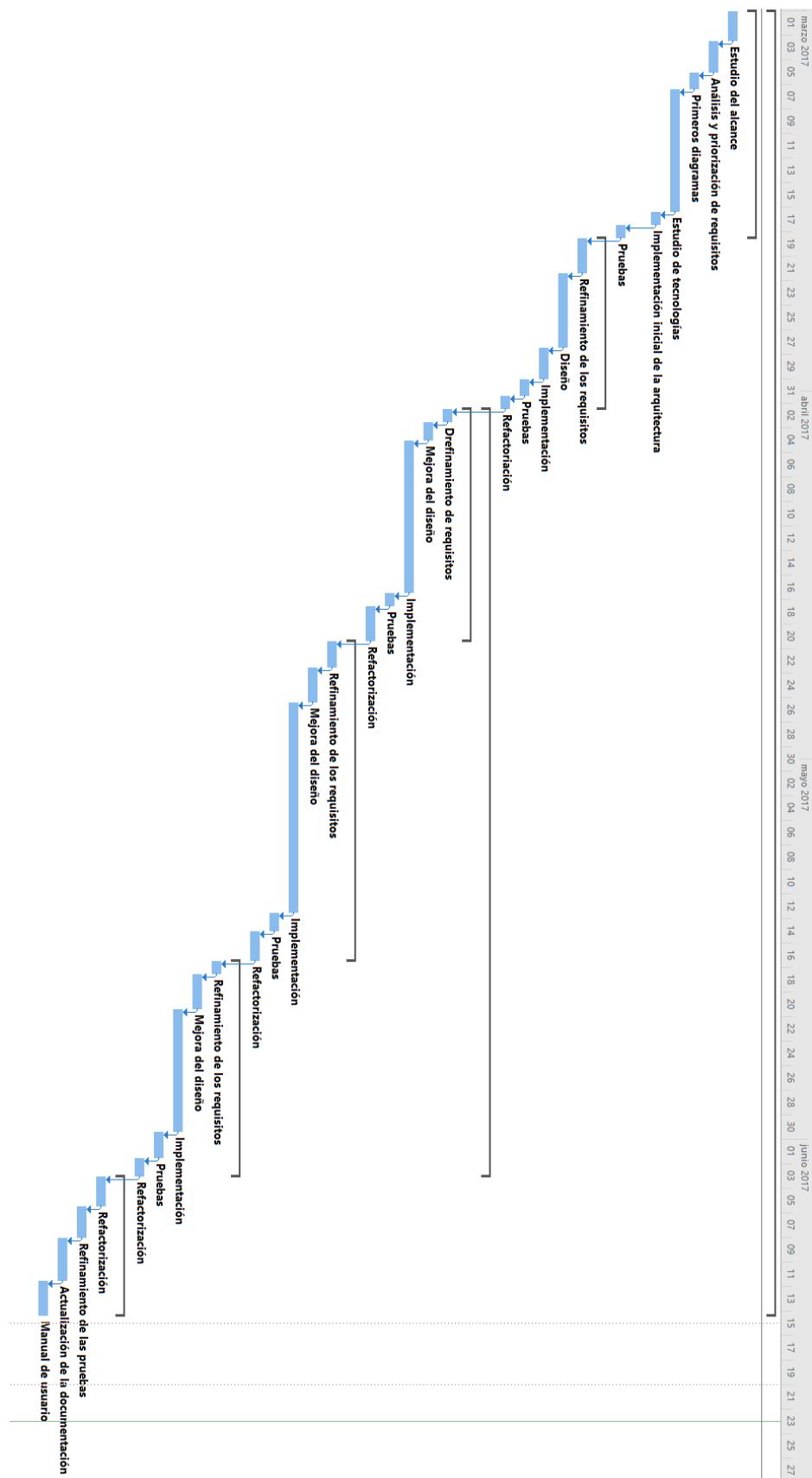


Figura 2: Diagrama de Gantt del desarrollo real del proyecto

# **Anexo II: Descripciones de casos de uso detallados**



## Índice de Tablas

Tabla 1: Descripción de caso de uso "consultar lista de autores".....	1
Tabla 2: Descripción de caso de uso "consultar autor".....	1
Tabla 3: Descripción de caso de uso "añadir autor".....	2
Tabla 4: Descripción de caso de uso "actualizar autor".....	3
Tabla 5: Descripción de caso de uso "eliminar autor".....	4
Tabla 6: Descripción de caso de uso "consultar lista de proyectos".....	4
Tabla 7: Descripción de caso de uso "consultar proyecto".....	5
Tabla 8: Descripción de caso de uso "añadir proyecto".....	5
Tabla 9: Descripción de caso de uso "actualizar proyecto".....	6
Tabla 10: Descripción de caso de uso "eliminar proyecto".....	7
Tabla 11: Descripción de caso de uso "consultar lista de revistas".....	7
Tabla 12: Descripción de caso de uso "consultar revista".....	8
Tabla 13: Descripción de caso de uso "añadir revista".....	8
Tabla 14: Descripción de caso de uso "actualizar revista".....	9
Tabla 15: Descripción de caso de uso "eliminar revista".....	10



<b>NOMBRE</b>	Gestión de autores – Consultar listado autores
<b>ACTORES</b>	Investigador
<b>OBJETIVO</b>	El usuario puede consultar la lista de autores registrados en el sistema.
<b>PRECONDICIÓN</b>	El usuario debe estar identificado en el sistema.
<b>FLUJO PRINCIPAL</b>	
ACTOR	SISTEMA
1. Usuario solicita acceso a la sección “autores”.	
	2. El sistema busca la lista de autores en la base de datos.
	3. El sistema busca los datos del investigador asociado al autor (si existe).
	4. El sistema devuelve una tabla con la lista de todos los autores además del nombre y apellidos del investigador que tiene asociado (si procede).

Tabla 1: Descripción de caso de uso "consultar lista de autores"

<b>NOMBRE</b>	Gestión de autores – Consultar autor
<b>ACTORES</b>	Investigador
<b>OBJETIVO</b>	El usuario puede consultar los datos de un autor del sistema así como los datos de investigador que tenga asociados (si procede).
<b>PRECONDICIÓN</b>	El usuario debe estar identificado en el sistema. Deben existir autores almacenados en el sistema.
<b>FLUJO PRINCIPAL</b>	
ACTOR	SISTEMA
1. El usuario pulsa en el botón “Ver” de un autor de la tabla.	
	2. El sistema busca los datos del autor en la base de datos.
	3. El sistema busca los datos de investigador del autor.
	4. El sistema muestra una pantalla con los datos del autor, y en caso de existir, sus datos de investigador.

Tabla 2: Descripción de caso de uso "consultar autor"

<b>NOMBRE</b>	Gestión de autores – Añadir autor
<b>ACTORES</b>	Administrador
<b>OBJETIVO</b>	El usuario añade un nuevo autor en el sistema.
<b>PRECONDICIÓN</b>	El usuario debe estar identificado en el sistema.
<b>FLUJO PRINCIPAL</b>	
ACTOR	SISTEMA
1. Usuario pulsa en el botón “añadir autor”.	
	2. El sistema devuelve un formulario con los datos a introducir.
3. El usuario introduce los datos del autor.	
	4. Se validan los datos del autor introducido. En caso de datos incorrectos <>datos incorrectos<>. En caso de seleccionarse un investigador en el campo correspondiente <>mostrar investigador<>.
	5. Se guarda el autor en la base de datos.
	6. El usuario es redirigido a la lista de autor mostrando un mensaje de éxito.
<b>FLUJO ALTERNATIVO: MOSTRAR INVESTIGADOR</b>	
	5. Se buscan los datos del investigador seleccionado en el sistema.
	6. Se muestran los datos en la pantalla de detalle.
<b>FLUJO ALTERNATIVO: DATOS INCORRECTOS</b>	
	5. Se muestran mensajes de error debajo de los campos mal introducidos explicando el problema.

Tabla 3: Descripción de caso de uso "añadir autor"

<b>NOMBRE</b>	Gestión de autores – Actualizar autor
<b>ACTORES</b>	Investigador
<b>OBJETIVO</b>	El usuario actualiza el autor en el sistema.
<b>PRECONDICIÓN</b>	<p>El usuario debe estar identificado en el sistema.</p> <p>Deben existir autores en la base de datos.</p> <p>El usuario puede ser:</p> <p>Administrador.</p> <p>Investigador, estando el autor a actualizar asociado a su perfil de investigador.</p>
<b>FLUJO PRINCIPAL</b>	
ACTOR	SISTEMA
1. Usuario pulsa, desde la lista de autores, sobre “Ver” en la fila del autor que quiere actualizar.	
	2. El sistema busca el autor en el sistema.
	3. En caso de que el autor tenga un investigador asociado, lo busca en la base de datos.
	4. El sistema devuelve un formulario con los datos de autor e investigador cubiertos.
5. El usuario edita los datos del autor.	
	6. Se validan los datos del autor introducido. En caso de datos incorrectos <<datos incorrectos>>. En caso de seleccionarse un investigador en el campo correspondiente <<mostrar investigador>>
	7. Se guarda el autor en la base de datos.
	8. El usuario es redirigido a la lista de autores mostrando un mensaje de éxito.
<b>FLUJO ALTERNATIVO: MOSTRAR INVESTIGADOR</b>	
	7. Se buscan los datos del investigador seleccionado en el sistema.
	8. Se muestran los datos en la pantalla de detalle.
<b>FLUJO ALTERNATIVO: DATOS INCORRECTOS</b>	
	7. Se muestran mensajes de error debajo de los campos mal introducidos explicando el problema.

Tabla 4: Descripción de caso de uso "actualizar autor"

<b>NOMBRE</b>	Gestión de autores – Eliminar autor
<b>ACTORES</b>	Administrador
<b>OBJETIVO</b>	El autor seleccionado se elimina del sistema.
<b>PRECONDICIÓN</b>	<p>Debe haber autores en el sistema.</p> <ul style="list-style-type: none"> <li>• El usuario debe estar identificado en el sistema.</li> <li>• El usuario puede ser: <ul style="list-style-type: none"> <li>◦ <u>Administrador</u>: En este caso puede eliminar a cualquier autor.</li> <li>◦ <u>Investigador</u>: En este caso solo podrá eliminar el autor correspondiente a su perfil de investigador.</li> </ul> </li> </ul>
<b>FLUJO PRINCIPAL</b>	
ACTOR	SISTEMA
1. En la lista de autores, el usuario pulsa en el botón “Ver” sobre una de las filas.	
	2. El sistema devuelve un formulario con los datos del autor llenados.
	3. En caso de tener un investigador asociado, se muestran sus datos.
4. El usuario pulsa en el botón “Eliminar autor”.	
	5. El autor se elimina de la base de datos.
	6. El sistema redirige al usuario al listado de autores mostrando un mensaje de éxito.

Tabla 5: Descripción de caso de uso "eliminar autor"

<b>NOMBRE</b>	Gestión de proyectos – Consultar listado proyectos
<b>ACTORES</b>	Investigador
<b>OBJETIVO</b>	El usuario puede consultar la lista de proyectos.
<b>PRECONDICIÓN</b>	El usuario debe estar identificado en el sistema.
<b>FLUJO PRINCIPAL</b>	
ACTOR	SISTEMA
1. Usuario solicita acceso a la sección “proyectos”.	
	2. El sistema busca la lista de proyectos en la base de datos.
	3. El sistema devuelve una tabla con la lista de todos los proyectos.

Tabla 6: Descripción de caso de uso "consultar lista de proyectos"

<b>NOMBRE</b>	Gestión de proyectos – Consultar proyecto
<b>ACTORES</b>	Investigador
<b>OBJETIVO</b>	El usuario puede consultar los datos de un proyecto almacenado en el sistema.
<b>PRECONDICIÓN</b>	El usuario debe estar identificado en el sistema. Deben existir proyectos almacenados en el sistema.
<b>FLUJO PRINCIPAL</b>	
ACTOR	SISTEMA
1. El usuario pulsa en el botón “Ver” de un proyecto de la tabla.	
	2. El sistema busca los datos del proyecto en la base de datos.
	3. El sistema busca la lista de autores relacionados con ese proyecto.
	4. El sistema muestra una pantalla con los datos del proyecto y autores asociados.

Tabla 7: Descripción de caso de uso "consultar proyecto"

<b>NOMBRE</b>	Gestión de proyectos – Añadir proyecto
<b>ACTORES</b>	Administrador
<b>OBJETIVO</b>	El usuario añade un nuevo proyecto en el sistema.
<b>PRECONDICIÓN</b>	El usuario debe estar identificado en el sistema.
<b>FLUJO PRINCIPAL</b>	
ACTOR	SISTEMA
1. Usuario pulsa en el botón “añadir proyecto”.	
	2. El sistema obtiene de la base de datos la lista de autores para que el usuario pueda asociarlos en caso de desecharlo.
	3. El sistema devuelve un formulario con los datos a introducir.
4. El usuario introduce los datos del proyecto así como los autores participantes.	
	5. Se validan los datos del proyecto introducido. En caso de datos incorrectos <<datos incorrectos>>.
	6. Se guarda el proyecto en la base de datos.
	7. El usuario es redirigido a la lista de proyectos mostrando un mensaje de éxito.
<b>FLUJO ALTERNATIVO: DATOS INCORRECTOS</b>	
	6. Se muestran mensajes de error debajo de los campos mal introducidos explicando el problema.

Tabla 8: Descripción de caso de uso "añadir proyecto"

<b>NOMBRE</b>	Gestión de proyectos – Actualizar proyecto
<b>ACTORES</b>	Investigador
<b>OBJETIVO</b>	El usuario actualiza el proyecto en el sistema.
<b>PRECONDICIÓN</b>	<ul style="list-style-type: none"> <li>• El usuario debe estar identificado en el sistema.</li> <li>• Deben existir proyectos en la base de datos.</li> <li>• El usuario debe tener el rol de: <ul style="list-style-type: none"> <li>◦ Administrador.</li> <li>◦ Investigador, y ser participante del proyecto a actualizar.</li> </ul> </li> </ul>
<b>FLUJO PRINCIPAL</b>	
ACTOR	SISTEMA
1. Usuario pulsa, desde la lista de proyectos, sobre “Ver” en la fila del proyecto que quiere actualizar.	
	2. El sistema busca el proyecto y sus autores participantes en la base de datos.
	3. El sistema busca la lista total de autores en la base de datos.
	4. El sistema devuelve un formulario con los datos del proyecto cubiertos además de la tabla de autores participantes en este.
5. El usuario edita los datos del proyecto.	
	6. Se validan los datos de proyecto introducidos. En caso de datos incorrectos <<datos incorrectos>>.
	7. Se guarda el proyecto en la base de datos.
	8. El usuario es redirigido a la lista de proyectos mostrando un mensaje de éxito.
<b>FLUJO ALTERNATIVO: DATOS INCORRECTOS</b>	
	7. Se muestran mensajes de error debajo de los campos mal introducidos explicando el problema.

