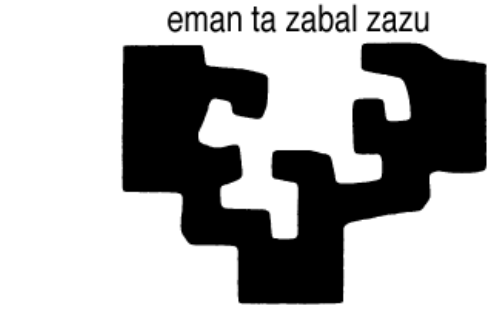


Facultad de Informática / Informatika

Sincronización con las API de Google en la plataforma de gestión empresarial AonSolutions

Alumno: D. Ander Ibáñez de Gauna Navazo

Director: D. German Rigau Claramunt

Proyecto Fin de Carrera, junio 2014

Agradecimientos

Antes de comenzar con la memoria, me gustaría agradecer a todas aquellas personas que me han ayudado en la realización de este proyecto, tanto en aspectos técnicos, como anímicos.

A todos los compañeros de AonSolutions, en especial a Raúl Trepiana, quien me ha ayudado durante todo el proceso del proyecto y a Julio García. A mi director de proyecto German Rigau.

A todos los amigos y compañeros que han estado conmigo a lo largo del todo el proyecto, los cuales han sido un gran apoyo en todo el proceso.

A mis padres por darme la oportunidad de realizar estos estudios superiores y por su apoyo a lo largo de toda la licenciatura.

Índice

1. INTRODUCCIÓN

1.1. Presentación

Este documento corresponde a la memoria del proyecto de fin de carrera a desarrollar por Ander Ibáñez de Gauna Navazo para la titulación de Ingeniería en Informática de la Facultad de Informática de San Sebastián. El proyecto consiste en la sincronización con ciertas API's de Google en la plataforma de gestión empresarial AonSolutions[1].

Dicho proyecto se realizará en conjunto con la empresa AonSolutions, teniendo como director del proyecto a Julio García y supervisión técnica a Raúl Trepiana por parte de dicha empresa, y con la supervisión como tutor del proyecto, German Rigau Claramunt por parte de la Universidad del País Vasco / Euskal Herriko Unibertsitatea.

1.2 AonSolutions

Imágen 1.1: Logo de AonSolutions.

AonSolutions es una plataforma de gestión empresarial para asesorías, empresas, autónomos y profesionales, que integra en un único sistema, todos los procesos de gestión de negocio de forma centralizada, evitando duplicidades y datos erróneos.

Éste cuadro (imagen 1.2) representa las extensiones de la plataforma AonSolutions, extensiones contratables tanto de forma individual como conjunta, conformando la solución a la medida de cada empresa.

Imágen1.2:Resúmen de la plataforma AonSolutions.

[1] Sitio web de AonSolutions: http://www.aonsolutions.es/

1.3. Motivación

AonSolutions es una plataforma de gestión empresarial, a la cual se le añadirán varias aplicaciones, con la sincronización de Google API's. En concreto se utilizarán:

- Sign-in with Google
- Google Calendar API
- Google Tasks API
- Google Drive API

Para empezar, se dispondrá de la opción de **iniciar sesión** en la plataforma AonSolutions **con la cuenta de Google**, para así conseguir una mayor seguridad en el inicio de sesión, liberar el tratamiento de contraseñas cifradas en la base de datos y la petición de permisos para el uso de las aplicaciones descritas anteriormente.

Se sincroniza **Google calendar** a AonSolutions con la intención de externalizar la agenda comercial a Google Calendar, de tal manera que cada participante en cada evento de la agenda comercial de AonSolutions, disponga de dicho evento en su cuenta personal de Google Calendar. De esta manera se consigue una mayor disponibilidad de los eventos de dicho calendario, adaptándose a todos los dispositivos en los que se pueda utilizar Google Calendar. *Ej. Smartphones, Tablets,…*

La sincronización con **Google Tasks** está ligeramente relacionado con Google Calendar, ya que cada tarea de Google Task aparece en el calendario de Google. El objetivo es prácticamente el mismo que con Google Calendar, pero sincronizando las tareas de cada proyecto.

Para la sincronización con **Google Drive**, se pretende disponer de un mayor espacio para la gestión de archivos en AonSolutions, guardando todos los archivos como, facturas, graficas, presentaciones, etc. en Google Drive, en vez de en la base de datos. Además se añadirá una nueva sección donde cada usuario pueda tener acceso a documentos personales desde AonSolutions.

Con todo esto se pretende facilitar el uso de AonSolutions entre los usuarios que dispongan de una cuenta de Google, liberar espacio en la base de datos, aumentar el espacio disponible para cada usuario/empresa y obtener una mayor seguridad en el inicio de sesión de AonSolutions.

1.3.1. Motivación Personal

En los últimos años, cada vez más son las aplicaciones que se ejecutan en la nube, dejando de lado en cierta medida a las aplicaciones de escritorio.

Trabajar con aplicaciones de Google, supone un plus en motivación, para comprender este tipo de aplicaciones y poder sacarles el mayor partido posible en una plataforma de gestión empresarial, como la utilizada en este proyecto.

1.4. Organización del Documento

Tras esta breve introducción presentamos la **planificación del proyecto** (DOP), donde se especificarán los objetivos, el método de trabajo utilizado, el alcance, las actividades, el plan de calidad, el plan de comunicaciones, el plan de cambios, el plan de contingencias y la viabilidad del proyecto

Después, se hará una breve introducción del **entorno de trabajo** del que se dispone antes de comenzar con el proyecto. Describiendo la plataforma AonSolutions y las API's deGoogle.

En las siguientes secciones, tendremos un apartado por cada iteración del proyecto, especificando en cada apartado, la captura de requisitos, el análisis, el diseño, la implementación, las pruebas y la implantación de su correspondiente iteración.

Luego, se describirá la **gestión del proyecto**, donde se podrá observar las diferencias entre lo que se ha planificado y lo que realmente ha sucedido a lo largo del proyecto, los problemas surgidos, etc.

Por último, se hará un análisis de todo el proyecto, en el apartado de **conclusiones**, comprobando si se han conseguido los objetivos, si el proyecto tiene continuación en un futuro y la experiencia obtenida en dicho proyecto.

2. DOP

2.1. Objetivos

- Utilizar diferentes herramientas de Google para la mejora de la plataforma de gestión empresarial AonSolutions, y así lograr que el cliente tenga más opciones y facilidades con la plataforma en la que se está trabajando.
- Utilizar la cuenta de Google para identificarse en la plataforma, y de esta manera conseguir una mayor seguridad.

2.2. Método de trabajo

El método de trabajo que se va a utilizar para la realización de este proyecto es Rational Unified Process (RUP), diferenciando las siguientes fases a lo largo de dicho proyecto.

Fase 0 → Formación
Fase 1 → DOP
Fase 2 → 1º iteración (Signin with Google)
Fase 3 → 2º iteración (Google Calendar)
Fase 4 → 3º iteración (Google Task)
Fase 5 → 4º iteración (Google Drive)
Fase 6 → Presentación del Proyecto

Tal y como se puede observar dentro de estas 6 fases se han definido 4 iteraciones, correspondientes a distintas funcionalidades que se le van a otorgar a la plataforma de gestión empresarial AonSolutions.

1ºIteración: La primera iteración corresponde a la autenticación para conectarse a la plataforma con la cuenta de usuario de Google.

2ºIteración: Se llevará a cabo la sincronización de la agenda de la plataforma con el calendario de Google.

3ºIteración: En esta iteración se sincronizarán las tareas de la plataforma AonSolutions con las tareas de la aplicación Task de google de cada usuario.

4ºIteración: Se agregará la funcionalidad de poder disponer los documentos que se manejan en la plataforma en Google Drive.

Por cada iteración se realizarán los siguientes pasos:

- *Captura de Requisitos*
  - Requisitos: Declaraciones de los servicios que debe proporcionar el sistema.
  - requisitos no funcionales: Restricciones de los servicios o funciones ofrecidas por el sistema.
  - modelo de casos de uso: Descripción de los pasos o actividades que deberán realizarse para llevar a cabo un proceso.
  - modelo de dominio: Recoge los tipos de objetos/clases más importantes.
  - Actores: Conjunto de papeles de una entidad exterior en relación con el sistema de software considerado.
  - Descripción: Descripción del caso de uso que se está analizando.
  - Precondiciones: Suposiciones acerca del estado del sistema antes de ejecutar la operación.

    - Post-condiciones: El estado del sistema después de ejecutar la operación.
    - Referencias a los requisitos: Números de referencia de las funciones del sistema, casos de uso, etc.
    - Escenario principal: Pasos realizados en la interacción entre los actores y el sistema o entre el sistema.

- *Análisis*
    - diagrama de secuencia del sistema: Representación que muestra, para un determinado caso de uso, los eventos generados por los actores externos, su orden y los eventos del sistema.
    - Contratos: Describen los efectos de las operaciones del sistema.

- *Diseño*: Producir un modelo o representación de una entidad que se va a construir posteriormente.
- *Implementación*: Codificación de la representación del software realizada en el diseño.
- *Pruebas*: Aseguramiento del funcionamiento del sistema.
- *Implantación*: Puesta en funcionamiento del servicio de la aplicación desarrollada.

2.3 Alcance

2.3.1. Recursos Humanos

- Director de proyecto: El responsable en dirigir el proyecto a nivel de la Universidad del País Vasco/Euskal Herriko Unibertsitatea es German Rigau.
- Director AonSolutions: El responsable en dirigir el proyecto por parte de la empresa AonSolutions es Julio García.
- Supervisor AonSolutions: El supervisor en aspectos técnicos del proyecto por parte de la empresa AonSolutions es Raúl Trepiana.
- Realizador del Proyecto: El encargado de realizar este proyecto es Ander Ibáñez de Gauna

2.3.2. Recursos Materiales

Los recursos materiales necesarios para la realización de este proyecto son los siguientes:

- Portátil Personal: Utilizado para la realización de la documentación y la implementación del proyecto.
- Ordenador de la empresa: Utilizado para la implementación del proyecto en conjunto con el portátil personal.
- Servidores de la empresa AonSolutions (Esferalia): En este proyecto se utilizan para guardar en el repositorio las implementaciones realizadas, y para lanzar la plataforma en la que se realiza dicho proyecto.

2.3.3. Recursos económicos

Sin tener en cuenta, los sueldos de los empleados de la empresa, en un principio el gasto del proyecto de 0 Euros, con la posibilidad de aumentar dicho coste, obteniendo el servicio de pago que Google ofrece para las empresas que desarrollan aplicaciones con las ya nombradas Google API’s.