Tabla 9: Descripción de caso de uso "actualizar proyecto"

<b>NOMBRE</b>	Gestión de proyecto – Eliminar proyecto	
<b>ACTORES</b>	Investigador	
<b>OBJETIVO</b>	El proyecto seleccionado se elimina del sistema.	
<b>PRECONDICIÓN</b>	<p>Debe haber proyectos en el sistema y el usuario debe estar identificado. El usuario puede ser:</p> <ul style="list-style-type: none"> <li>• <u>Administrador</u>: En este caso puede eliminar cualquier proyecto.</li> <li>• <u>Investigador</u>: En este caso solo podrá eliminar los proyectos en los que participe.</li> </ul>	
<b>FLUJO PRINCIPAL</b>		
ACTOR	SISTEMA	
1. En la lista de proyectos, el usuario pulsa en el botón “Ver” sobre una de las filas.		
	2. El sistema obtiene los datos del proyecto junto con los autores participantes y la lista total de autores.	
	3. El sistema devuelve un formulario con los datos del proyecto llenados y la lista de autores participantes.	
4. El usuario pulsa en el botón “Eliminar proyecto”.		
	5. El proyecto se elimina de la base de datos.	
	6. El sistema redirige al usuario al listado de proyectos mostrando un mensaje de éxito.	

Tabla 10: Descripción de caso de uso "eliminar proyecto"

<b>NOMBRE</b>	Gestión de revistas – Consultar listado revistas	
<b>ACTORES</b>	Investigador	
<b>OBJETIVO</b>	El usuario puede consultar la lista de revistas.	
<b>PRECONDICIÓN</b>	El usuario debe estar identificado en el sistema.	
<b>FLUJO PRINCIPAL</b>		
ACTOR	SISTEMA	
1. Usuario solicita acceso a la sección “revistas”.		
	2. El sistema busca la lista de revistas en la base de datos.	
	3. El sistema devuelve una tabla con la lista de todos los revistas.	

Tabla 11: Descripción de caso de uso "consultar lista de revistas"

<b>NOMBRE</b>	Gestión de revistas – Consultar revista
<b>ACTORES</b>	Investigador
<b>OBJETIVO</b>	El usuario puede consultar los datos de una publicación en revista almacenada en el sistema.
<b>PRECONDICIÓN</b>	El usuario debe estar identificado en el sistema. Deben existir revistas almacenados en el sistema.
<b>FLUJO PRINCIPAL</b>	
ACTOR	SISTEMA
4. El usuario pulsa en el botón “Ver” de una revista de la tabla.	
	5. El sistema busca los datos de la publicación en revista en la base de datos.
	6. El sistema busca la lista de autores relacionados con esa publicación en revista.
	7. El sistema muestra una pantalla con los datos de la revista y autores asociados.

Tabla 12: Descripción de caso de uso "consultar revista"

<b>NOMBRE</b>	Gestión de revistas – Añadir revista
<b>ACTORES</b>	Administrador
<b>OBJETIVO</b>	El usuario añade una nueva revista en el sistema.
<b>PRECONDICIÓN</b>	El usuario debe estar identificado en el sistema.
<b>FLUJO PRINCIPAL</b>	
ACTOR	SISTEMA
1. Usuario pulsa en el botón “añadir”.	
	2. El sistema obtiene de la base de datos la lista de publicaciones en revista para que el usuario pueda asociarlos en caso de desecharlo.
	3. El sistema devuelve un formulario con los datos a introducir.
4. El usuario introduce los datos de la publicación en revista así como los autores participantes.	
	5. Se validan los datos de la publicación introducida. En caso de datos incorrectos <<datos incorrectos>>.
	6. Se guarda la publicación en revista en la base de datos.
	7. El usuario es redirigido a la lista de publicaciones en revista mostrando un mensaje de éxito.
<b>FLUJO ALTERNATIVO: DATOS INCORRECTOS</b>	
	6. Se muestran mensajes de error debajo de los campos mal introducidos explicando el problema.

Tabla 13: Descripción de caso de uso "añadir revista"

<b>NOMBRE</b>	Gestión de revistas – Actualizar revista	
<b>ACTORES</b>	Investigador	
<b>OBJETIVO</b>	El usuario actualiza la publicación en revista en el sistema.	
<b>PRECONDICIÓN</b>	<ul style="list-style-type: none"> <li>• El usuario debe estar identificado en el sistema.</li> <li>• Deben existir publicaciones en revista en la base de datos.</li> <li>• El usuario debe tener el rol de: <ul style="list-style-type: none"> <li>◦ Administrador.</li> <li>◦ Investigador, y ser participante de la publicación a actualizar.</li> </ul> </li> </ul>	
<b>FLUJO PRINCIPAL</b>		
ACTOR	SISTEMA	
1. Usuario pulsa, desde la lista de publicaciones en revista, sobre “Ver” en la fila del proyecto que quiere actualizar.		
	2. El sistema busca la publicación en revista y sus autores participantes en la base de datos.	
	3. El sistema busca la lista total de autores en la base de datos.	
	4. El sistema devuelve un formulario con los datos de la publicación en revista cubiertos además de la tabla de autores participantes en este.	
5. El usuario edita los datos de la publicación.		
	6. Se validan los datos de la publicación introducida. En caso de datos incorrectos <<datos incorrectos>>.	
	7. Se guarda la publicación en la base de datos.	
	8. El usuario es redirigido a la lista de publicaciones en revista mostrando un mensaje de éxito.	
<b>FLUJO ALTERNATIVO: DATOS INCORRECTOS</b>		
	7. Se muestran mensajes de error debajo de los campos mal introducidos explicando el problema.	

Tabla 14: Descripción de caso de uso "actualizar revista"

<b>NOMBRE</b>	Gestión de revista – Eliminar revista
<b>TIPO</b>	Principal
<b>ACTORES</b>	Investigador
<b>OBJETIVO</b>	El publicación seleccionada se elimina del sistema.
<b>PRECONDICIÓN</b>	<ul style="list-style-type: none"> <li>• Debe haber publicaciones en revista en el sistema.</li> <li>• El usuario debe estar identificado en el sistema. El usuario puede ser: <ul style="list-style-type: none"> <li>◦ Administrador: En este caso puede eliminar cualquier publicación.</li> <li>◦ Investigador: En este caso solo podrá eliminar las publicaciones en las que participe.</li> </ul> </li> </ul>
<b>FLUJO PRINCIPAL</b>	
ACTOR	SISTEMA
1. En la lista de publicaciones en revista, el usuario pulsa en el botón “Ver” sobre una de las filas.	
	2. El sistema obtiene los datos de la publicación junto con los autores participantes y la lista total de autores.
	3. El sistema devuelve un formulario con los datos de la publicación llenos y la lista de autores participantes.
4. El usuario pulsa en el botón “Eliminar”.	
	5. La publicación se elimina de la base de datos.
	6. El sistema redirige al usuario al listado de publicaciones en revista mostrando un mensaje de éxito.

Tabla 15: Descripción de caso de uso "eliminar revista"

# **Anexo III: Diseño estático en el servidor**



## **Índice**

1. Diagrama de clases de gestión de autores.....	1
2. Diagrama de clases de gestión de proyectos.....	2
3. Diagrama de clases de gestión de publicaciones en revista.....	3



## **Índice de Figuras**

Figura 1: Diagrama de clases de gestión de autores.....	1
Figura 2: Diagrama de clases de gestión de proyectos.....	2
Figura 3: Diagrama de clases de gestión de publicaciones en revista.....	3



# 1. Diagrama de clases de gestión de autores

En la Figura 1 se presenta el diagrama de clases relacionado con la gestión de autores. A continuación se explican las responsabilidades de las distintas clases:

- AuthorController: clase encargada de recibir las peticiones relacionadas con los autores y generar una respuesta.
- AuthorRepository: clase encargada de acceder a los datos almacenados relacionados con los autores.
- Author: representa a un autor en el sistema.
- AuthorVO: representa a un autor en la vista. Se encarga de realizar las transformaciones entre los autores de la vista y las entidades autor del sistema.
- AuthorProject: representa la relación entre un proyecto y un autor.
- AuthorCongress: representa la relación entre un congreso y un autor.
- AuthorJournal: representa la relación entre una revista y un autor.
- AuthorTable: representa a la tabla autores de la base de datos. Esta clase es utilizada por el *framework* para convertir los autores del sistema en su entidad correspondiente de base de datos y viceversa.
- AuthorProjectTable: representa a la tabla que relaciona autores y proyectos en base de datos. Esta clase es utilizada por el *framework* para convertir la clase en el sistema en su entidad correspondiente de base de datos y viceversa.
- AuthorCongressTable: representa la tabla que relaciona congresos y autores en base de datos. Esta clase es utilizada por el *framework* para la conversión entre la entidad de base de datos y la clase correspondiente.
- AuthorJournalTable: representa la tabla que relaciona revistas y autores en base de datos. Esta clase es utilizada por el *framework* para la conversión entre la entidad de base de datos y la clase correspondiente.

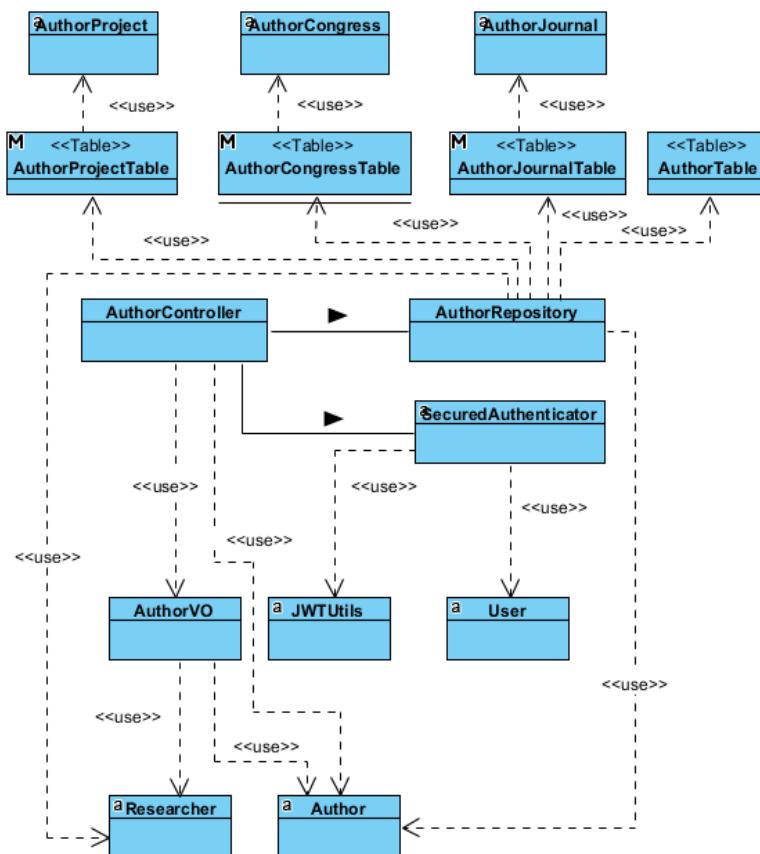


Figura 1: Diagrama de clases de gestión de autores

## 2. Diagrama de clases de gestión de proyectos

En la Figura 2 se presenta el diagrama de casos de uso de gestión de proyectos. A continuación se explica la responsabilidad de cada una de las clases implicadas:

- ProjectController: clase encargada de recibir las peticiones relacionadas con los proyectos y generar una respuesta.
- ProjectRepository: clase encargada de acceder a los datos almacenados relacionados con los proyectos.
- AuthorProjectRepository: clase encargada de realizar operaciones de base de datos que se relacionen con ambas entidades, proyectos y autores.
- Project: representa a un proyecto en el sistema.
- ProjectVO: representa a un proyecto en la vista. Se encarga de realizar las transformaciones entre los proyectos de la vista y las entidades proyecto del sistema.
- Role: representa a un rol de un autor en un proyecto.
- AuthorOfProjectVO: Representa un autor que además tiene atributos relacionados con el proyecto que no era posible introducir en otro sitio. En el caso de esta entidad se añade el atributo que representa el rol que tiene en el autor en el proyecto.

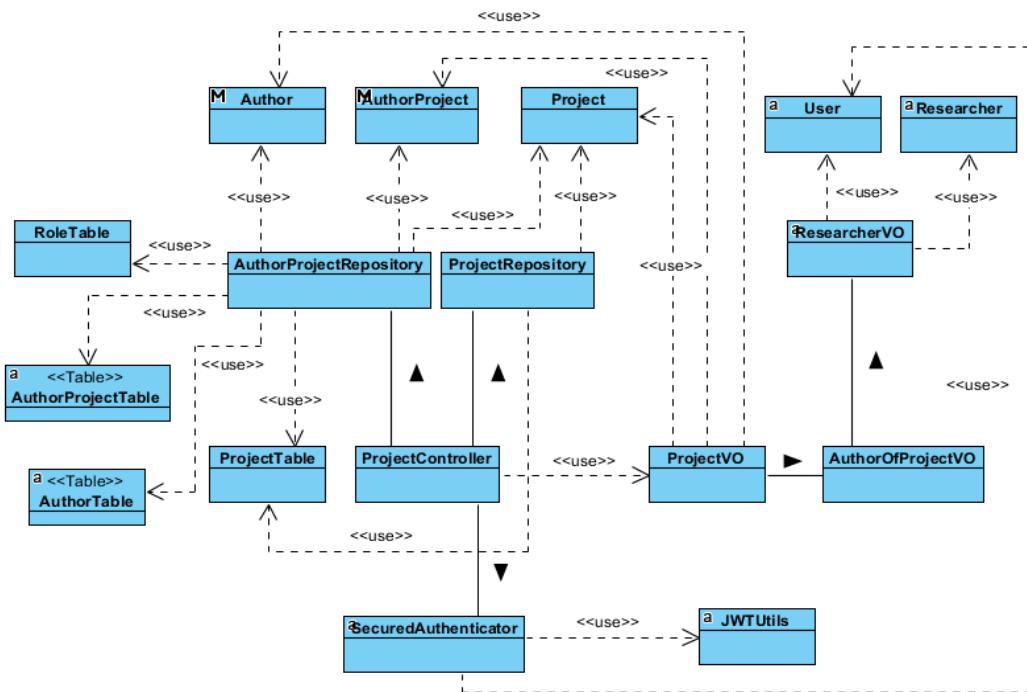


Figura 2: Diagrama de clases de gestión de proyectos

### 3. Diagrama de clases de gestión de publicaciones en revista

En la Figura 3 se presenta el diagrama de clases relacionado con las publicaciones en revista. A continuación se explican las responsabilidades de cada clase:

- JournalController: clase encargada de recibir las peticiones relacionadas con las publicaciones en revista y generar una respuesta.
- JournalRepository: clase encargada de acceder a los datos almacenados relacionados con publicaciones en revista.
- AuthorJournalRepository: clase encargada de realizar operaciones de base de datos que se relacionen con ambas entidades, congresos y revistas.
- Journal: representa a una publicación en revista en el sistema.
- JournalVO: representa a una publicación en revista en la vista. Se encarga de realizar las transformaciones entre las revistas de la vista y las entidades revista del sistema.
- AuthorOfJournalVO: Representa un autor que además tiene atributos relacionados con una publicación en revista.

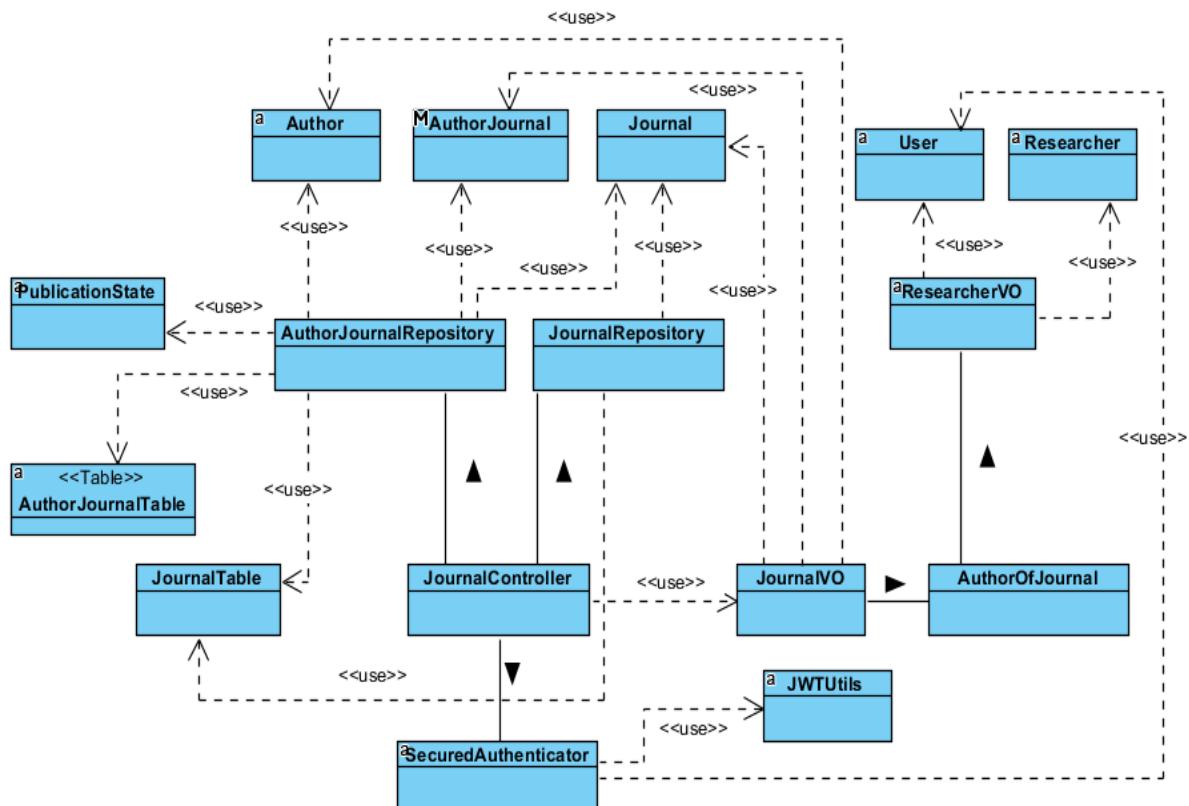


Figura 3: Diagrama de clases de gestión de publicaciones en revista



# **Anexo IV: Detalle de diseño estático en el cliente**



## **Índice**

1. Paquete de constantes.....	1
2. Paquete de acciones.....	1
3. Paquete de reductores.....	1
4. Paquete de utilidad.....	1



## **Índice de Figuras**

Figura 1: Diagrama de clases mixto del paquete de constantes.....	1
Figura 2: Diagrama de clases mixto del paquete de acciones.....	1
Figura 3: Diagrama de clases mixto del paquete de reductores.....	1



## 1. Paquete de constantes

En la Figura 1 se pueden ver los ficheros que contienen las constantes usadas en la aplicación:

- actionTypes.js: contiene las constantes utilizadas para enviar acciones desde el paquete de acciones.
- calendar-culture.js: contiene los datos de información de idioma de los calendarios.
- calendar-format.js: contiene las constantes que se mostrarán en la pantalla de calendarios.

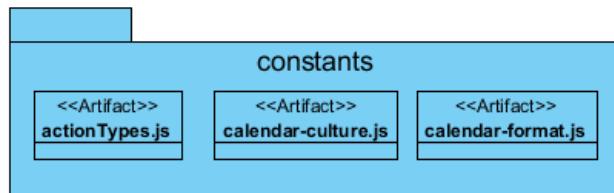


Figura 1: Diagrama de clases mixto del paquete de constantes

## 2. Paquete de acciones

En la Figura 2 se presenta el diagrama de clases mixto de acciones. Cada acción recibe el nombre de la clase de la cual es responsable de dichas acciones. Las acciones son recogidas por el reductor correspondiente.

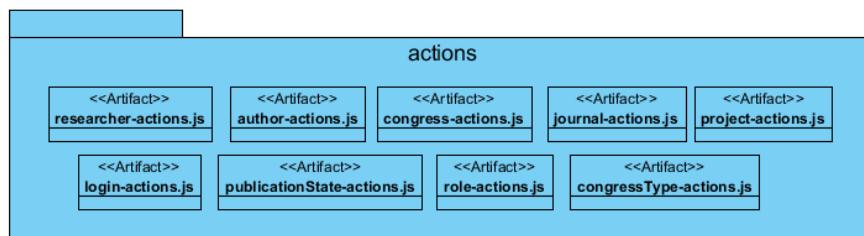


Figura 2: clases del paquete de acciones

## 3. Paquete de reductores

En la Figura 3 se presentan los ficheros en cargados de recoger las acciones enviadas en la aplicación. Cada fichero tiene el nombre de la entidad de la cual es responsable de actualizar su estado.

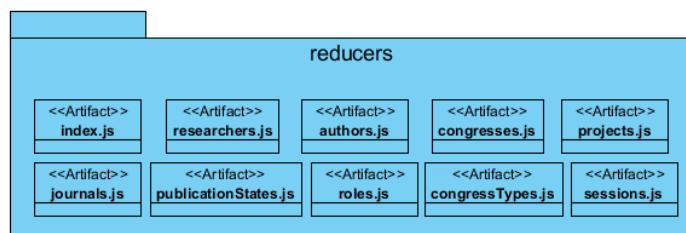


Figura 3: clases del paquete de reductores

## 4. Paquete de utilidad

En la Figura 4 se muestra el diagrama de clases mixto del paquete de utilidad. A continuación se explica la responsabilidad de cada entidad:

- DateUtils: se encarga de procesar las fechas.
- NotificationUtils.js: abstracción encargada de enviar notificaciones. Se utiliza desde la pantalla de calendarios.
- SessionUtils: abstracción encargada de gestionar la información de sesiones de usuario interactuando con el *localStorage* del navegador.

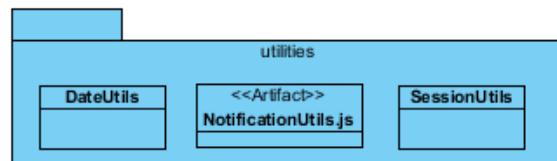


Figura 4: clases del paquete de utilidad

# **Anexo V: Diagramas de secuencia detallados**



# Índice

1. Diagramas de secuencia secundarios.....	1
1.1. Diagrama de obtención de tipos de congreso.....	1
1.2. Diagrama de secuencia de obtención de estados de publicación.....	1
1.3. Diagrama de secuencia de obtención de roles.....	2
1.4. Diagrama de obtención de investigadores.....	2
1.5. Diagrama de secuencia de obtención de autores.....	3
2. Diagramas de secuencia principales.....	3
2.1. Diagrama de secuencia de identificarse.....	3
2.2. Diagrama de secuencia de salir.....	4
2.3. Diagrama de secuencia de añadir investigador.....	4
2.4. Diagrama de secuencia de actualizar investigador.....	5
2.5. Diagrama de secuencia de consultar lista de autores.....	6
2.6. Diagrama de secuencia de añadir autor.....	6
2.7. Diagrama de secuencia de actualizar autor.....	7
2.8. Diagrama de secuencia de eliminar autor.....	8
2.9. Diagrama de secuencia de consultar lista de revistas.....	8
2.10. Diagrama de secuencia de añadir revista.....	9
2.11. Diagrama de secuencia de actualizar revista.....	10
2.12. Diagrama de secuencia de eliminar revista.....	11
2.13. Diagrama de secuencia de consultar lista de proyectos.....	11
2.14. Diagrama de secuencia de añadir proyecto.....	12
2.15. Diagrama de secuencia de actualizar proyecto.....	13
2.16. Diagrama de secuencia de eliminar proyecto.....	14
2.17. Diagrama de secuencia de consultar lista de congresos.....	14
2.18. Diagrama de secuencia de añadir congreso.....	15



## Índice de Figuras

Figura 1: Diagrama de secuencia de obtención de tipos de congreso.....	1
Figura 2: Diagrama de secuencia de obtención de estados de publicación.....	1
Figura 3: Diagrama secuencia de obtención de roles.....	2
Figura 4: Diagrama de secuencia de obtención de investigadores.....	2
Figura 5: Diagrama de secuencia de obtención de autores.....	3
Figura 6: Diagrama de secuencia de identificarse.....	3
Figura 7: Diagrama de secuencia de salir.....	4
Figura 8: Diagrama de secuencia de añadir investigador.....	4
Figura 9: Diagrama de secuencia de actualizar investigador.....	5
Figura 10: Diagrama de secuencia de consultar lista de autores.....	6
Figura 11: Diagrama de secuencia de añadir autor.....	6
Figura 12: Diagrama de secuencia de actualizar autor.....	7
Figura 13: Diagrama de secuencia de eliminar autor.....	8
Figura 14: Diagrama de secuencia de consultar lista de revistas.....	8
Figura 15: Diagrama de secuencia de añadir revista.....	9
Figura 16: Diagrama de secuencia de actualizar revista.....	10
Figura 17: Diagrama de secuencia de eliminar revista.....	11
Figura 18: Diagrama de secuencia de consultar lista de proyectos.....	11
Figura 19: Diagrama de secuencia de añadir proyecto.....	12
Figura 20: Diagrama de secuencia de actualizar proyecto.....	13
Figura 21: Diagrama de secuencia de eliminar proyecto.....	14
Figura 22: Diagrama de secuencia de consultar lista de congresos.....	14
Figura 23: Diagrama de secuencia de añadir congreso.....	15



# 1. Diagramas de secuencia secundarios

## 1.1. Diagrama de obtención de tipos de congreso

En la aplicación es necesario mostrar un listado con los tipos de congreso para permitir que el propio usuario pueda seleccionarlos. Esto, por lo tanto, da lugar a la creación del diagrama de la Figura 1, el cual se utilizará posteriormente de apoyo para la parte de añadido y actualización de un congreso.

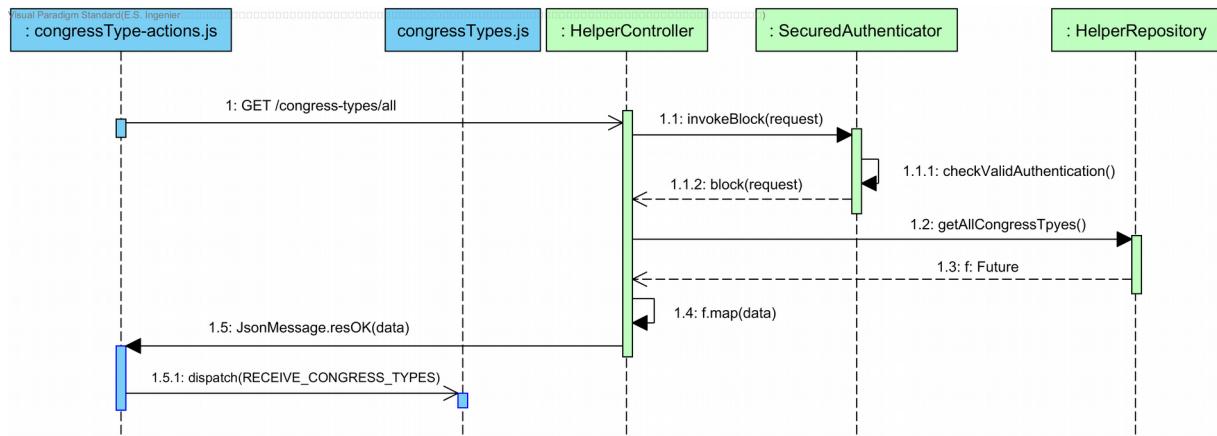


Figura 1: Diagrama de secuencia de obtención de tipos de congreso

## 1.2. Diagrama de secuencia de obtención de estados de publicación

En la Figura 2 se muestra el diagrama que permite obtener la lista de estados posibles para una publicación. Este diagrama se utiliza en las secciones de revistas y congresos.

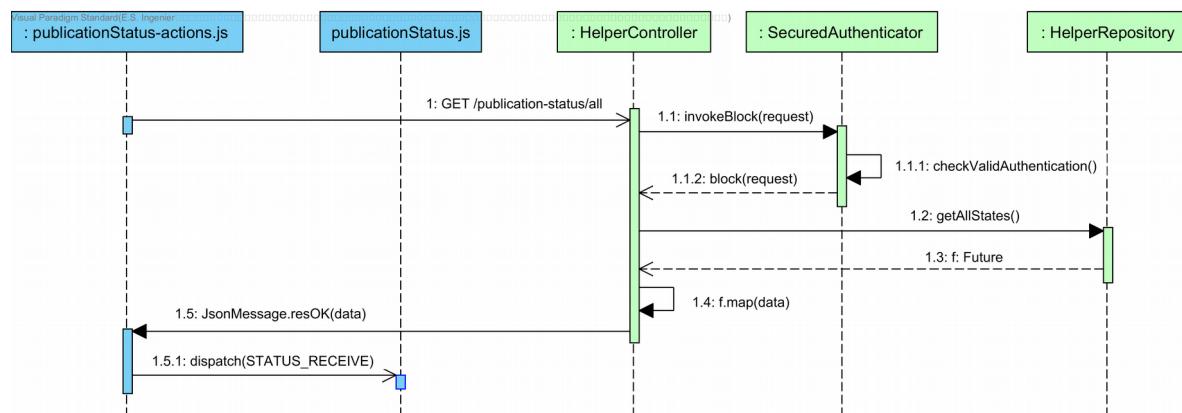


Figura 2: Diagrama de secuencia de obtención de estados de publicación

### 1.3. Diagrama de secuencia de obtención de roles

En la Figura 3 se presenta la obtención de los posibles roles que puede tener asociado un autor respecto a un proyecto.