2.3.4 Herramientas Utilizadas

Las herramientas técnicas necesarias para la realización de este proyecto son las siguientes:

- **Tomcat**: El servidor Tomcat es una aplicación web basada en Java creada para ejecutar servlets y páginas JSP.
- **Eclipse**: Es un entorno de desarrollo integrado, de código abierto y multiplataforma.
- **Maven**: Es una herramienta de software para la gestión y construcción de proyecto en java.
- **Google Drive**: Repositorio de archivos de Google, utilizado para almacenar toda la documentación correspondiente al proyecto.
- **Jenkins**: Es un software de integración continua open source escrito en Java para el desarrollo de software.
- **MySql Workbench**: es una herramienta visual de diseño de bases de datos para el sistema de base de datos MySQL.
- **Microsoft Project**: Es un software de administración de proyectos diseñado, desarrollado y comercializado por Microsoft.
- **Microsoft Word**: Es un software destinado al procesamiento de textos.
- **Microsoft Power Point**: Es un software destinado a la creación de presentaciones.
- **ArgoUML**: Es una aplicación para la realización de diagramas UML escrita en Java.
- **Jooq**: Es una biblioteca de software de base de datos en Java que implementa el modelo de registro activo.

2.3.5. EDT

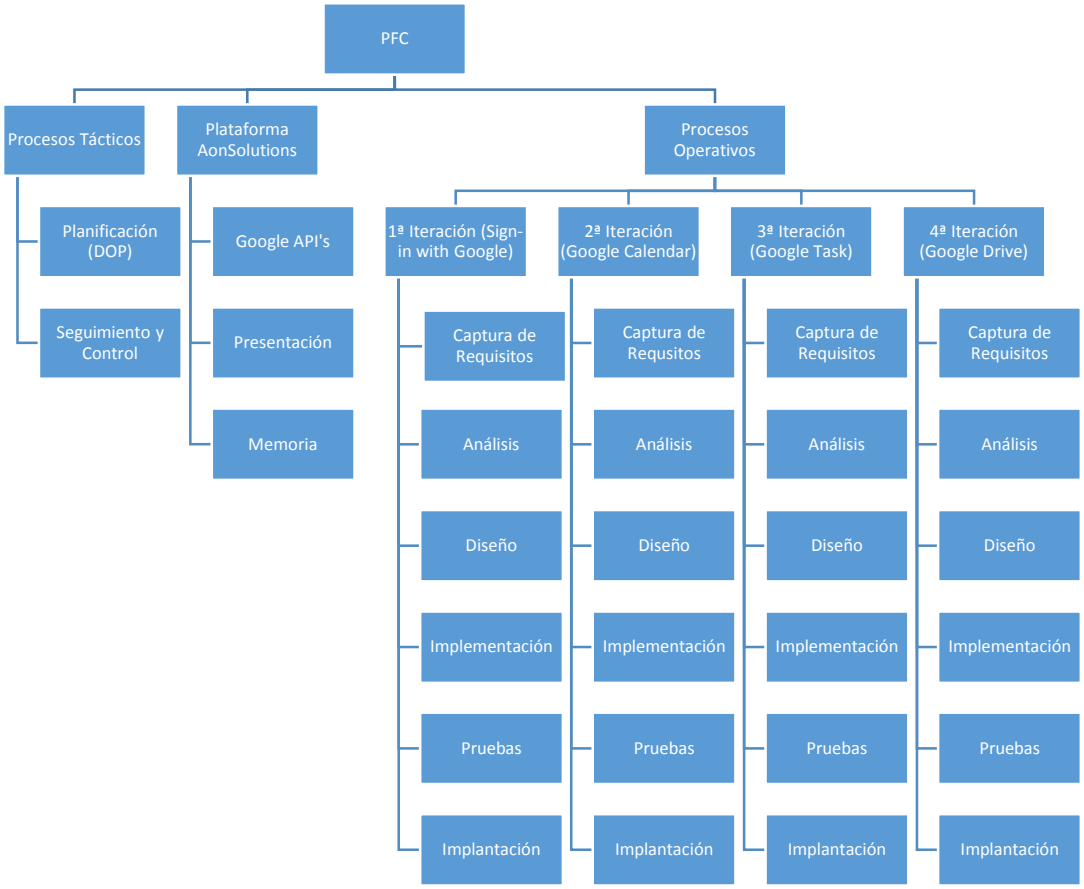


Imagen 2.1: Estructura de Desglose del Trabajo.

2.4. Actividades

2.4.1. Tareas

PROCESOS TÁCTICOS

- Planificación/DOP .....15 H.
- Seguimiento y control del proyecto.....20 H.

1ª ITERACIÓN SIGNIN WITH GOOGLE

- Captura de requisitos .....6 H.
- Análisis.....10 H.
- Diseño.....15 H.
- Implementación.....20 H.
- Pruebas.....10 H.
- Implantación.....5 H.
- Implantación.....10 H.

2ª ITERACIÓN G.CALENDAR

- Captura de requisitos .....6 H.
- Análisis.....10 H.
- Diseño.....13 H.
- Implementación.....18 H.
- Pruebas.....10 H.
- Implantación.....5 H.
- Implantación.....10 H.

3ª ITERACIÓN G.TASK

- Captura de requisitos .....8 H.
- Análisis.....12 H.
- Diseño.....17 H.
- Implementación.....22 H.
- Pruebas.....15 H.
- Implantación.....5 H.
- Implantación.....10 H.

4ª ITERACIÓN G.DRIVE

- Captura de requisitos .....6 H.
- Análisis.....10 H.
- Diseño.....15 H.
- Implementación.....20 H.
- Pruebas.....10 H.
- Implantación.....7 H.
- Implantación.....10 H.

PROCESOS FORMATIVOS

- AonSolutions.....10 H.
- Google API's .....15 H.
- Presentación .....15 H.
- Memoria .....50 H.

2.4.2. Esfuerzo Temporal

Tareas	***Horas***
Procesos Tácticos	35 H.
1ª Iteración	78 H.
2ª Iteración	76 H.
3ª Iteración	71 H.
4ª Iteración	89 H.
Procesos Formativos	90 H.
Total	439 H.

2.4.3. Planificación temporal

Diagrama de Gantt

	Nombre de tarea	Duración	Comienzo	Fin
1	**Fase 0 : Formación**	**5 días**	**lun 23/09/13**	**vie 27/09/13**
2	Familiarización con AonSolutions	2 días	lun 23/09/13	mar 24/09/13
3	Familiarización con Google API's	3 días	mié 25/09/13	vie 27/09/13
4	**Fase 1 : Gestión**	**141 días**	**mar 17/09/1**	**mar 01/04/14**
5	Planificación (DOP)	3 días	mar 17/09/13	jue 19/09/13
6	Seguimiento y Control	137 días	lun 23/09/13	mar 01/04/14
7	**Fase 2 : Google Calendar**	**21 días**	**lun 30/09/13**	**lun 28/10/13**
8	Captura de Requisitos	1 día	lun 30/09/13	lun 30/09/13
9	Análisis	3 días	mar 01/10/1	jue 03/10/13
10	Diseño	3 días	lun 07/10/13	mié 09/10/13
11	Implementación	7 días	jue 10/10/13	vie 18/10/13
12	Pruebas	2 días	lun 21/10/13	mar 22/10/13
13	Implantación	1 día	mié 23/10/1	mié 23/10/13
14	Entregable	3 días	jue 24/10/13	lun 28/10/13
15	**Fase 3 : Google Task**	**20 días**	**mar 29/10/1**	**lun 25/11/13**
16	Captura de Requisitos	1 día	mar 29/10/13	mar 29/10/13
17	Análisis	3 días	mié 30/10/1	vie 01/11/13
18	Diseño	3 días	lun 04/11/13	mié 06/11/13
19	Implementación	6 días	jue 07/11/13	jue 14/11/13
20	Pruebas	2 días	lun 18/11/13	mar 19/11/13
21	Implantación	1 día	mié 20/11/1	mié 20/11/13
22	Entregable	3 días	jue 21/11/13	lun 25/11/13

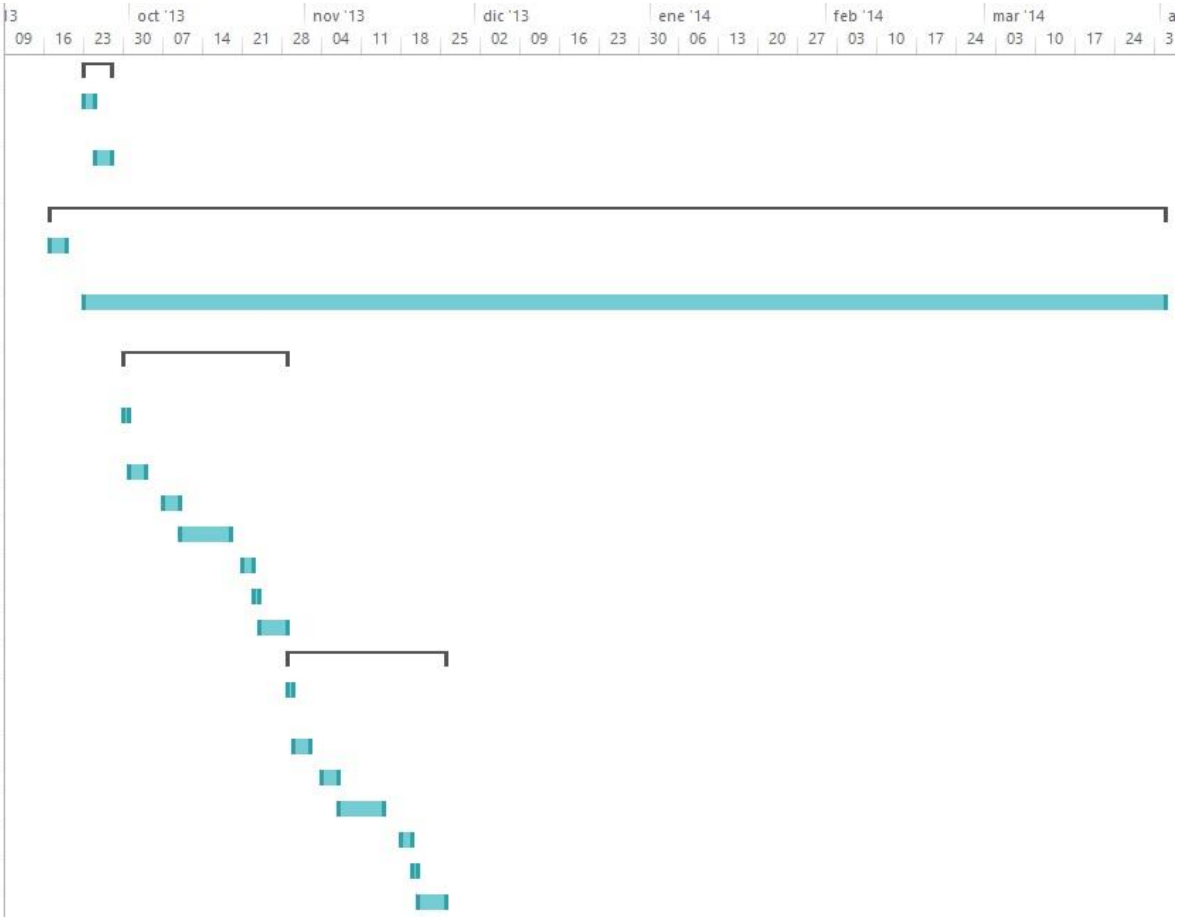


Imagen 2.2: Primera parte del Diagrama Gantt (Hasta la fase 3).