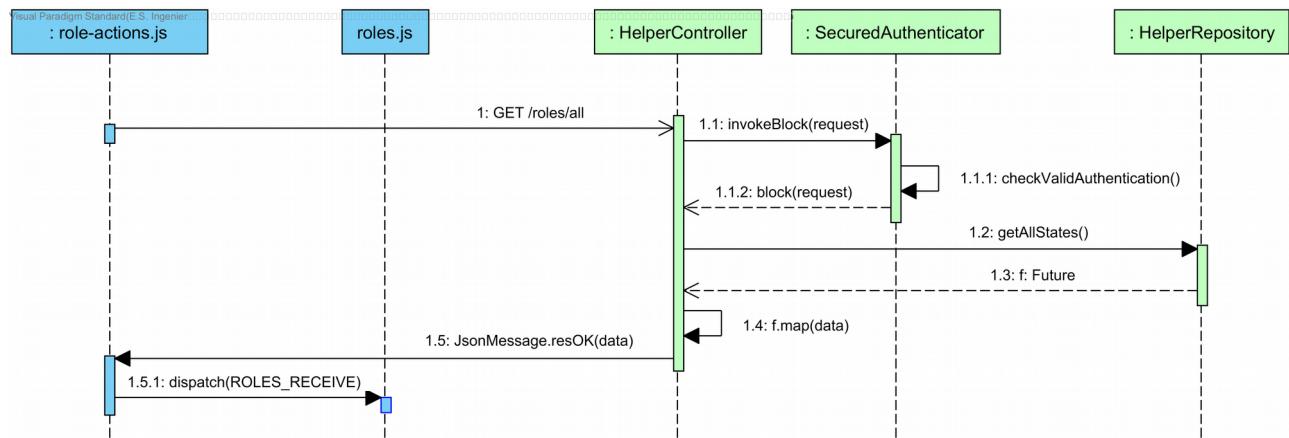


Figura 3: Diagrama secuencia de obtención de roles

### 1.4. Diagrama de obtención de investigadores

En la Figura 4 se puede ver el diagrama de secuencia que representa la manera en que se obtienen los investigadores del sistema.

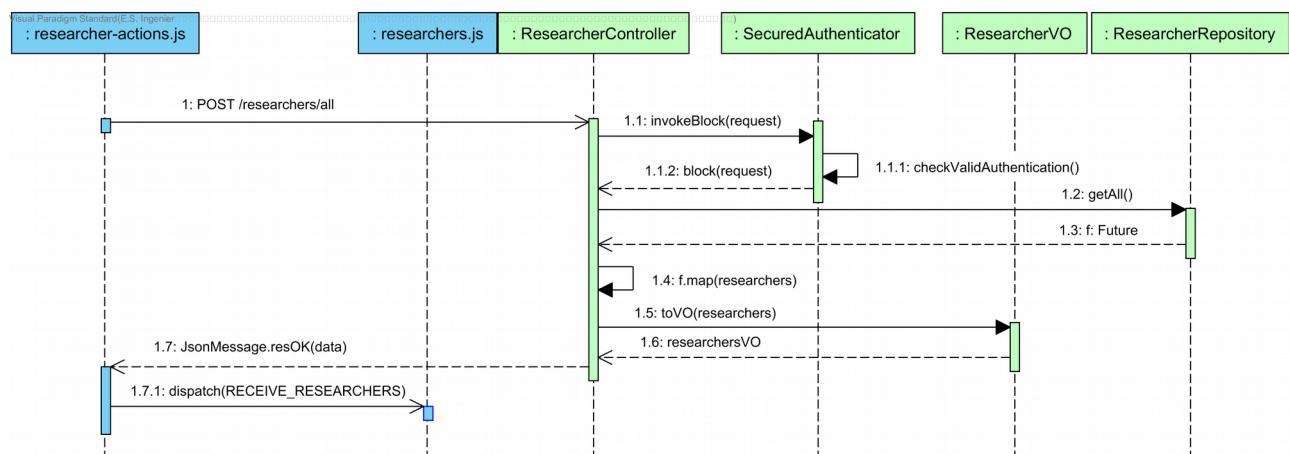
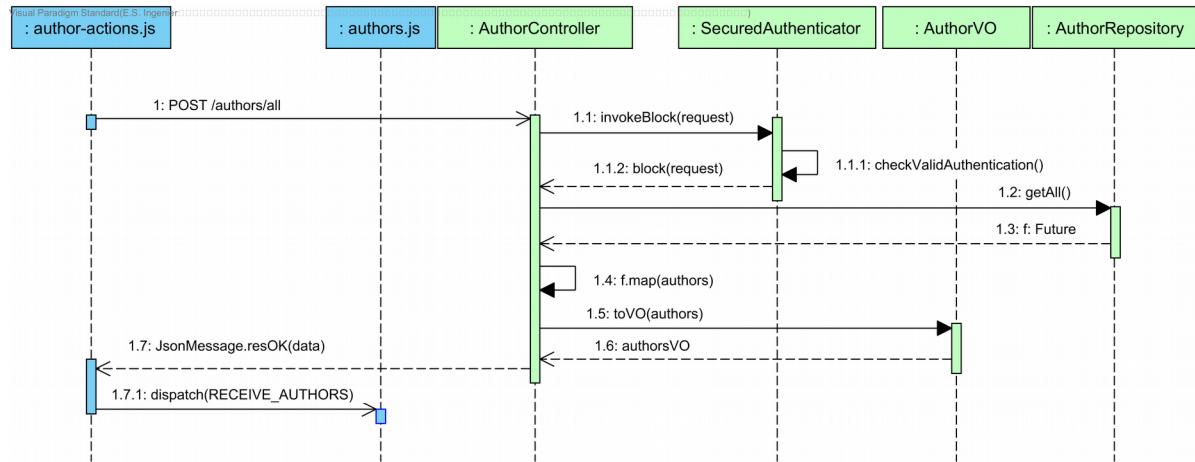


Figura 4: Diagrama de secuencia de obtención de investigadores

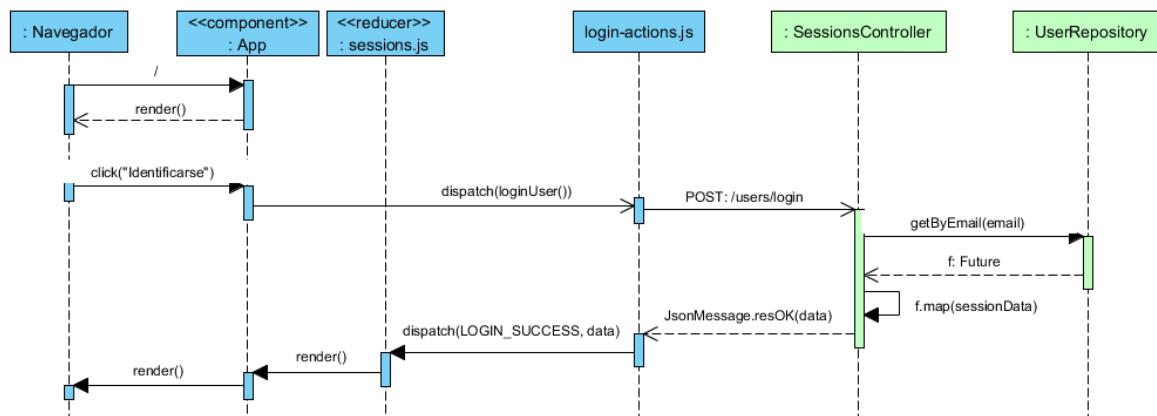
## 1.5. Diagrama de secuencia de obtención de autores

En la Figura 5 se muestra el diagrama de secuencia que representa la manera en que se consulta la lista de autores del sistema.



## 2. Diagramas de secuencia principales

### 2.1. Diagrama de secuencia de identificarse



## 2.2. Diagrama de secuencia de salir

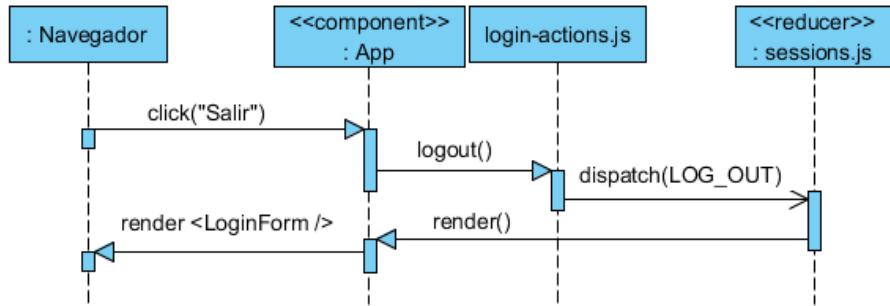


Figura 7: Diagrama de secuencia de salir

## 2.3. Diagrama de secuencia de añadir investigador

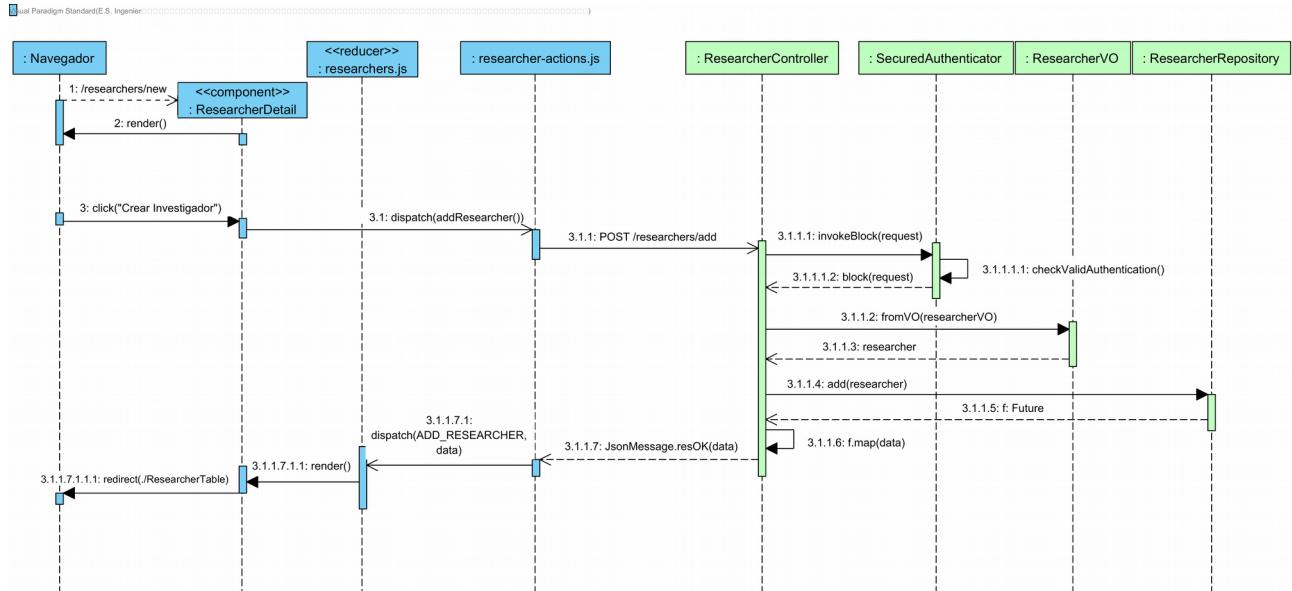


Figura 8: Diagrama de secuencia de añadir investigador

## 2.4. Diagrama de secuencia de actualizar investigador

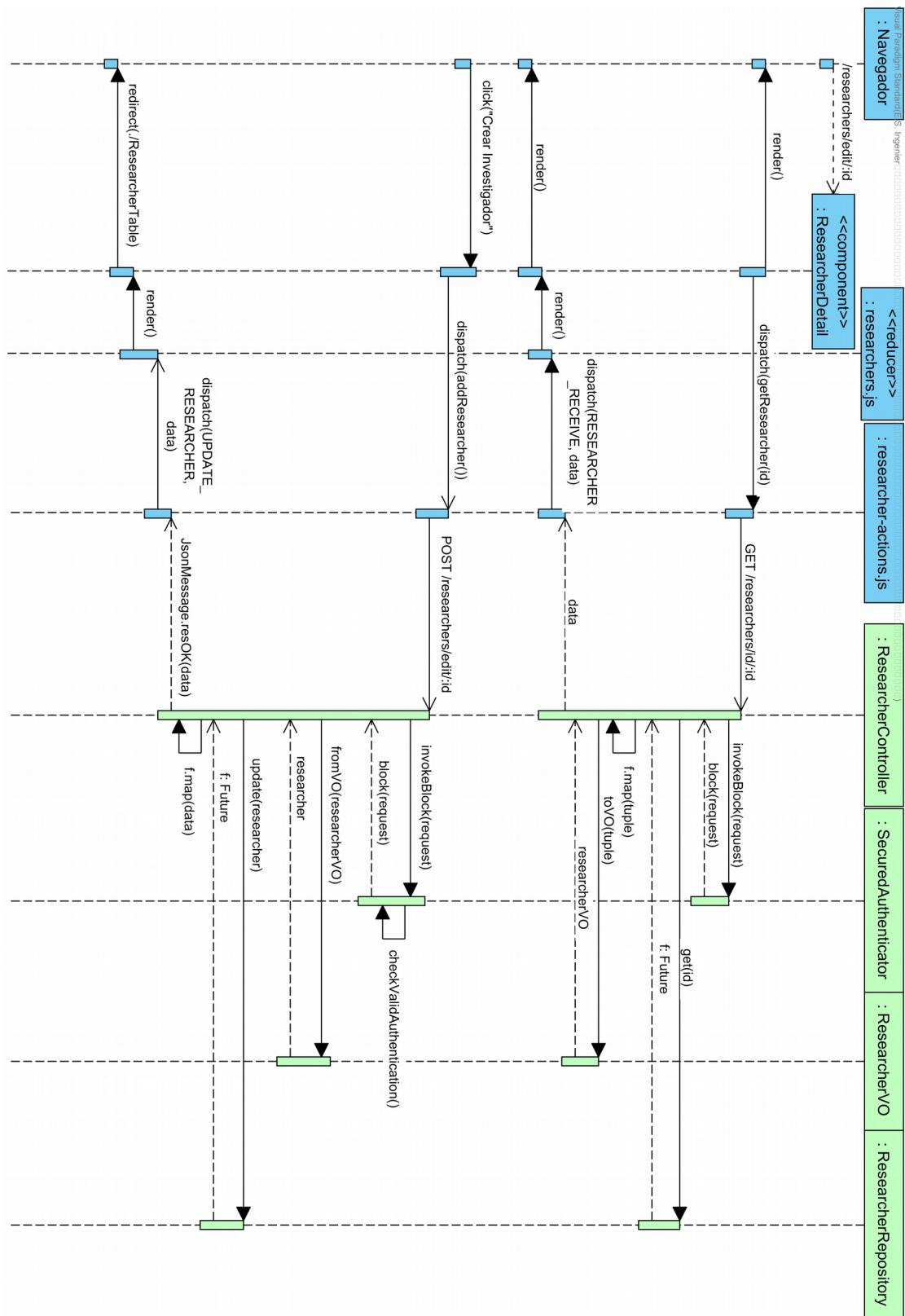


Figura 9: Diagrama de secuencia de actualizar investigador

## 2.5. Diagrama de secuencia de consultar lista de autores

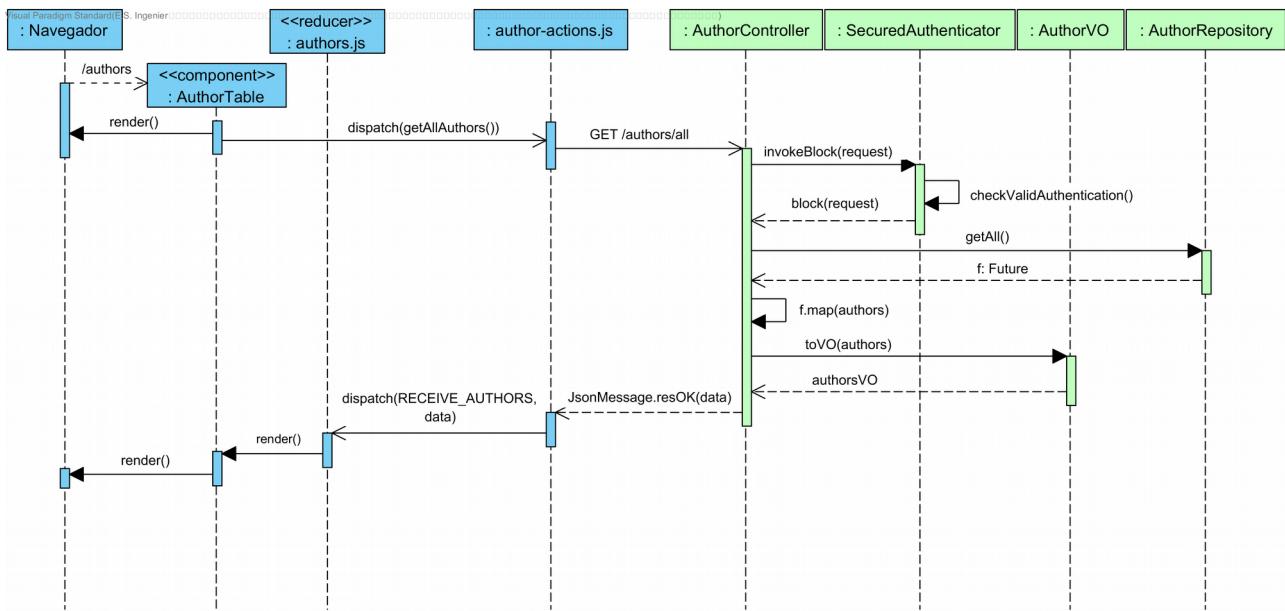


Figura 10: Diagrama de secuencia de consultar lista de autores

## 2.6. Diagrama de secuencia de añadir autor

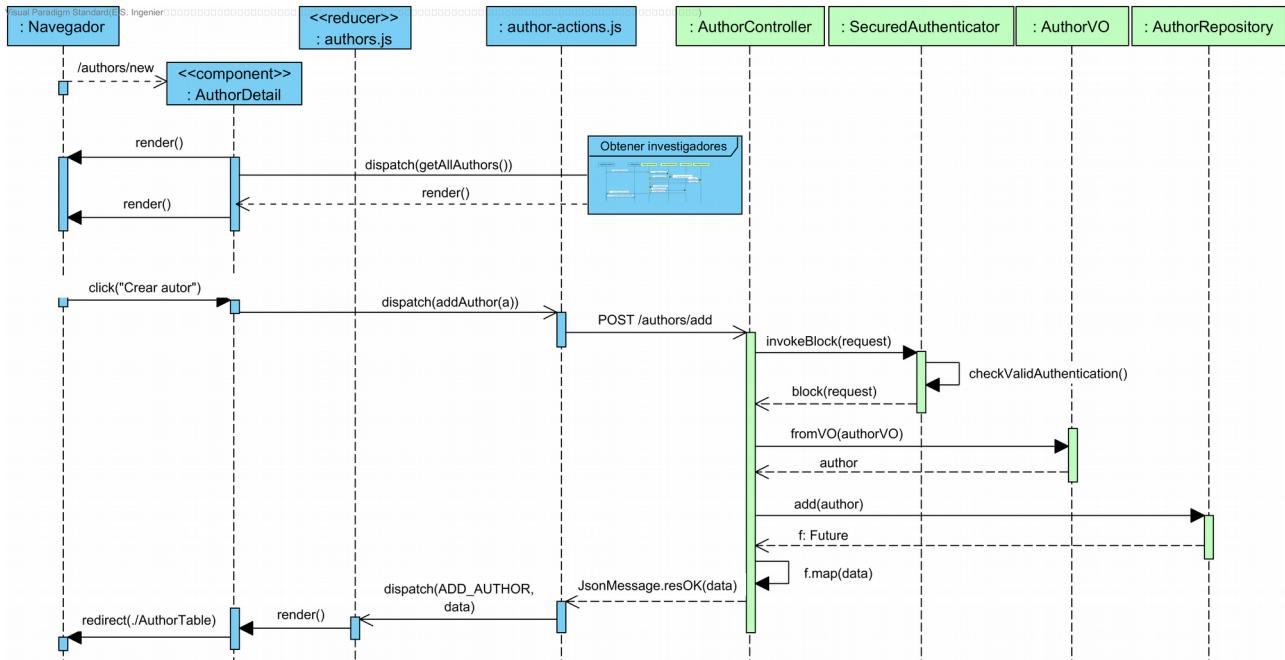
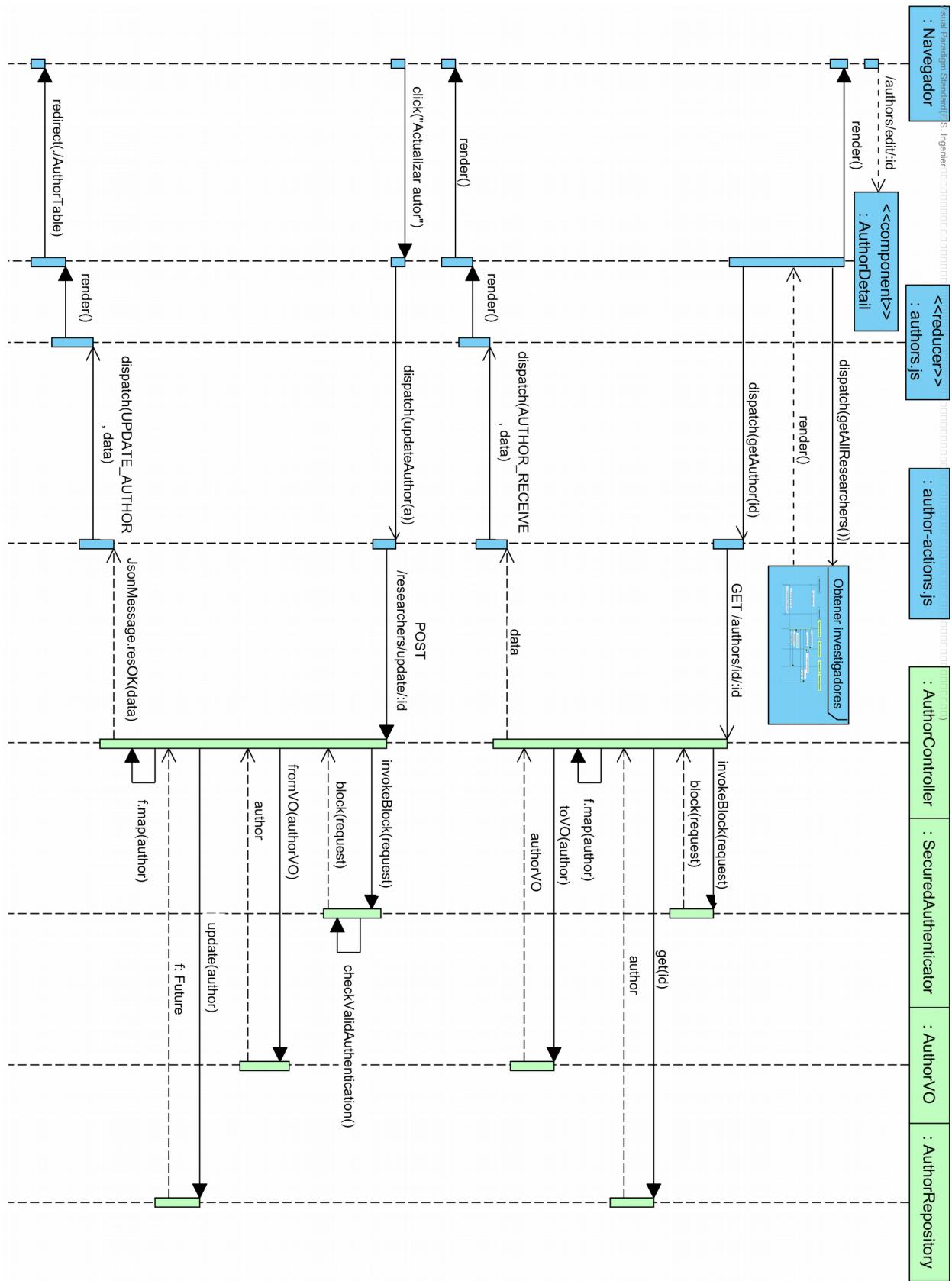