23	**Fase 4 : Google Drive**	**25 días**	**mar 26/11/13**	**lun 30/12/13**
24	Captura de Requisitos	2 días	mar 26/11/13	mié 27/11/13
25	Análisis	3 días	jue 28/11/13	lun 02/12/13
26	Diseño	3 días	mar 03/12/1:	jue 05/12/13
27	Implementación	9 días	lun 09/12/13	jue 19/12/13
28	Pruebas	2 días	vie 20/12/13	lun 23/12/13
29	Implantación	1 día	mar 24/12/1:	mar 24/12/13
30	Entregable	3 días	jue 26/12/13	lun 30/12/13
31	**Fase 5 : OpenID**	**22 días**	**lun 27/01/14**	**mar 25/02/14**
32	Captura de Requisitos	1 día	lun 27/01/14	lun 27/01/14
33	Análisis	3 días	mar 28/01/1	jue 30/01/14
34	Diseño	3 días	lun 03/02/14	mié 05/02/14
35	Implementación	7 días	mié 05/02/1	jue 13/02/14
36	Pruebas	2 días	lun 17/02/14	mar 18/02/14
37	Implantación	1 día	mié 19/02/1	mié 19/02/14
38	Entregable	4 días	jue 20/02/14	mar 25/02/14
39	**Fase 6 : Presentación**	**22 días**	**lun 03/03/14**	**mar 01/04/14**
40	Memoria	15 días	lun 03/03/14	vie 21/03/14
41	Presentación	7 días	lun 24/03/14	mar 01/04/14
42	Revisión / Colchón	33 días	mié 02/04/1	vie 16/05/14

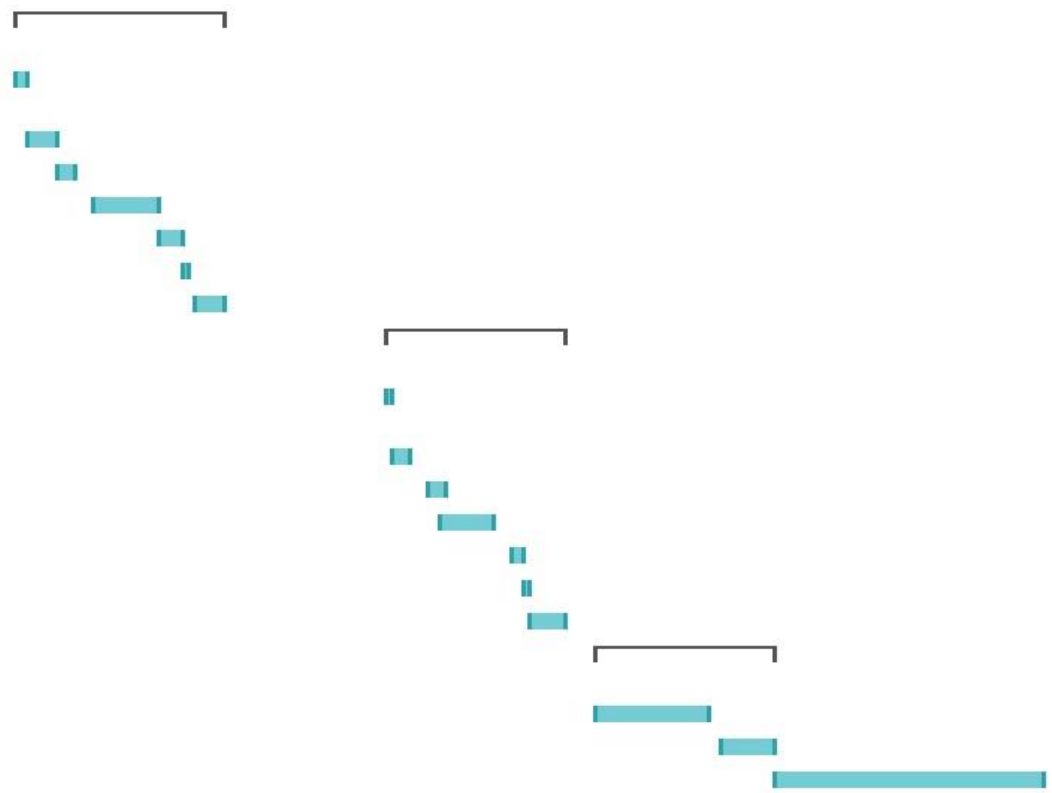


Imagen 2.3: Segunda parte del Diagrama Gantt.

Diagrama de Hitos

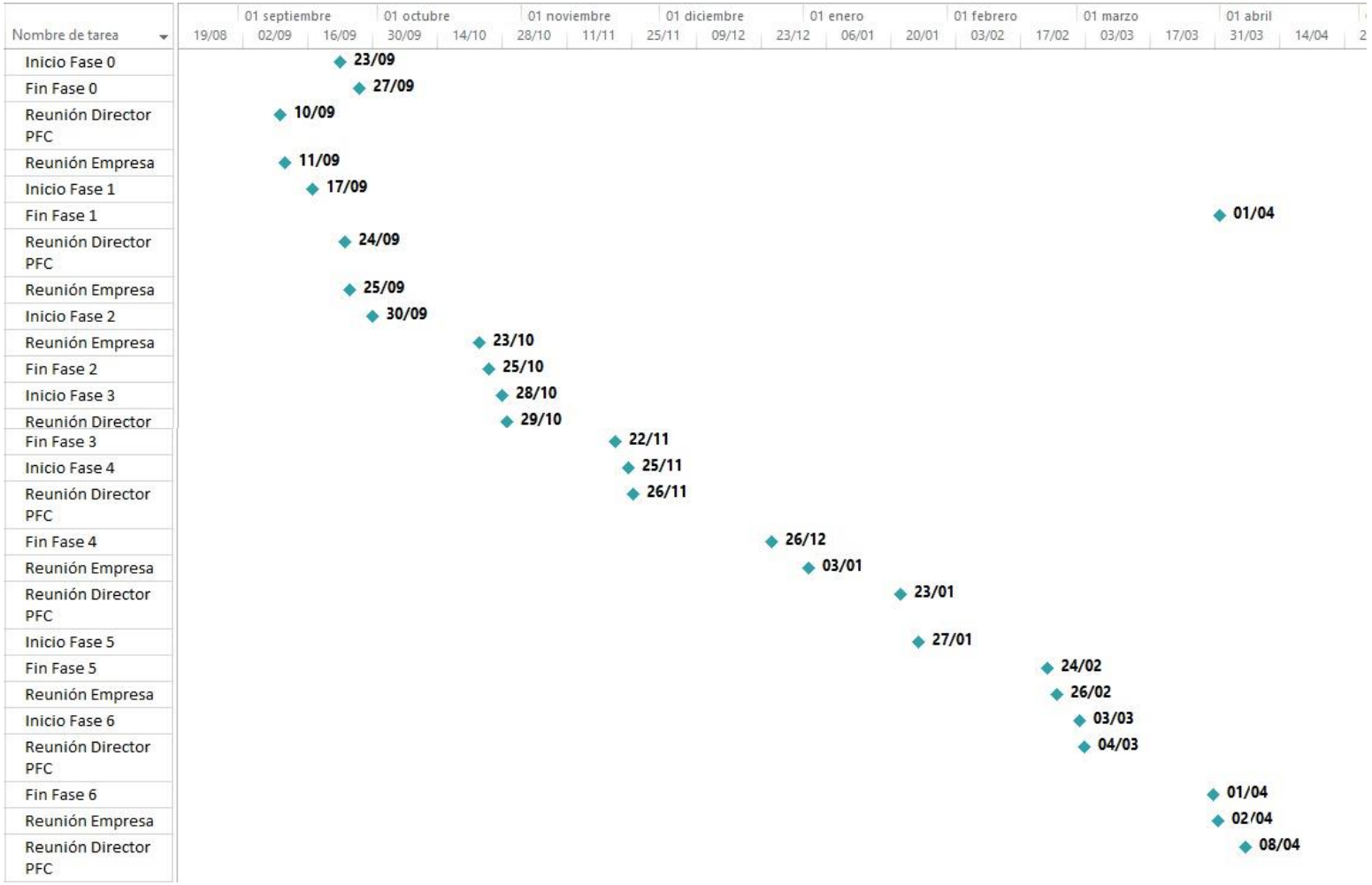


Imagen 2.4: Diagrama de Hitos.

2.5. Plan de comunicación

2.5.1. Identificación de los interesados

Interesado	*Rol*	*Correo electrónico*	*Móvil*
German Rigau	Director del Proyecto (UPV/EHU)	Confidencial	Confidencial
Julio García	Director de Proyecto (AonSolutions)	Confidencial	Confidencial
Raúl Trepiana	Supervisor del Proyecto (AonSolutions)	Confidencial	Confidencial
Ander Ibáñez de Gauna	Desarrollador del Proyecto (Alumno)	aibanezdegau004@gmail.com	Confidencial

2.5.2. Planificación de las comunicaciones.

En la siguiente tabla, se especificará las diferentes necesidades de información que tienen cada uno de los interesados.

Las comunicaciones serán vía correo electrónico como medio principal y en caso de no estar disponible por este medio se utilizara el teléfono para contactar con el receptor, el idioma establecido es el castellano.

Se concertarán reuniones al principio y al final de cada fase.

Asunto	**Emisor**	**Receptor**
Concertar Reunión + Respuesta del Receptor	Desarrollador del Proyecto (Alumno)	Director de Proyecto (UPV/EHU)
Concertar Reunión +Respuesta del Receptor	Desarrollador del Proyecto (Alumno)	Director de Proyecto y Supervisor del Proyecto (AonSolutions)
Dudas / Problemas +Respuesta del Receptor	Desarrollador del Proyecto (Alumno)	Director de Proyecto (UPV/EHU)
Dudas / Problemas +Respuesta del Receptor	Desarrollador del Proyecto (Alumno)	Director de Proyecto y Supervisor del Proyecto (AonSolutions)
Documentación Generada +Respuesta del Receptor	Desarrollador del Proyecto (Alumno)	Director de Proyecto (UPV/EHU)
Control del Proyecto +Respuesta del Receptor	Director de Proyecto (EHU/UPV)	Desarrollador del Proyecto (Alumno)
Control del Proyecto +Respuesta del Receptor	Director de Proyecto y Supervisor del Proyecto (AonSolutions)	Desarrollador del Proyecto (Alumno)

2.6. Plan de calidad

Teniendo en cuenta que el objetivo del proyecto es conseguir un nivel de calidad alto, se detallará el sistema de calidad que se va a utilizar a lo largo del proyecto, para así obtener los mejores resultados posibles.

2.6.1. Planificación de la Calidad

A continuación se listan las calidades que tendrán los diferentes componentes del proyecto.Es decir, cada iteración a no se dará por terminada hasta cumplir los niveles de calidad expuestos.

Sign-in with Google

- Se cumplirá con los requisitos que se especifican en la captura de requisitos de la iteración.
- El inicio de sesión deberá ser rápido, y no quedarse esperando un periodo de tiempo grande, para dar un buen servicio al usuario.
- Se pedirá permisos para las aplicaciones implementadas al iniciar sesión, de esta manera se consigue todo de una tirada, y no será necesario hacer dicha petición al utilizar cada aplicación por separado.

Google Calendar

- Se cumplirá con los requisitos que se especifican en la captura de requisitos de la iteración.
- La sincronización se llevara a cabo en un determinado tiempo, para asegurar la máxima rapidez de la aplicación.
- A cada evento sincronizado se le añadirá información extra (proyecto al que pertenece, cliente potencial, etc.), para un uso más completo por parte del usuario.
- Se utilizará una cuenta de servicio de Google, por lo que no es necesaria la autenticación del usuario, simplemente tiene que añadir su dirección de correo.

Google Task

- Se cumplirá con los requisitos que se especifican en la captura de requisitos de la iteración.
- La sincronización se llevara a cabo en un determinado tiempo, para asegurar la máxima rapidez de la aplicación.
- A cada tarea sincronizada se le añadirá información extra (proyecto al que pertenece,…), para un uso más completo por parte del usuario.

Google Drive

- Se cumplirá con los requisitos que se especifican en la captura de requisitos de la iteración.
- La sincronización se llevara a cabo en un determinado tiempo, para asegurar la máxima rapidez de la aplicación.
- Se controlará el tiempo de subida y descarga de los archivos, con la intención de ofrecer un servicio rápido al usuario.
- Se añadirá un apartado donde cada usuario pueda disponer de sus archivos personales en Google Drive.

2.6.2. Aseguramiento y control de la Calidad

Cada una de las iteraciones no se darán por terminadas hasta que cumplan los niveles de calidad propuestos anteriormente, consiguiendo así los objetivos del proyecto.

Para controlar la calidad de cada aplicación (iteración), se realizarán serie de pruebas con las herramientas de Junit, para asegurar el correcto funcionamiento, de todos los requisitos propuestos para cada una de las iteraciones.

Además personalmente se utilizarán las aplicaciones desarrolladas en un prototipo donde se comprobará el funcionamiento desde la cuenta personal de Google del desarrollador.

También, se comprobarán si el tiempo de respuesta el óptimo para la navegación del usuario en dicha plataforma.

Por último se implantará el sistema en el servicio de un cliente de confianza de la empresa, para disponer de una opinión de los nuevos servicios desarrollados, y poder realizar mejoras propuestas, en caso de que se disponga de tiempo de acuerdo con lo planificado y teniendo en cuenta la gestión de cambios.

2.7. Plan de contingencia

Riesgos	***Gravedad***	***Probabilidad***	***Medidas***
Pérdida de información, documentación y código del proyecto.	Alta	Media	El código se almacenará en el repositorio de la empresa, y los documentos en drive y disco duro.
Mala organización del trabajo, con retrasos en las tareas a realizar.	Alta	Media	En caso de que esto ocurra se reorganizaran los trabajos a realizar. También se ha planificado un cierto tiempo de colchón si sucede algún retraso.
Posibilidad de baja durante unos días, ya sea por enfermedad, problemas personales, etc.	Media	Media	Se ha planificado un cierto tiempo de colchón por si sucede algún retraso en el proyecto.
Fallo de algún servicio externo a la empresa.	Alta	Baja	Comunicar el error a los responsables del servicio y buscar otra alternativa con características similares.

Fallo de algún servicio interno de la empresa como roturas de servidores, etc.	Alta	Media	Comunicar el fallo a los responsables de dichos servicios dentro de la empresa, y esperar la solución.
Problemas con el software heredado dentro de la plataforma que influya a la parte correspondiente a este proyecto.	Media	Media	Comunicar el problema al desarrollador correspondiente.
Errores en la configuración de las herramientas utilizadas en la realización del proyecto.	Baja	Media	Utilizar el tiempo de colchón reservado para imprevistos.
Inexperiencia en las herramientas a utilizar en el proyecto.	Baja	Media	Se ha reservado un tiempo específico para la formación en las herramientas a utilizar en el proyecto.
Dificultad en la comunicación, ya sea con el director o con los operarios de la empresa.	Media	Media	Se ha planificado una reunión al inicio y final de cada fase, y se dispondrá de varios medios para la comunicación.
Disponibilidad de las dos partes en las reuniones planificadas, pueden causar retrasos en el proyecto por tomas de decisiones inadecuadas.	Media	Media	Tener un tiempo de colchón planificada ante retrasos. Solicitar la reunión con antelación para que no haya imprevistos de última hora.
Cambios inesperados en el proyecto.	Media	Alta	Se dispone de un plan de cambios y de un tiempo específico de colchón para cualquier imprevisto en el proyecto.

2.8. Gestión de Cambios

A medida que avanza el proyecto, éste puede someterse a muchos cambios, debido a mejores o necesidades del producto. Para gestionar adecuadamente los cambios que surjan durante el proyecto se va a definir un método de evaluación para dichos cambios, teniendo en cuenta que los costes no serán los mismos en una fase inicial del proyecto que en una fase final.

Se tendrán en cuenta los siguientes aspectos:

- **Viabilidad**: El director del proyecto y el supervisor del proyecto (AonSolutions) se reunirán con el desarrollador del proyecto para decidir la viabilidad del cambio a realizar, la toma de decisión será conjunta, teniendo más peso la decisión de los miembros de la empresa AonSolutions.
- **Recursos Afectados**: Las nuevas tareas a realizar se llevaran a cabo de manera que menos afecte a lo ya desarrollado del producto.
- **Ejecución de los cambios**: Una vez tomada la decisión el desarrollador del proyecto, procederá a realizar los cambios aceptados en el proyecto.

2.9. Viabilidad

Para la realización del proyecto el tiempo aproximado de trabajo es de unas 400 horas, la estimación que se ha realizado en este aspecto es satisfactorio debido a que aproximadamente se cumple con las 400 horas.

Los recursos necesarios para la realización del proyecto no tienen coste alguno, ya que se dispone de todo lo necesario para su realización.

Se dispone de un plan de contingencia con el cuál se pueden abordar todos los problemas que puedan surgir.

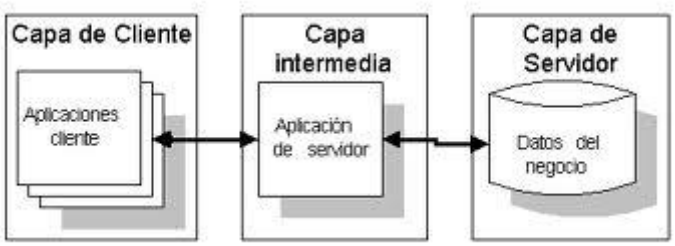
Por lo tanto, después de tener en cuenta estos aspectos, se considera la realización de este proyecto viable.

3. ENTORNO DE TRABAJO

3.1. Arquitectura de la plataforma AonSolutions

La arquitectura utilizada en la plataforma de gestión empresarial AonSolutions es una arquitectura por capas.

Arquitectura por capas: Es una arquitectura cliente-servidor en el que el objetivo principal es la separación de la lógica de negocios, de la lógica de diseño y de la lógica de datos, tal y como se puede observar en la imagen 3.11.



Imágen 3.11: Esquema de la arquitectura.

- **Capa de Presentación**: es la que ve el usuario (también se la denomina "capa de usuario"), presenta el sistema al usuario, le comunica la información y captura la información del usuario en un mínimo de proceso (realiza un filtrado previo para comprobar que no hay errores de formato). También es conocida como interfaz gráfica y debe tener la característica de ser "amigable" (entendible y fácil de usar) para el usuario. Esta capa se comunica únicamente con la capa de negocio.

- **Capa de la lógica de Negocio**: es donde residen los programas que se ejecutan, se reciben las peticiones del usuario y se envían las respuestas tras el proceso. Se denomina capa de negocio (e incluso de lógica del negocio) porque es aquí donde se establecen todas las reglas que deben cumplirse. Esta capa se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados, y con la capa de datos, para solicitar al gestor de base de datos almacenar o recuperar datos de él. También se consideran aquí los programas de aplicación.
- **Capa de Datos**: es donde residen los datos y es la encargada de acceder a los mismos. Está formada por uno o más gestores de bases de datos que realizan todo el almacenamiento de datos, reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio.

3.2. Entorno de trabajo de las Google API's

Imágen 3.12: Google API's

Este proyecto está basado en la integración de las aplicaciones de Google en la plataforma AonSolutions, más en concreto Google Calendar API, Google Tasks API y Google Drive API. Para ello Google dispone de las librerías en Java para poder sincronizar estas aplicaciones con la plataforma.

3.2.1. Google Calendar API

Google Calendar API permite desarrollar aplicaciones cliente que crean, editan, eliminan o buscan eventos o calendarios. Estan disponibles las bibliotecas para varios lenguajes de programación (en la actualidad Java, Python, PHP,. NET y Ruby) para acceder y editar los datos de Google Calendar. En el caso de este proyecto se usarán la librerias de Java.

Los sitios o aplicaciones que desean una mayor integración con Google Calendar pueden aprovechar la API de Google Calendar. Por ejemplo, para desarrollar una aplicación web para crear o mostrar datos de calendario, o una aplicación de escritorio que sincroniza el calendario de un usuario con una aplicación de escritorio existente, llevando la experiencia del calendario de Google a una nueva plataformas.

3.2.2. Google Tasks API

Google Tasks API ofrece a los desarrolladores un potente conjunto de criterios de valoración de la API para la búsqueda, lectura y actualización de contenidos de Google Tasks y metadatos. Estan disponibles las bibliotecas para varios lenguajes de programación (en la actualidad Java, Python y PHP) para acceder y editar los datos de Google Tasks.

La Tareas API Google puede ser útil para varios proyectos de integración con aplicaciones web u otras API de Google. Por ejemplo, para gestionar listas de tareas de Google en una aplicación móvil, o para integrar tareas en una más amplia aplicación de flujo de trabajo.