Figura 11: Diagrama de secuencia de añadir autor

## 2.7. Diagrama de secuencia de actualizar autor



*Figura 12: Diagrama de secuencia de actualizar autor*

## 2.8. Diagrama de secuencia de eliminar autor

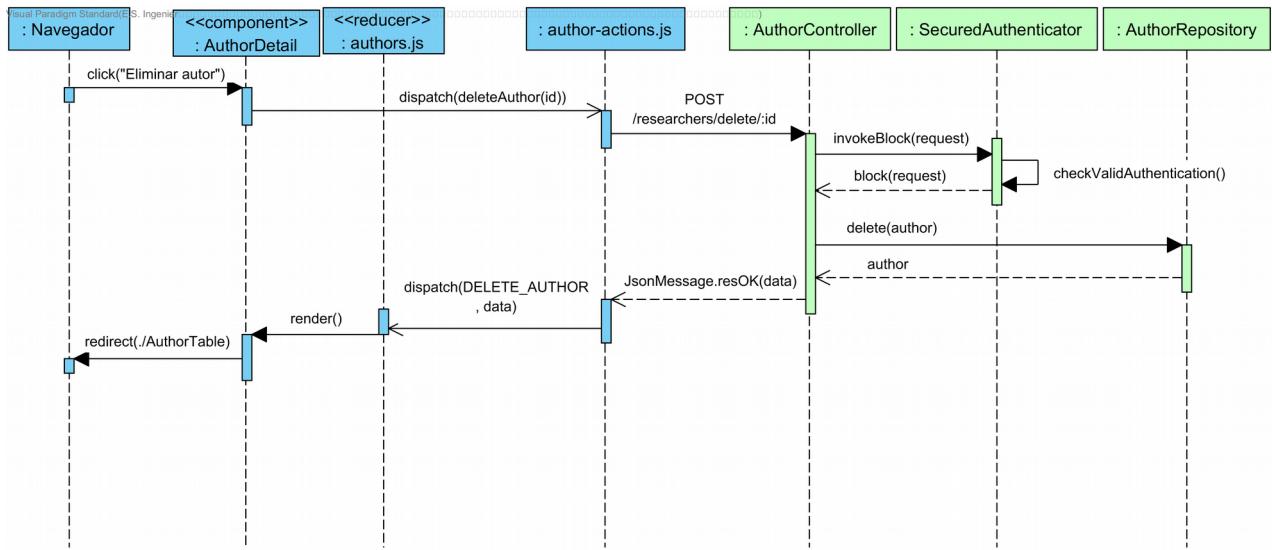


Figura 13: Diagrama de secuencia de eliminar autor

## 2.9. Diagrama de secuencia de consultar lista de revistas

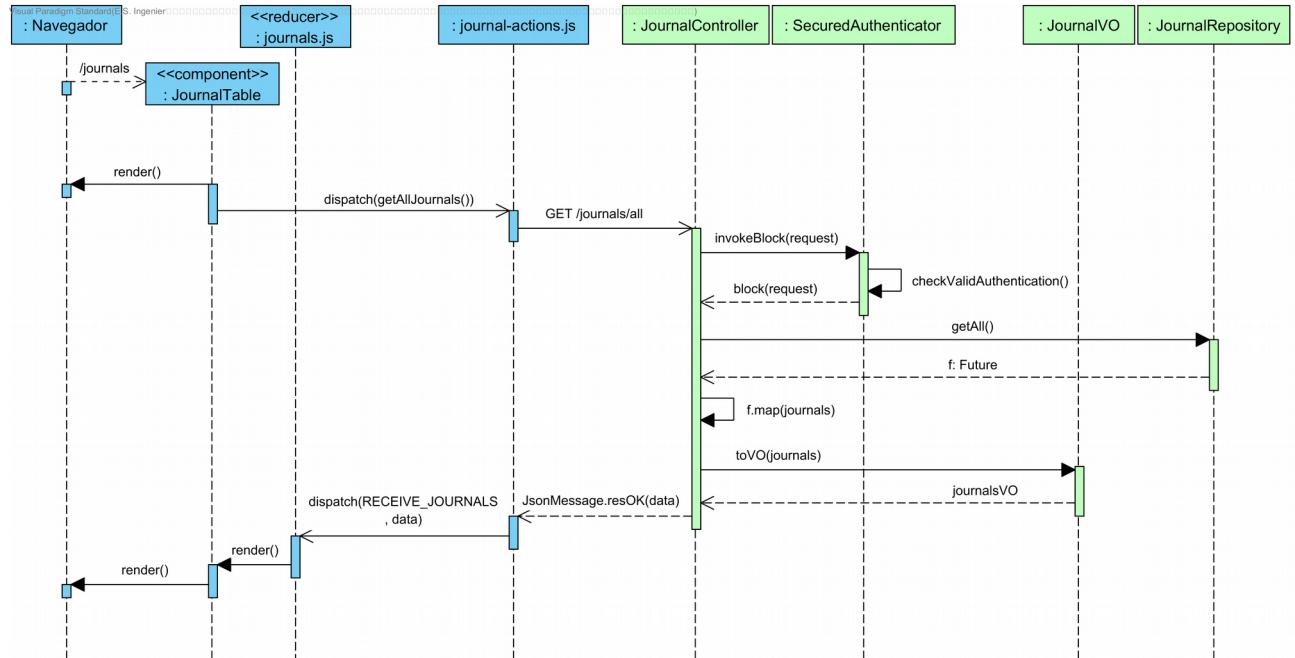


Figura 14: Diagrama de secuencia de consultar lista de revistas

## 2.10. Diagrama de secuencia de añadir revista

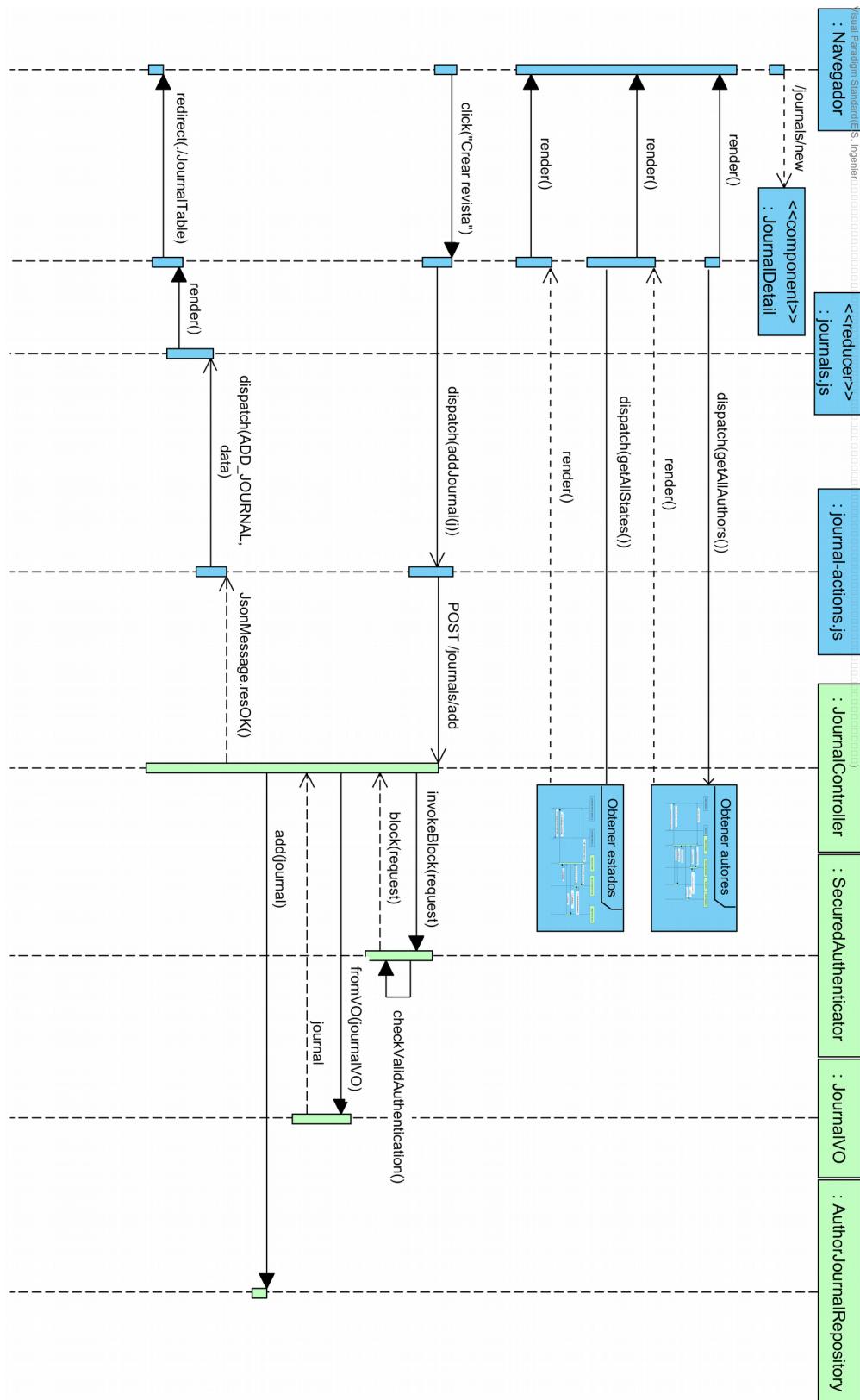
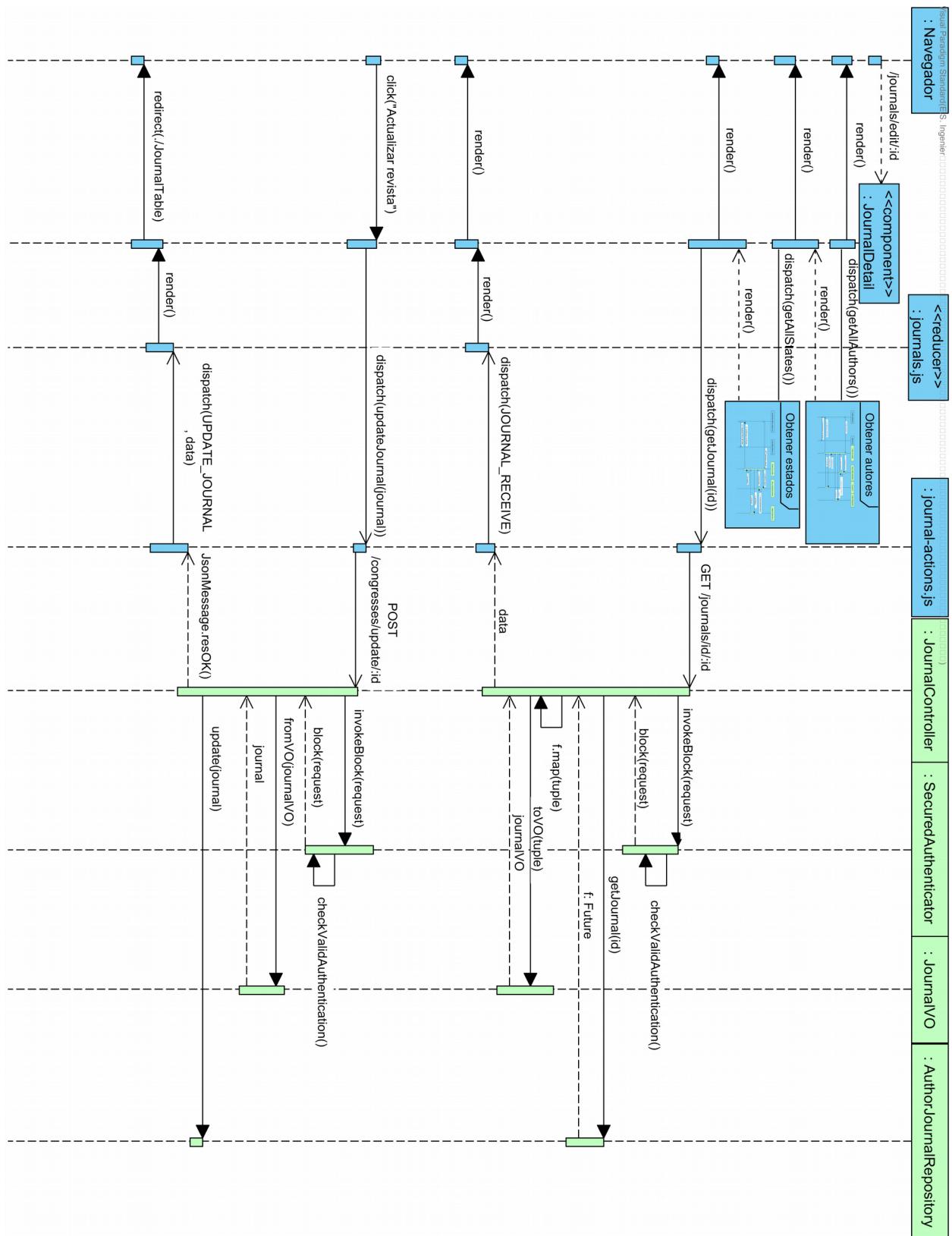