3.2.3. Google Drive API

Google Drive API da la posibilidad de desarrollar aplicaciones sincronizando Google drive para la búsqueda, subido o descarga de archivos. Están disponibles las bibliotecas para varios lenguajes de programación, en este caso se utilizarán las librerias de Java.

La funcionalidad principal de Google Drive API es la carga y descarga de archivos en Google Drive. Sin embargo, ofrece mucho más que un simple almacenamiento. Por ejemplo, creación de directorios, compartir archivos, etc.

3.2.4. OpenID

Es un estándar de identificación digital descentralizado, con el que un usuario puede identificarse en una página web a través de una URL (o un XRI en la versión actual) y puede ser verificado por cualquier servidor que soporte el protocolo.

En los sitios que soporten OpenID, los usuarios no tienen que crearse una nueva cuenta de usuario para obtener acceso. En su lugar, solo necesitan disponer de un identificador creado en un servidor que verifique OpenID, llamado proveedor de identidad o IdP.

El proveedor de identidad puede confirmar la identificación OpenID del usuario a un sitio que soporte este sistema.

3.2.5. Google Oauth 2 API

Google API's utiliza el protocolo OAuth 2.0 para autenticarse y hacer la petición de permisos con una cuenta de Google.

OAuth 2.0 es un protocolo abierto, que permite la autorización segura de una API de modo estándar y simple para aplicaciones de escritorio, móviles y web. Proporciona a los usuarios un acceso a sus datos al mismo tiempo que proyege las credenciales de su cuenta, en otras palabras, permite a un usuarios del sitio A compartir su información del sitio A (proveedor de servio) con el sitio B (consumidor) sin compartir toda su identidad.

3.3. Conexión con Google

Para realizar la conexión con Google en cada una de las aplicaciones se ha utilizado el protocolo Oauth 2, ya explicado en el anterior apartado. Entre todas las Google API's vamos a tener en cuenta 2 formas diferentes para la conexión,

En el primer caso será una autenticación de la propia cuenta del usuario o cliente, el cual deberá aceptar los permisos para el posterior tratamiento de datos por parte de la nueva aplicación.

Y en por otra parte una conexión a Google mediante una cuenta de servicio de la aplicación, es decir, una cuenta en Google de la aplicación, la cual no dispone de un servicio de interfaz gráfica de la cuenta. De esta manera el usuario no deberá aceptar ningún permiso, ya que, se compartiría la información necesaria desde la cuenta de servicio a la cuenta del usuario.

Para llevar a cabo dicha conexión es preciso completar una serie de pasos fundamentales.

1. **Obtener las credentiales OAuth 2.0 de *Google Developers Console*:**

   Acceder a la web de *Google Developers Console* con el objetivo de obtener la credentiales OAuth 2.0 para poder realizar la conexión con Google. Para ello hay que crear dichos credenciales, dependiendo del tipo de aplicación para la que se vayan a utilizar. En nuestro caso, ya que, vamos a utilizar dos opciones

eligiriamos por una parte las opciones para una aplicación de servidor, y por otra parte las opciones para una cuenta de servicio (*Service Account*)

2. **Obtener un token de acceso del servidor de autorización de Google.**

   Antes de que la aplicación pueda acceder a los datos privados de la cuenta utilizando cualquiera de las Google API, es necesario obtener un token de acceso a la API de Google. Un único token puede conceder distintos grados de acceso a las API, es decir, que mediante el parámetro Scope[1], puede contener información de acceso para múltiples Google API.

   Se disponen varias maneras para realizar esta petición, que varían dependiendo el tipo de aplicación que se este desarrollando.En el caso de esta aplicación se aprovecha la iteración 1 (Sign-in with Google) para obtener el token de acceso para todas las API de Google que necesiten el permiso de acceso por parte del usuario.

3. **Enviar el token de acceso a la API de Google**

   Una vez la aplicación ya haya obtenido el token de acceso, lo envía a la Google API correspondiente en un encabezado de autorizacón HTTP. Los token de acceso son válidos sólo para el conjunto de operaciones y recursos que se describen en el alcance de la solicitud del token.

4. **Actualizar el token de acceso (si fuera necesario)**

   Los tokens de acceso tienen una vida útil limitada. Si la aplicación necesita acceso a una API de Google más allá del tiempo de vida de un solo token de acceso, se puede obtener un token de actualización. Dicho token de actualización permite a la aplicación obtener nuevos tokens de acceso.

A continuación se describirán de una forma más detallada las dos opciones de acceso a las Google API que se van a utilizar en nuestro modulo, ya nombradas anteriormente, aplicación de servidor (con autenticación del usuario) y una cuenta de servicio de la propia aplicación (*Service Account*)

3.3.1. Aplicación de Servidor

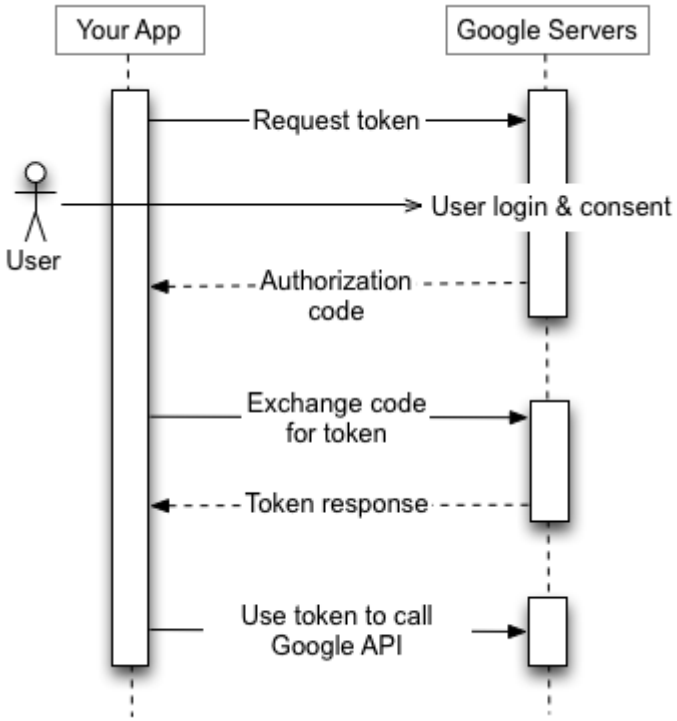
El endpoint de Google OAuth es compatible con aplicaciones de servidores web que utilizan lenguajes y frameworks como PHP, Java, Python, Ruby, y ASP.NET. En el caso de este proyecto se usará el lenguaje de programación java para la parte del servidor.

La secuencia de autorización comienza cuando su aplicación redirige el navegador a una dirección URL de Google. Dicha URL incluye parámetros de consulta que indican el tipo de acceso que se solicita. Google se encarga de la autenticación de usuario, la selección de sesiones, y el consentimiento del usuario. El resultado es un código de autorización, que la aplicación puede intercambiar para un token de acceso y un contador de refresco.

La aplicación debe almacenar el token de actualización para su uso futuro y utilizar el símbolo de acceso para acceder a una API de Google. Una vez que el token de acceso expira, la aplicación utiliza el token de actualización para obtener una nueva.

[1] Parámetro Scope: Los permisos que se le van a pedir al usuario.

Todo esto se puede observar en la imagen 4.2 del documento.



Imágen 4.2: Diagrama de secuencia conexión con Google.

3.3.2. Cuenta de Servicio (*Service Account*)

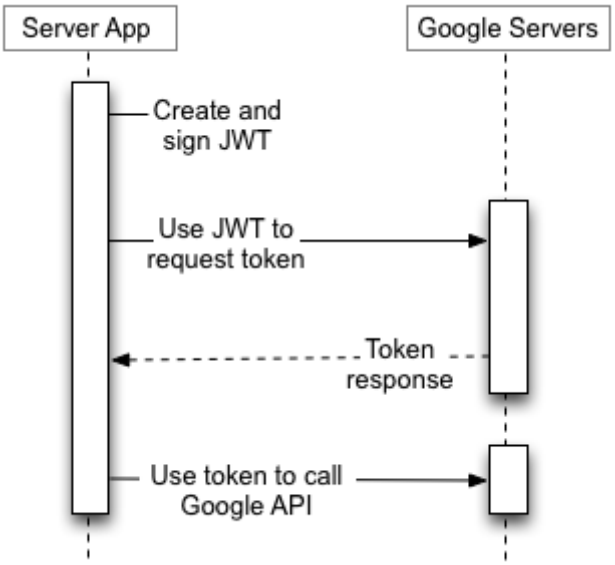
Las Google API's pueden actuar en "nombre" de la aplicación sin tener acceso a la información del usuario. En otras palabras, consequir el acceso a las Google API's sin iniciar sesión con la cuenta del usuario y sin el consentimiento del usuario.

Para estos tipos de interacciones de servidor a servidor se necesita una cuenta de servicio, que es una cuenta que pertenece a su aplicación en lugar de a un usuario. La aplicación llama a las API de Google en nombre de la cuenta de servicio, y tal y como se ha dicho antes, no se requiere el consentimiento del usuario.

Estos escenarios de servicios por la cuenta de servicio requieren aplicaciones para crear y firmar criptográficamente JSON Tokens web (JWTs). Para realizar estas tareas se ha utilizado la biblioteca Jackson JSON.

Las credenciales de una cuenta de servicio, que se obtiene de *Google Developers Console*, incluyen una dirección de correo electrónico generado que es única, un ID de cliente, y al menos un par de claves pública / privada. Se utiliza el ID de cliente y una clave privada para crear un JWT firmado y construye una solicitud para el token de acceso en un formato adecuado. La aplicación envía la solicitud a los servidores de autorización de Google OAuth 2.0, que devuelve un token de acceso. La aplicación utiliza el token para acceder a la API de Google. Cuando el token expira, la aplicación repite el proceso.

Todo esto se puede observar en la imagen 4.3 del documento.



Imágen 4.3: Diagrama de secuencia de la conexión con Google (Service Account).

3.4. Tecnologías Utilizadas

Java[1]: La plataforma de gestión empresarial AonSolutions está desarrollada mayoritariamente en Java, lenguaje de programación de propósito general, concurrente, orientado a objetos y basado en clases que fue diseñado específicamente para tener tan pocas dependencias de implementación como fuera posible. Su intención es permitir que los desarrolladores de aplicaciones escriban el programa una vez y lo ejecuten en cualquier dispositivo.

Imágen 3.3: Logo Java.

JavaServer Faces[2] **(JSF):** Es una tecnología y framework para aplicaciones Java basadas en web que simplifica el desarrollo de interfaces de usuario en aplicaciones Java EE. JSF usa JavaServer Pages (JSP) como la tecnología que permite hacer el despliegue de las páginas, pero también se puede acomodar a otras tecnologías como XUL.

Imágen 3.4: Logo JSF.

Eclipse[3]: Es una plataforma de desarrollo open source basada en Java. Es un desarrollo de IBM cuyo código fuente fue puesto a disposición de los usuarios. En sí mismo Eclipse es un marco y un conjunto de servicios para construir un entorno de desarrollo a partir de componentes conectados (plug-in).Hay plug-ins para el desarrollo de Java (JDT Java Development Tools) así como para el desarrollo en C/C++, COBOL, etc.

Imágen 3.5: Logo eclipse.

[1] Sitio web de Java: https://www.java.com/es/
[2] Sitio web de JSF: http://www.oracle.com/technetwork/java/javaee/javaserverfaces-139869.html
[3] Sitio web de eclipse: http://www.eclipse.org/

Maven[1]: Es una herramienta de software para la gestión y construcción de proyectos Java creada por Jason van Zyl, de Sonatype, en 2002. Es similar en funcionalidad a Apache Ant, pero tiene un modelo de configuración de construcción más simple, basado en un formato XML. Maven utiliza un Project Object Model (POM) para describir el proyecto de software a construir, sus dependencias de otros módulos y componentes externos, y el orden de construcción de los elementos.

Imágen 3.6: Logo Maven.

Tomcat[2]: Es uno de los desarrollos del proyecto Jakarta que crea y mantiene soluciones de código abierto basadas en Java. Tomcat, es uno de los proyectos de código abierto liderado por la *Apache Software Foundation*. El servidor Tomcat es una aplicación web basada en Java creada para ejecutar servlets y páginas JavaScript.

Imágen 3.7: Logo Tomcat.

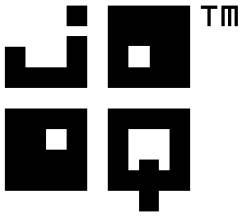
MySql[3]: Es un sistema gestor de bases de datos muy conocido y ampliamente usado por su simplicidad y notable rendimiento. Es una opción atractiva tanto para aplicaciones comerciales, como de entretenimiento precisamente por su facilidad de uso y tiempo reducido de puesta en marcha. Su libre distribución en internet bajo licencia GPL le otorga como beneficios adicionales contar con un alto grado de estabilidad y un rápido desarrollo.

Imágen 3.8: Logo MySql.

Jenkins[4]: Es un servidor de integración continua, antes conocido como Hudson. Nos facilita integrar los cambios que vayamos haciendo en nuestros proyectos ya que Jenkins los va construyendo de manera continua. Así, si hemos añadido código que ha "roto" el proyecto haciendo que algún test falle, Jenkins nos los hará saber.

Imágen 3.9: Logo Jenkins.

Jooq[5]**:** Es una biblioteca de software de base de datos en Java que implementa el modelo de registro activo. Su propósito es ser a la vez relacional y orientada al proporcionar un lenguaje específico del dominio para crear consultas de las clases generadas a partir de un esquema de base de datos de objetos.



Imágen 3.10: Logo jOOQ

[1] Sitio web de Maven: http://maven.apache.org/
[2] Sitio web de Tomcat: http://tomcat.apache.org/
[3] Sitio web de MySql: http://www.mysql.com/
[4] Sitio web de Jenkins: http://jenkins-ci.org/
[5] Sitio web de jOOQ: http://www.jooq.org/

4. ITERACIÓN 1: SIGN-IN WITH GOOGLE

4.1. Introducción

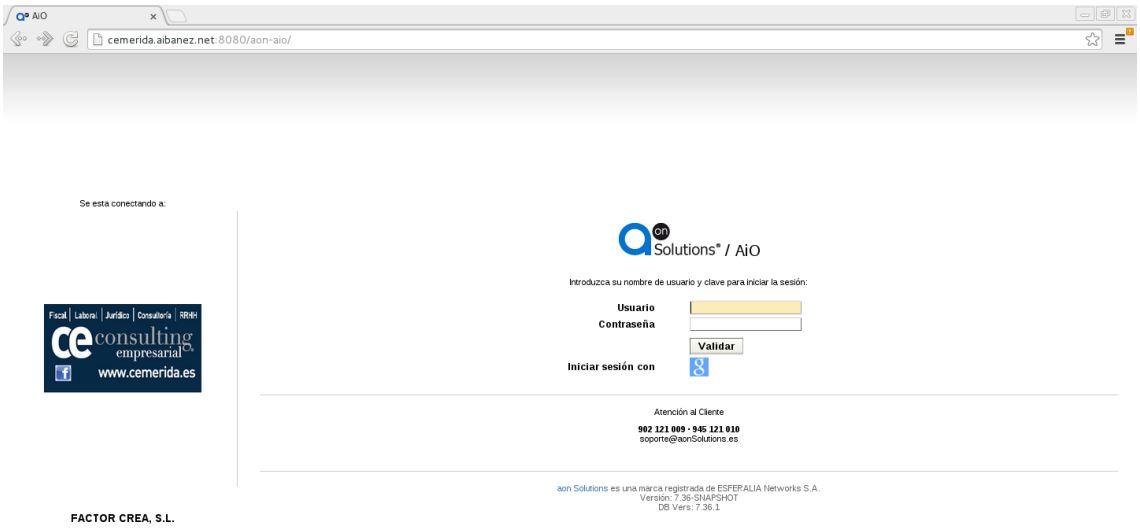
El objetivo de ésta iteración, es realizar el inicio de sesión de AonSolutions, mediante la cuenta de Google del usuario.

También se pedirán los permisos correspondientes de Google Task API y Google Drive API.

Para realizar este proceso se utiliza Google OAuth2 API, para conseguir los datos de la conexión con Google. Y se aprovecha parte del inicio de sesión con AonSolutions.

4.1.1. Ejecución Visual

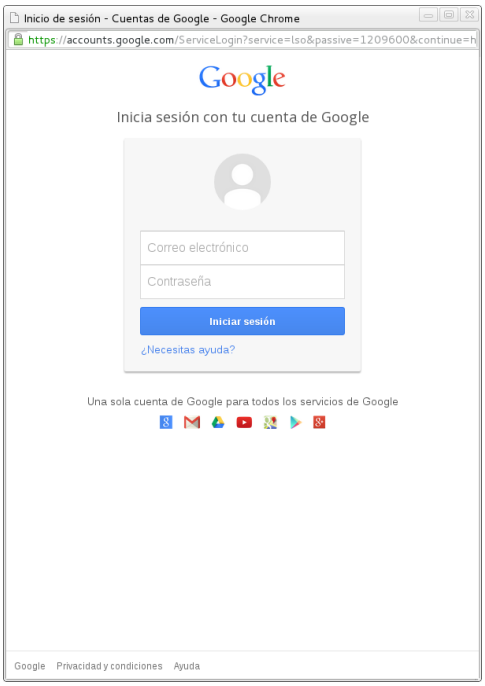
Se inicia el sistema de la plataforma de gestión empresarial, la primera pantalla que se divisará será el inicio de sesión, ya sea mediante usuario y contraseña o mediante una cuenta de Google (Icono de Google), se puede ver en la siguiente imagen 4.1.



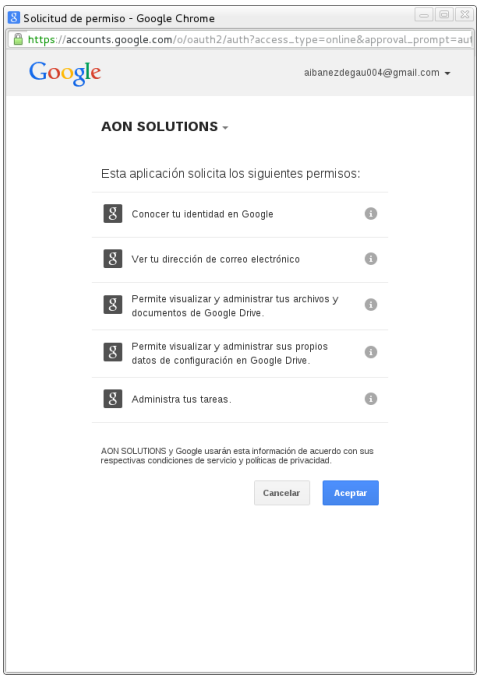
Imágen 4.1: Pantalla Login de AonSolutions.

Para realizar esta ejecución, y poder observar que la cuerta iteración del proyecto, es decir, el inicio de sesión mediante Google, funciona correctamente seleccionaremas dicha opción, pulsando en la imagen o icono de Google.

Una vez pulsado el botón, aparecerá otra ventana pidiendo la autencación (Usuario y contraseña de Google) de Google. Después de identificarse, el sistema de Google, pedirá al usuario los permisos pertinentes, en el caso de esta aplicación, pedirá el acceso a los datos básicos de la cuenta, los permisos para administrar Google Drive, y los permisos para administrar Goole Task. Se observar en las imágenes 4.2 y 4.3.



Imágen 4.2: Autenticación con Google



Imágen 4.3: Petición de permisos Google.

Cuando el usuario ya se ha dentificado y aceptado los permisos con su cuenta correspondiente de Google, accede a la página principal de AonSolutions, lo que significa que el sistema de logueo implementado en la primera iteración del proyecto, funciona de forma correcta.

4.1.2. Ejecución Interna

Una vez el usuario se ha autenticado con su correspondiente cuenta de Google, el sistema, internamente busca si el email de dicha cuenta está sincronizado con la cuenta de algún usuario de AonSolutions. En caso afirmativo, el sistema da acceso al usuario, con la cuenta de usuario correspondiente al email introducido. En caso contrario, deniega el acceso a la plataforma para dicho usuario.