Figura 15: Diagrama de secuencia de añadir revista

## 2.11. Diagrama de secuencia de actualizar revista



*Figura 16: Diagrama de secuencia de actualizar revista*

## 2.12. Diagrama de secuencia de eliminar revista

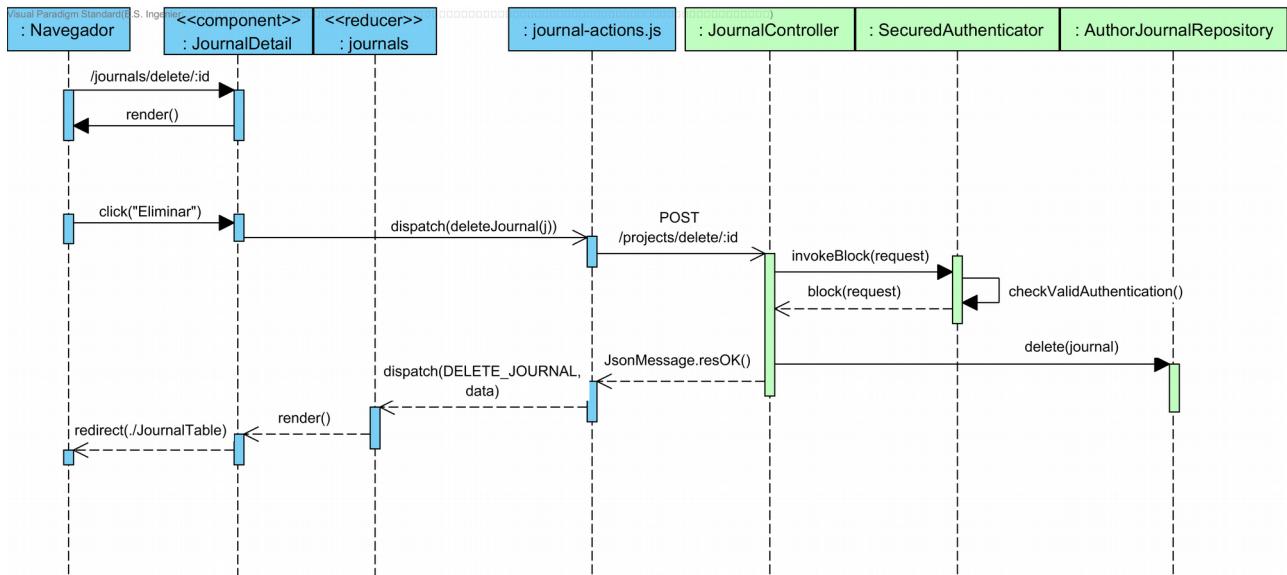


Figura 17: Diagrama de secuencia de eliminar revista

## 2.13. Diagrama de secuencia de consultar lista de proyectos

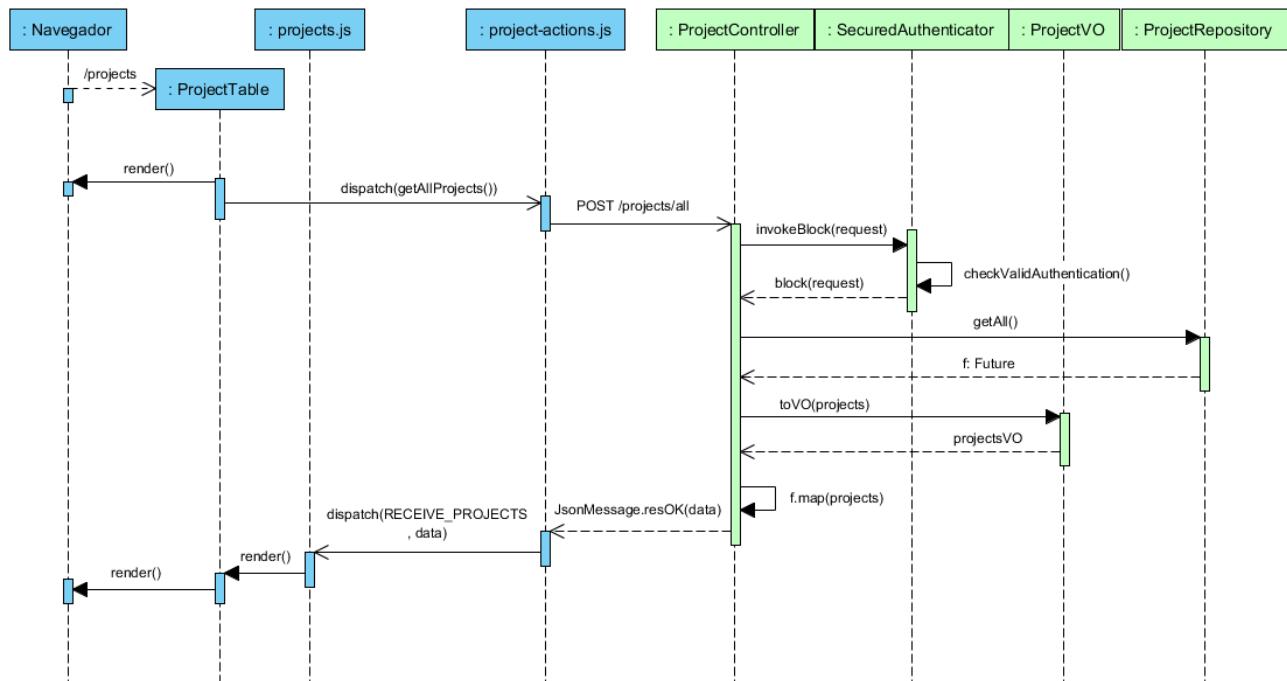


Figura 18: Diagrama de secuencia de consultar lista de proyectos

## 2.14. Diagrama de secuencia de añadir proyecto

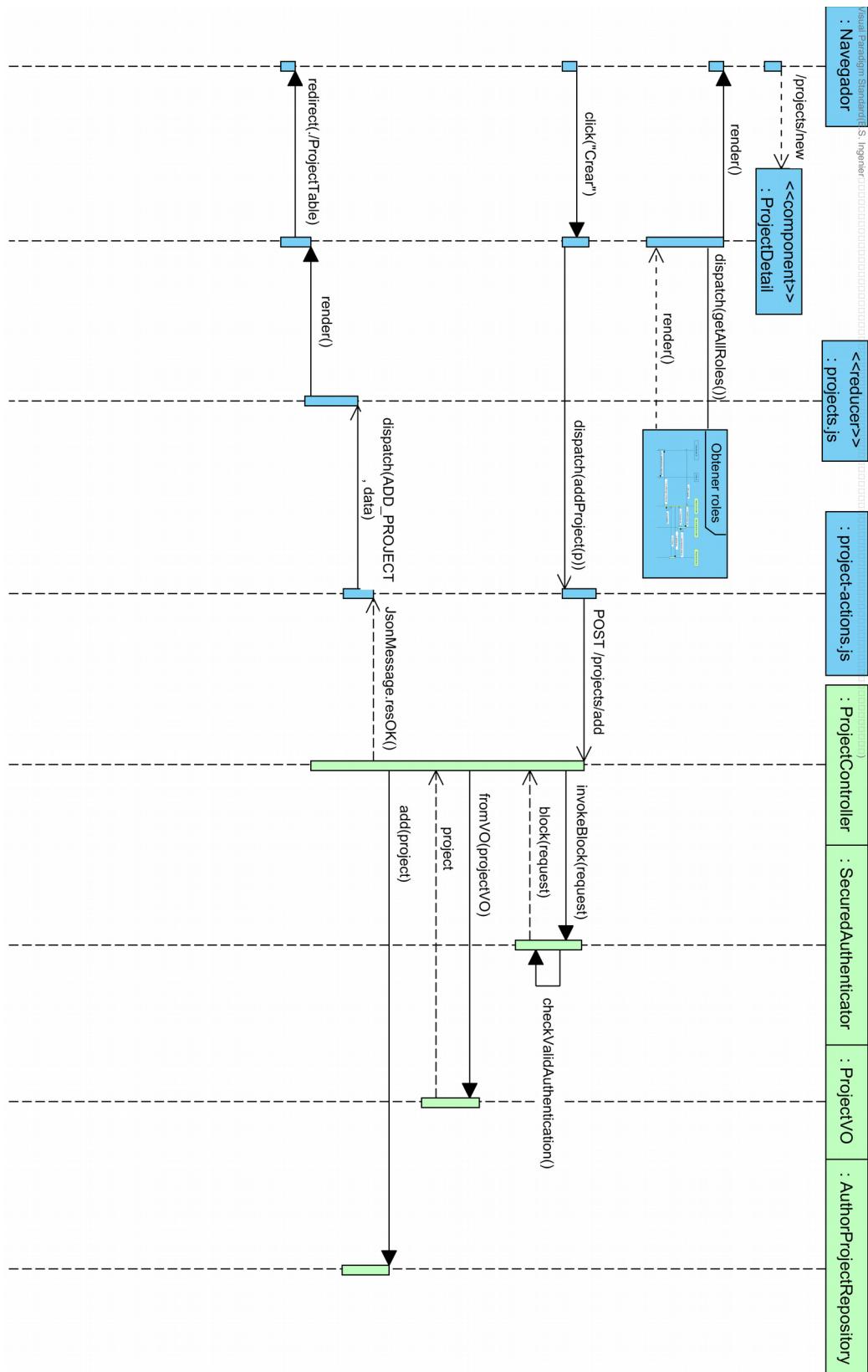


Figura 19: Diagrama de secuencia de añadir proyecto

## 2.15. Diagrama de secuencia de actualizar proyecto

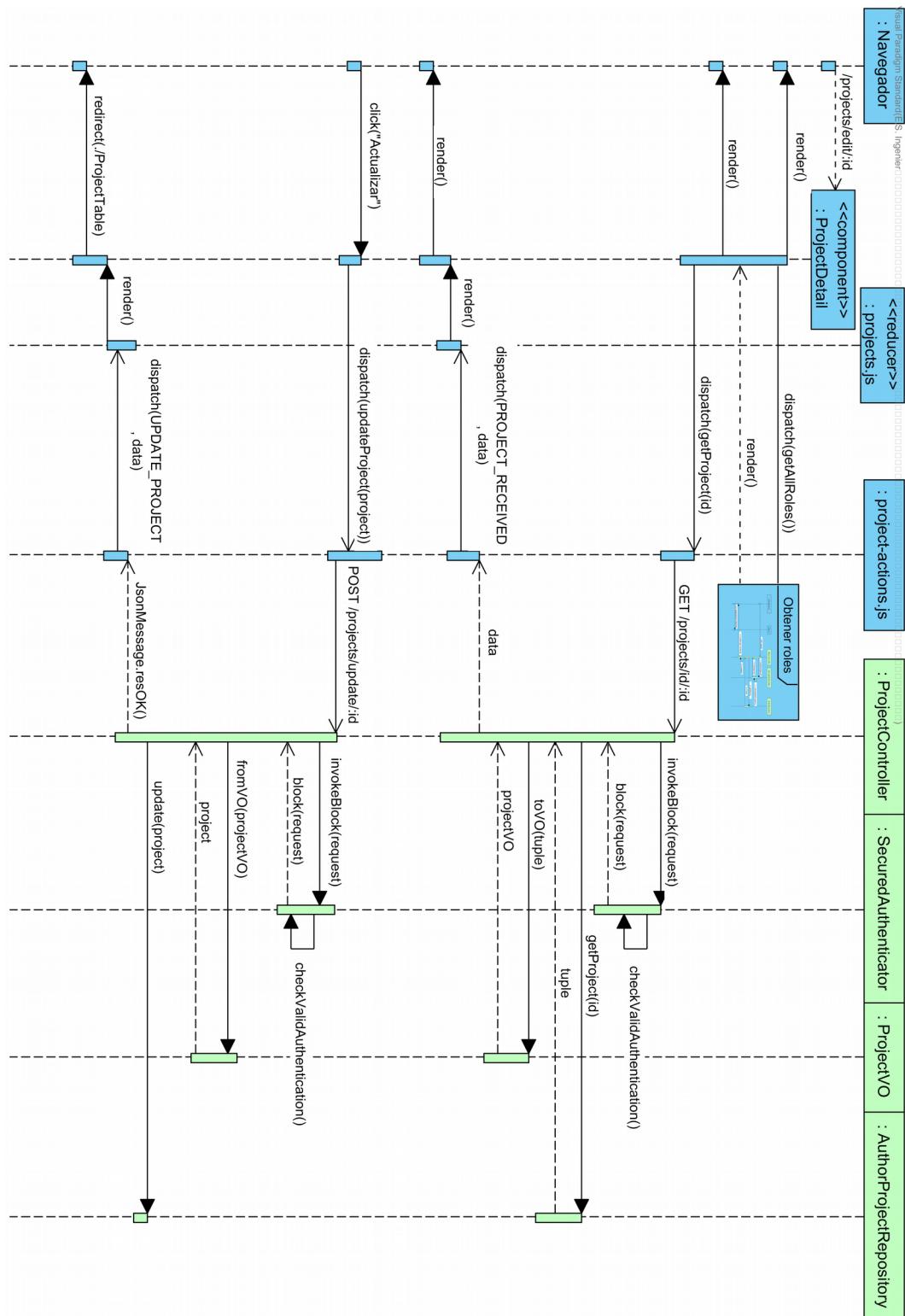


Figura 20: Diagrama de secuencia de actualizar proyecto

## 2.16. Diagrama de secuencia de eliminar proyecto

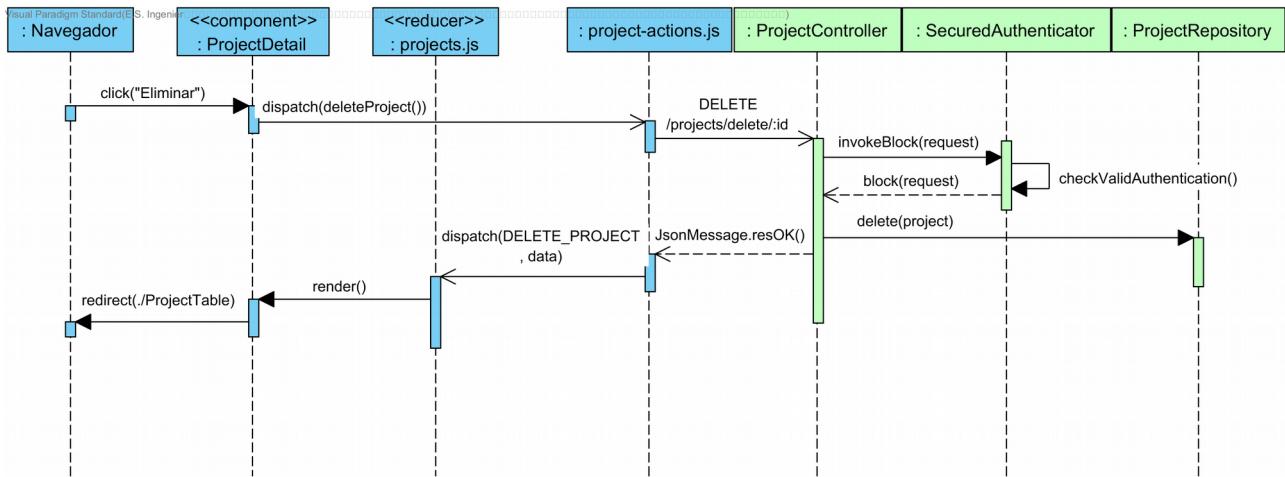


Figura 21: Diagrama de secuencia de eliminar proyecto

## 2.17. Diagrama de secuencia de consultar lista de congresos

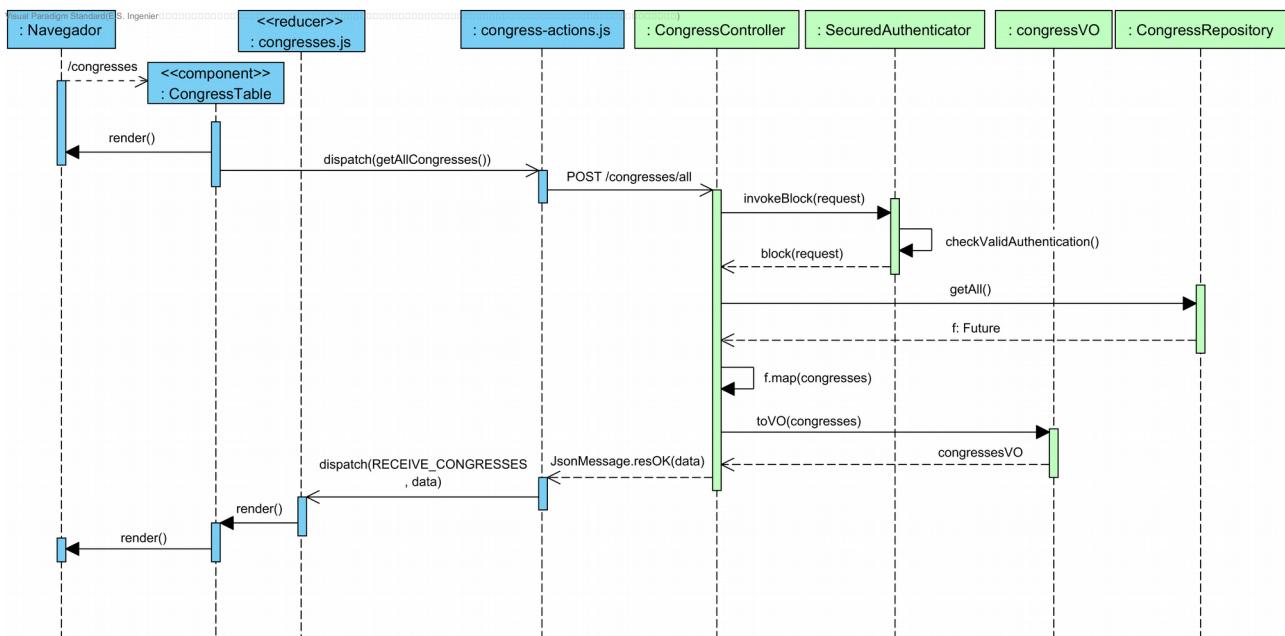


Figura 22: Diagrama de secuencia de consultar lista de congresos

## 2.18. Diagrama de secuencia de añadir congreso

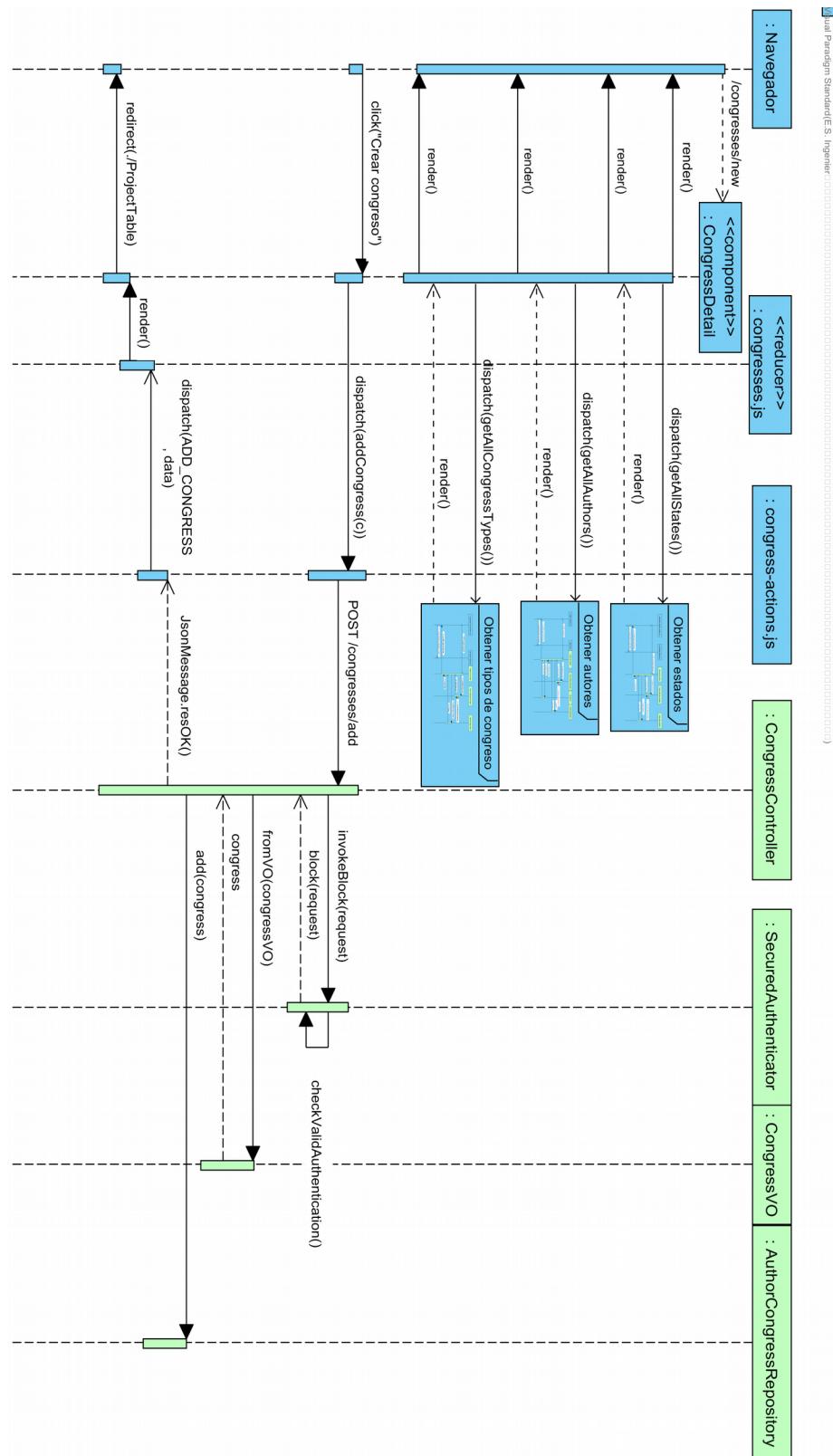


Figura 23: Diagrama de secuencia de añadir congreso



# **Anexo VI: Pruebas detalladas**



# Índice

1. Escenario de identificación.....	1
2. Escenario de salida del sistema.....	1
3. Escenarios de creación y actualización de un investigador.....	1
4. Eliminación de un investigador.....	2
5. Creación/actualización de un autor.....	2
6. Eliminación de un autor.....	2
7. Inserción o modificación de un proyecto.....	3
8. Eliminación de un proyecto.....	3
9. Inserción o modificación de un congreso.....	3
10. Eliminación de un congreso.....	4
11. Inserción o modificación de una publicación en revista.....	4
12. Eliminación de una publicación en revista.....	5
13. Visualización de calendarios.....	5