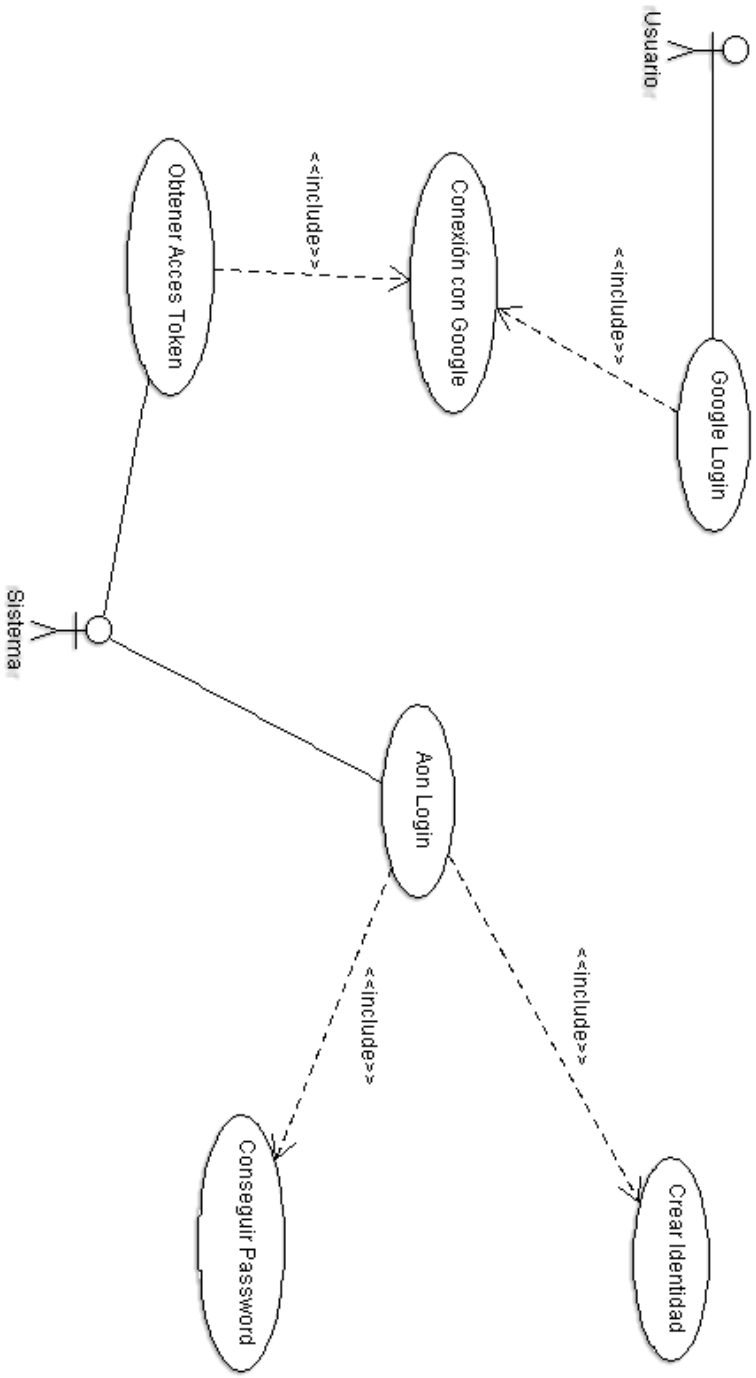
Cuando el el usuario si que está relacionado con una cuenta de Google, se crea la identidad del usuario, para poder acceder a su propia cuenta de AonSolutions, junto con los servicios que el usuario tenga disponible.

4.2. Captura de Requisitos

4.2.1. Requisitos

REFERENCIA	REQUISITOS
#1	Conectarse con Google.
#2	Obtener el token de acceso.
#3	Comprobar si el usuario se ha conectado por Google o ha utilizado el sistema de AonSolutions.
#4	Conseguir el nombre de usuario a partir del email.
#5	Crear Identidad.
#6	Conseguir el password del usuario.
#7	Login de AonSolutions

4.2.2. Modelo de Casos de Uso



Imágen 4.4: Modelo de casos de uso de la iteración 4.

4.2.3. Casos de Uso

CU: Google Login

Actores: Usuario

Descripción: El usuario pulsa el botón para loguearse con Google y el sistema le dará acceso a otra ventana donde se identificará con su propia cuenta de Google.

Precondiciones: --

Post-condiciones: --

Referencias: #1

Escenario Principal:
1. El usuario elige la opción de inicio de sesión mediante la cuenta de Google.
2. El sistema le redirige a otra ventana para que incie sesión en Google, y acepte los permisos de la aplicación.

CU: Obtener Acces Token

Actores: Sistema

Descripción: Una vez el usuario se haya identificado con su cuenta personal de Google, el sistema hace una petición con la intención de conseguir los token de acceso a las Google API's que lo necesiten, que son Google OAuth2 API, Google Drive API y Google Task API.

Precondiciones: El usuario ya está autenticado en Google.

Post-condiciones: --

Referencias: #1, #2

Escenario Principal:
1. El sistema hace la petición de los token de acceso a las Google API.
2. Google devuelve los Acces token que se han pedido.

CU: Aon Login

Actores: Sistema

Descripción: El sistema genera un password y le pasa el username ("OpenIdEmail=------@------.com") y el password generado al sistema de inicio de sesión de AonSolutions.

Precondiciones: Disponer de username y password.

Post-condiciones: El usuario ya esta conectado (en caso de que los datos introducidos sean correctos.)

Referencias: #5, #6, #7

Escenario Principal:

1. El sistema genera un password.
2. El sistema genera el username a partir del email.
3. El sistema se loguea con los datos del username y password.

CU: Crear Identidad

Actores: Sistema

Descripción: El sistema crea la identidad del usuario para poder a su cuenta correspondiente. Para ello primero comprueba si se trata de un inicio de sesión normal o se ha hecho a través de Google. Si lo hace a través de Google busca el username correspondiente en la base de datos por el email. Por último ejecuta createIdentity() ya definida en la plataforma.

Precondiciones: Disponer de un username.

Post-condiciones: La identidad se ha creado si los datos son correctos.

Referencias: #3, #4, #5

Escenario Principal:

1. El sistema comprueba si se ha iniciado sesión a través de Google o de la forma normal.

Si inicia sesión con Google

2. El sistema busca en la base de datos el username correspondiente al email con el que se ha identificado el usuario.
3. El sistema crea la identidad mediante la función ya disponible en la plataforma createIdentity ().

CU: Conseguir Password

Actores: Sistema

Descripción: Si se ha conectado mediante Google el sistema consigue el password generado anteriormente, de está manera cuando se verifiquen accedera a la plataforma. Si no ha inciado sesión con Google accede al password de la base de datos para verificarlo con el introducido por el usuario.

Precondiciones: --

Post-condiciones: --

Referencias: #3, #6

Escenario Principal:

1. El sistema comprueba si el usuario ha iniciado sesión con Google o de la forma normal.

Si lo ha hecho con Google:

2. El sistema busca el password generado anteriormente.

Si no

3. El sistema busca el password del usuario en la base de datos.

4.3. Análisis

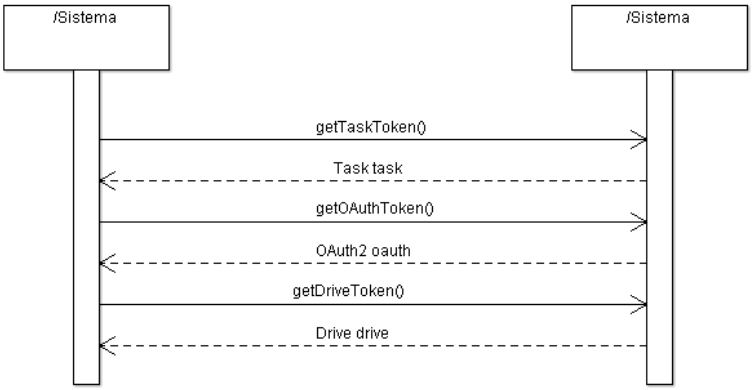
CU: Google Login

Diagrama de secuencia del sistema

Imágen 4.5: Diagrama de secuencia del sistema, Google Login.

CU: Obtener Acces Token

Diagrama de secuencia del sistema



Imágen 4.6: Diagrama de secuencia del sistema, Obtener Acces Token.

Contratos

Contrato 1:

- Nombre: getTaskToken()
- Responsabilidades: Devuelve el token de acceso para Google Task API
- Excepciones: --
- Precondiciones: --
- Post-condiciones: --
- Salida: Task task

Contrato 2:

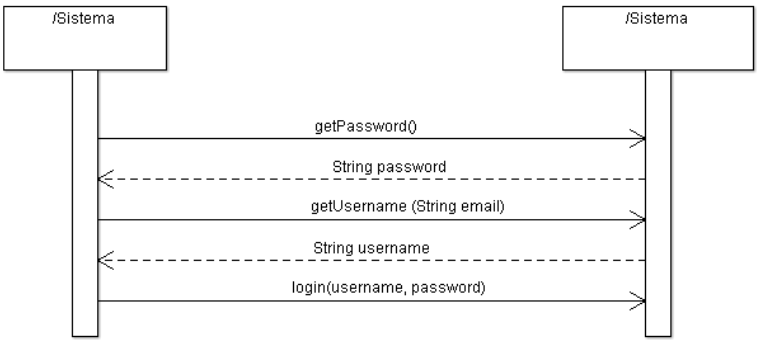
- Nombre: getOAuth2Token()
- Responsabilidades: Devuelve el token de acceso para Google OAuth2 API
- Excepciones: --
- Precondiciones: --
- Post-condiciones: --
- Salida: OAuth2 oauth

Contrato 3:

- Nombre: getDriveToken()
- Responsabilidades: Devuelve el token de acceso para Google Drive API
- Excepciones: --
- Precondiciones: --
- Post-condiciones: --
- Salida: Drive drive.

CU: Aon Login

Diagrama de secuencia del sistema



Imágen 4.7: Diagrama de secuencia del sistema, Aon Login.

Contratos

Contrato 1:

- Nombre: getPassword()
- Responsabilidades: Genera un password.
- Excepciones: --
- Precondiciones: --
- Post-condiciones: --
- Salida: String password

Contrato 2:

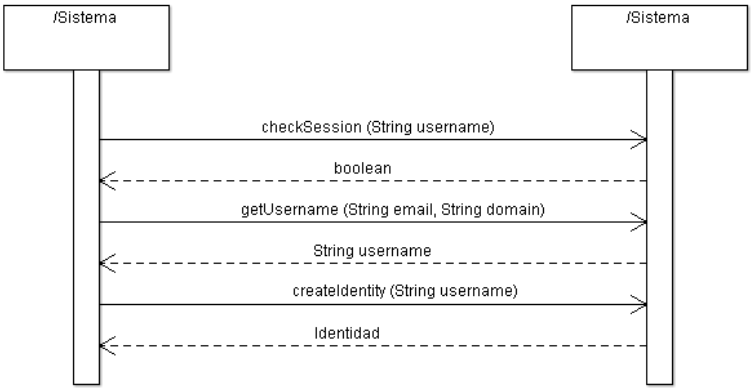
- Nombre: getUsername(String email)
- Responsabilidades: Genera el username a partir del email.
- Excepciones: --
- Precondiciones: Disponer de el email con el que se ha conectado a Google el usuario.
- Post-condiciones: --
- Salida: String username.

Contrato 3:

- Nombre: login(username, password)
- Responsabilidades: Inicia sesión en la plataforma de gestión empresarial AonSolutions.
- Excepciones: --
- Precondiciones: Disponer del username y del password.
- Post-condiciones: El usuario ya esta conectado en la plataforma AonSolutions.
- Salida: --

CU: Crear Identidad

Diagrama de secuencia del sistema



Imágen 4.8: Diagrama de secuencia del sistema, Crear identidad.