## Índice de Tablas

Tabla 1: Pruebas: escenario de identificación 1.....	1
Tabla 2: Pruebas: escenario de identificación 2.....	1
Tabla 3: Pruebas: escenario de salida del sistema 1.....	1
Tabla 4: Pruebas: escenario de creación de un investigador 1.....	1
Tabla 5: Pruebas: escenario de actualización de un investigador 1.....	2
Tabla 6: Pruebas: escenario de eliminación de un investigador 1.....	2
Tabla 7: Pruebas: escenario de creación/actualización de un autor 1.....	2
Tabla 8: Pruebas: escenario de eliminación de un autor 1.....	2
Tabla 9: Pruebas: escenario de inserción o modificación de un proyecto 1.....	3
Tabla 10: Pruebas: escenario de eliminación de un proyecto 1.....	3
Tabla 11: Pruebas: escenario de inserción o modificación de un congreso 1.....	3
Tabla 12: Pruebas: escenario de inserción o actualización de un congreso 2.....	3
Tabla 13: Pruebas: escenario de eliminación de un congreso 1.....	4
Tabla 14: Pruebas: escenario de inserción o modificación de publicación en revista 1.....	4
Tabla 15: Pruebas: escenario de inserción o modificación de publicación en revista 2.....	4
Tabla 16: Pruebas: escenario de eliminación de una publicación en revista.....	5
Tabla 17: Pruebas: escenario de visualización de calendario.....	5



## 1. Escenario de identificación

<b>Descripción</b>	Escenario que cubre la acción de realizarse correctamente en el sistema.
<b>Precondiciones</b>	El usuario no puede estar identificado en el sistema.
<b>Entrada esperada</b>	El usuario introduce el nombre de usuario: “ <a href="mailto:admin@admin.com">admin@admin.com</a> ” y la contraseña “1234”.
<b>Salida esperada</b>	El usuario es redirigido a la página de calendarios.
<b>Resultado</b>	El usuario se identifica correctamente en el sistema.

Tabla 1: Pruebas: escenario de identificación 1

<b>Descripción</b>	Escenario que cubre la acción intentar identificarse de manera errónea en el sistema.
<b>Precondiciones</b>	El usuario no puede estar identificado en el sistema.
<b>Entrada esperada</b>	El usuario introduce el nombre de usuario: “ <a href="mailto:admin@admin.com">admin@admin.com</a> ” y la contraseña “123456”.
<b>Salida esperada</b>	El usuario recibe un mensaje de error en color rojo.
<b>Resultado</b>	El usuario sigue en la pantalla de login con un mensaje de error.

Tabla 2: Pruebas: escenario de identificación 2

## 2. Escenario de salida del sistema

<b>Descripción</b>	Escenario que cubre la acción de salir del sistema.
<b>Precondiciones</b>	El usuario debe estar identificado en el sistema.
<b>Entrada esperada</b>	El usuario pulsa en el botón “Salir” situado en la parte superior derecha de la pantalla.
<b>Salida esperada</b>	El usuario es redirigido a la página de identificación.
<b>Resultado</b>	El usuario está situado en la página de identificación.

Tabla 3: Pruebas: escenario de salida del sistema 1

## 3. Escenarios de creación y actualización de un investigador

<b>Descripción</b>	Escenario que cubre la creación de un nuevo investigador.
<b>Precondiciones</b>	El usuario debe estar identificado en el sistema y tener permisos de administrador.
<b>Entrada esperada</b>	El usuario pulsa en el botón “Nuevo investigador” y introduce los siguientes datos, por orden: <a href="mailto:prueba@prueba.es">prueba@prueba.es</a> , 1234, 12345, Marcos, Núñez, Curros Enríquez, 982533391, acceso usuario y administrador.
<b>Salida esperada</b>	El usuario recibe un mensaje en rojo indicando que las contraseñas no coinciden.
<b>Resultado</b>	El usuario recibe un mensaje de contraseñas no coincidentes. El usuario no puede ser almacenado hasta que no se corrija.

Tabla 4: Pruebas: escenario de creación de un investigador 1

<b>Descripción</b>	Escenario que cubre la actualización de un investigador.
<b>Precondiciones</b>	El usuario debe estar identificado en el sistema y no tener permisos de administrador. El usuario debe pulsar en el botón de su perfil, situado en la esquina superior derecha de la pantalla.
<b>Entrada esperada</b>	Cambiar el nombre por “Alex” y la dirección por “Calle Emilia Pardo Bazán”.
<b>Salida esperada</b>	El usuario es redirigido a la página de lista de investigadores con un mensaje de éxito.
<b>Resultado</b>	El usuario se ha actualizado y aparece en la lista de investigadores con sus datos correctos.

Tabla 5: Pruebas: escenario de actualización de un investigador 1

## 4. Eliminación de un investigador

<b>Descripción</b>	Escenario que cubre la eliminación de un investigador del sistema.
<b>Precondiciones</b>	El usuario debe tener permisos de administrador y estar situado en la pantalla de detalle de un investigador cualquiera, salvo el primero y principal de la aplicación, el cual no está permitido eliminar.
<b>Entrada esperada</b>	Pulsar en el botón “Eliminar”.
<b>Salida esperada</b>	El usuario es redirigido a la página de lista de investigadores con un mensaje de éxito.
<b>Resultado</b>	La lista de investigadores se ha actualizado y ya no existe el investigador buscado.

Tabla 6: Pruebas: escenario de eliminación de un investigador 1

## 5. Creación/actualización de un autor

<b>Descripción</b>	Escenario que cubre la creación o actualización de un investigador del sistema.
<b>Precondiciones</b>	El usuario debe tener permisos de administrador y, desde la pantalla del listado de autores, haber pulsado en “Nuevo autor” o en el botón “Detalle” de uno existente.
<b>Entrada esperada</b>	Introducir o modificar los siguientes datos: <a href="mailto:micorreo@gmail.com">micorreo@gmail.com</a> , Marcos Núñez-Celeiro. Si se desea se puede asociar un investigador.
<b>Salida esperada</b>	El usuario es redirigido a la página de lista de autores con un mensaje de éxito.
<b>Resultado</b>	La lista de autores se ha actualizado con el nuevo autor.

Tabla 7: Pruebas: escenario de creación/actualización de un autor 1

## 6. Eliminación de un autor

<b>Descripción</b>	Escenario que cubre la eliminación de un autor del sistema.
<b>Precondiciones</b>	El usuario debe tener permisos de administrador y estar situado en la pantalla de detalle de un autor cualquiera.
<b>Entrada esperada</b>	Pulsar en el botón “Eliminar”.
<b>Salida esperada</b>	El usuario es redirigido a la página de lista de autores con un mensaje de éxito.
<b>Resultado</b>	La lista de autores se ha actualizado y ya no existe el autor buscado.

Tabla 8: Pruebas: escenario de eliminación de un autor 1

## 7. Inserción o modificación de un proyecto

<b>Descripción</b>	Escenario que cubre la creación o actualización de un proyecto en el sistema.
<b>Precondiciones</b>	El usuario debe tener permisos de administrador y, desde la pantalla del listado de proyectos, haber pulsado en “Nuevo proyecto” o en el botón “Detalle” de uno existente.
<b>Entrada esperada</b>	Introducir o modificar los siguientes datos: 17A2224, Proyecto1, 02/10/2017, 04/10/2017, 100000. Si se desea se pueden añadir autores existentes.
<b>Salida esperada</b>	El usuario es redirigido a la página de lista de proyectos con un mensaje de éxito.
<b>Resultado</b>	La lista de proyectos se ha actualizado con el nuevo proyecto.

Tabla 9: Pruebas: escenario de inserción o modificación de un proyecto 1

## 8. Eliminación de un proyecto

<b>Descripción</b>	Escenario que cubre la eliminación de un proyecto del sistema.
<b>Precondiciones</b>	El usuario no puede tener permisos de administrador y estar situado en la pantalla de detalle de un proyecto en el que esté asignado.
<b>Entrada esperada</b>	Pulsar en el botón “Eliminar”.
<b>Salida esperada</b>	El usuario es redirigido a la página de lista de proyectos con un mensaje de éxito.
<b>Resultado</b>	La lista de autores se ha actualizado y ya no existe el autor buscado.

Tabla 10: Pruebas: escenario de eliminación de un proyecto 1

## 9. Inserción o modificación de un congreso

<b>Descripción</b>	Escenario que cubre la creación o actualización de un congreso en el sistema.
<b>Precondiciones</b>	El usuario debe tener permisos de administrador y, desde la pantalla del listado de proyectos, haber pulsado en “Nuevo congreso” o en el botón “Detalle” de uno existente.
<b>Entrada esperada</b>	Introducir o modificar los siguientes datos: PubCongreso1, Congreso1, Ourense, España, 02/10/2017, 04/10/2017, ACEPTADO, POSTER, internacional. Si se desea se pueden añadir autores.
<b>Salida esperada</b>	El usuario es redirigido a la página de lista de congresos con un mensaje de éxito.
<b>Resultado</b>	La lista de congresos se ha actualizado con el nuevo congreso.

Tabla 11: Pruebas: escenario de inserción o modificación de un congreso 1

<b>Descripción</b>	Escenario que cubre la creación de un congreso en el sistema.
<b>Precondiciones</b>	El usuario debe tener permisos de administrador y, desde la pantalla del listado de congresos, haber pulsado en “Nuevo congreso”.
<b>Entrada esperada</b>	Pulsar en el botón guardar sin introducir ningún dato.
<b>Salida esperada</b>	Los campos título, nombre, lugar y país deben mostrarse con un mensaje de error.
<b>Resultado</b>	El congreso no se inserta en el sistema. Se muestran errores en el formulario y se explica al usuario como solucionarlos.

Tabla 12: Pruebas: escenario de inserción o actualización de un congreso 2

## 10. Eliminación de un congreso

<b>Descripción</b>	Escenario que cubre la eliminación de un congreso del sistema.
<b>Precondiciones</b>	El usuario no puede tener permisos de administrador y debe estar situado en la pantalla de detalle de un congreso en el que participe como autor.
<b>Entrada esperada</b>	Pulsar en el botón “Eliminar”.
<b>Salida esperada</b>	El usuario es redirigido a la página de lista de congresos con un mensaje de éxito.
<b>Resultado</b>	La lista de congresos se ha actualizado y ya no existe el congreso.

Tabla 13: Pruebas: escenario de eliminación de un congreso 1

## 11. Inserción o modificación de una publicación en revista

<b>Descripción</b>	Escenario que cubre la creación o actualización de una publicación en revista en el sistema.
<b>Precondiciones</b>	El usuario debe tener permisos de administrador y, desde la pantalla del listado de revistas, haber pulsado en “Nueva revista” o en el botón “Detalle” de una existente.
<b>Entrada esperada</b>	Introducir o modificar los siguientes datos: R2234, publicacion1, revista1, 114, 133, 3:12, 08/10/2017, editorial1, lugar1, issn1, ENVIADO. Si se desea se pueden añadir autores.
<b>Salida esperada</b>	El usuario es redirigido a la página de lista de publicaciones en revista con un mensaje de éxito.
<b>Resultado</b>	La lista de publicaciones en revista se ha actualizado con la nueva publicación.

Tabla 14: Pruebas: escenario de inserción o modificación de publicación en revista 1

<b>Descripción</b>	Escenario que cubre la creación o actualización de una publicación en revista en el sistema.
<b>Precondiciones</b>	El usuario debe tener permisos de administrador y, desde la pantalla del listado de revistas, haber pulsado en “Nueva revista” o en el botón “Detalle” de una existente.
<b>Entrada esperada</b>	Deben dejarse todos los campos vacíos, salvo el de página de inicio, en el que se introducirá 120, y el de página fin en el que se introducirá 90. Después pulsar en el botón de guardar.
<b>Salida esperada</b>	La pantalla de detalle debe mostrar errores indicando que el campo está vacío bajo las siguientes cajas de texto: código, título, revista, fecha, editorial y lugar. También deben mostrarse errores bajo las cajas de texto de página de inicio y página de fin indicando que la de fin debe ser mayor que la de inicio.
<b>Resultado</b>	Se continúa en la pantalla de detalle con un muestreo correcto de errores. La revista no se añade/actualiza en el sistema.

Tabla 15: Pruebas: escenario de inserción o modificación de publicación en revista 2

## 12. Eliminación de una publicación en revista

<b>Descripción</b>	Escenario que cubre la eliminación de una publicación en revista del sistema.
<b>Precondiciones</b>	El usuario debe tener permisos de administrador y estar situado en la pantalla de detalle de una revista en el que participe como autor.
<b>Entrada esperada</b>	Pulsar en el botón “Eliminar”.
<b>Salida esperada</b>	El usuario es redirigido a la página de lista de publicaciones en revista con un mensaje de éxito.
<b>Resultado</b>	La lista de publicaciones en revista se ha actualizado y ya no existe la publicación.

Tabla 16: Pruebas: escenario de eliminación de una publicación en revista

## 13. Visualización de calendarios

<b>Descripción</b>	Escenario que cubre la visualización del calendario.
<b>Precondiciones</b>	El usuario debe estar identificado en el sistema.
<b>Entrada esperada</b>	Pulsar en el botón “Calendario”. Deben publicaciones en el sistema que tengan fecha de inicio o fin situada al mes actual.
<b>Salida esperada</b>	La aplicación muestra un calendario con actividades situadas en el día correcto.
<b>Resultado</b>	Se muestra el calendario con las actividades en su día y mes correcto.

Tabla 17: Pruebas: escenario de visualización de calendario