Contratos

Contrato 1:

- Nombre: checkSession(String username)
- Responsabilidades: Comprueba si el nombre del usuario es del tipo de username específico para cuendo se conectan con Google o no. (OpenIdEmail= -----@----.com)
- Excepciones: --
- Precondiciones: Disponer del username introducido.
- Post-condiciones: --
- Salida: boolean

Contrato 2:

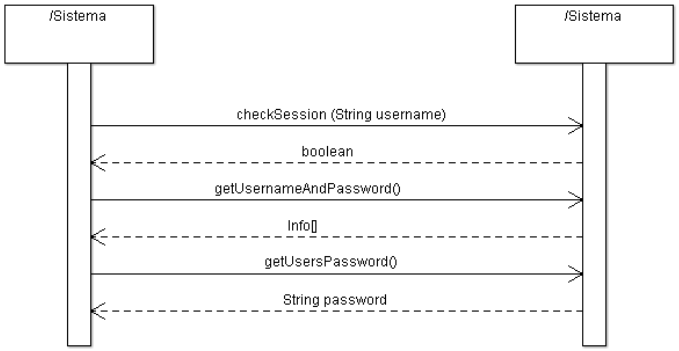
- Nombre: getUsername(String email, String domain)
- Responsabilidades: Busca en la base de datos el username correspondiente al correo electrónico introducido.
- Excepciones: --
- Precondiciones: Disponer del email y el dominio.
- Post-condiciones: --
- Salida: String username.

Contrato 3:

- Nombre: createIdentity(String username)
- Responsabilidades: Crea la identidad del usuario.
- Excepciones: --
- Precondiciones: Disponer del username de AonSolutions.
- Post-condiciones: --
- Salida: Identidad.

CU: Conseguir Password

Diagrama de secuencia del sistema



Imágen 4.9: Diagrama de secuencia del sistema, conseguir password.

Contratos

Contrato 1:

- Nombre: checkSession(String username)
- Responsabilidades: Comprueba si el nombre del usuario es del tipo de username específico para cuendo se conectan con Google o no. (OpenIdEmail= -----@----.com)
- Excepciones: --
- Precondiciones: Disponer del username introducido.
- Post-condiciones: --
- Salida: boolean

Contrato 2:

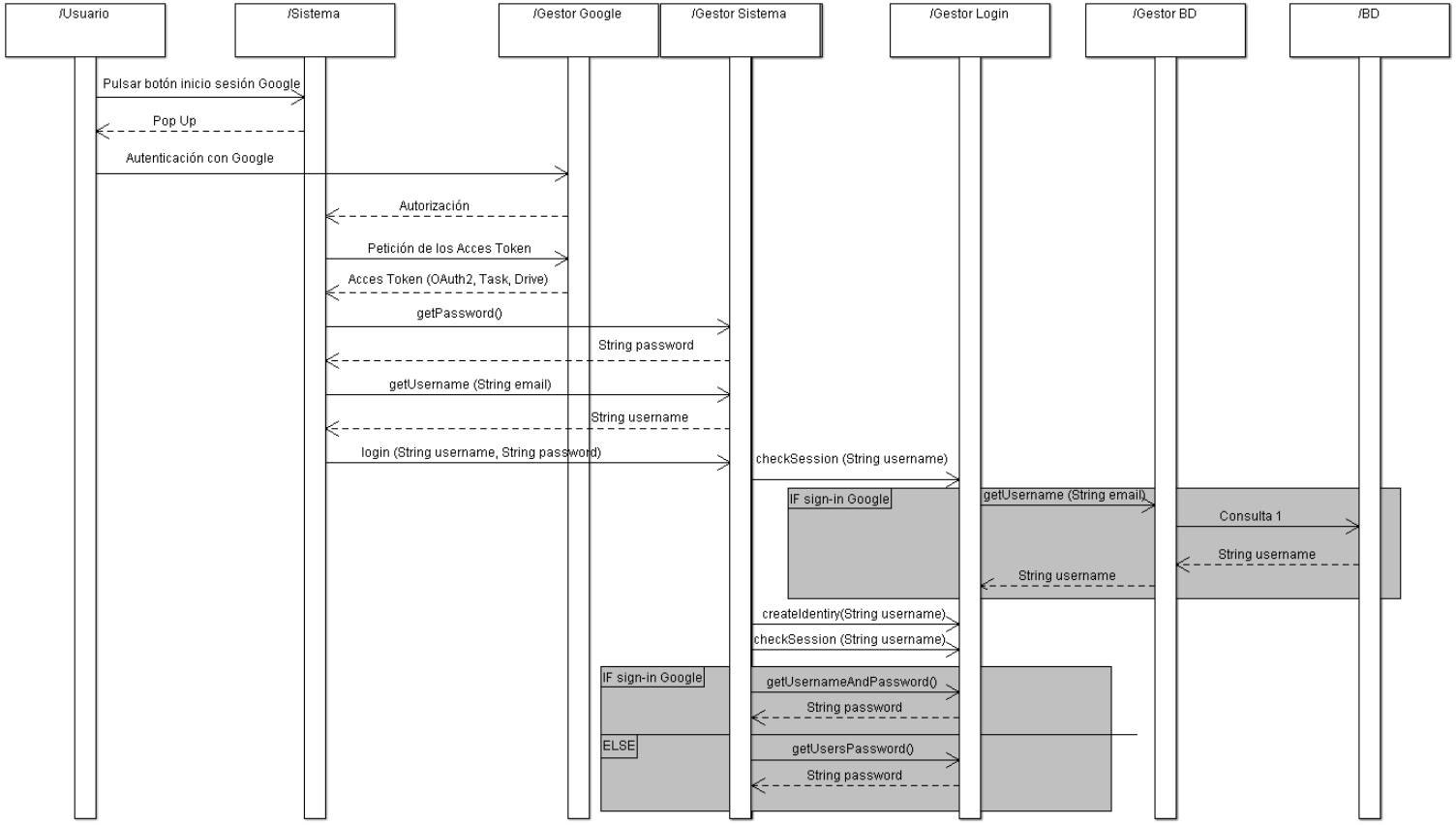
- Nombre: getUsernameAndPassword()
- Responsabilidades: Devuelve en un array el nombre de usuario y la contraseña introducida por el usuario.
- Excepciones: --
- Precondiciones: --
- Post-condiciones: --
- Salida: Info []

Contrato 3:

- Nombre: getUsersPassword()
- Responsabilidades: Devuelve la password correpondiente al usuario de la base de datos.
- Excepciones: --
- Precondiciones: --
- Post-condiciones: --
- Salida: String password

4.4. Diseño

Para reducir el número de diagramas de secuencia en este documento se ha decidido realizar el diagrama de secuencia de todos los casos de uso en uno.



Imágen 4.10: Diagrama de secuencia de la iteración 4.

Consulta 1

```
SELECT U.login

FROM  user AS U INNER JOIN mail_account AS MA ON U.id = MA.user_id

WHERE MA.email =  &EMAIL
```

Las fases de implementación, pruebas e implantación de ésta iteración se detallan en los apartados 8, 9 y 10 respectivamente. Donde se tratarán dentro de cada uno de los apartados, cada una de las fases de las cuatro iteraciones del proyecto.

5. ITERACIÓN 2: GOOGLE CALENDAR API

5.1. Introducción

El objetivo de ésta segunda iteración del proyecto, es realizar la sincronización de la agenda comercial de AonSolutions con Google Calendar.

Para ello se creará un Calendario en la cuenta de servicio de Google calendar por cada empresa que disponga de este servicio. Compartiendo el calendario con el email (cuenta de Google) de la empresa.

Cada uno de los eventos de la agenda comercial de AonSolutions, se sincronizarán con Google calendar, introduciéndolos en el calendario de la empresa correspondiente. El evento se compartirá con correspondiente comercial relacionado con dicho evento.

5.1.1. Ejecución Visual

Para añadir, borrar o modificar un evento de la agenda comercial de AonSolutions, se tendrá que accer a la pestaña de *comercial – Agenda Comercial*, y desde ese apartado se podrán tratar los eventos que se dispongan.

Para realizar la sincronización con Google Calendar, se ejecutan las funciones necesarias para ello, utilizando un listener a cada una de las tres operaciones nombradas en alterior párrafo.

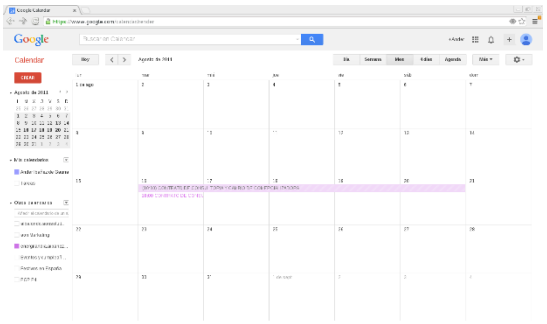
5.1.2. Ejecución Interna

Ésta es la parte donde la funcionalidad no se puede observar desde la interfaz gráfica de la plataforma de gestión empresarial AonSolutions. Por ello se han diferenciado dos apartados en esta parte de la documentación.

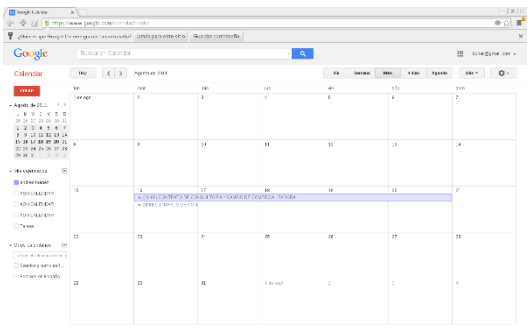
Se dispone de una función main, la cuál llama a las funciones necesarias para poder realizar la sincronización correctamente. Esta función main, será ejecutada cada cierto tiempo, normalmente en horario nocturno, para asegurarse de que el sistema sigue sincronizado de forma correcta. Pero lo importante de ésta parte es la primera ejecución, ya que después de esta primera ejecución se podría realizar un seguimiento sin necesidad de volver a ejecutarlo, aunque si que se vuelve ejecutar, eso si con el objetivo ya descrito del aseguramente de la sincronización.

Primero se va a realizar la comprobación de lo implementado en la primera iteración, ejecutando su correspondiente función main. Para realizar dicha comprobación se utilizarán dos cuentas de Google diferentes, uno que simulará ser el correo electrónico o una cuenta de la empresa, la cuál almacenará los calendarios o el calendario correspondiente a la empresa, y el otro correspondiente al correspondiente comercial realacionado con el evento en cuestión.

Después de realizar la sincronización, se puede observar tanto el calendario como los eventos en la cuenta de Google Calendar de la empresa, y el evento en la cuenta de Google Calendar del comercial, tal y como se puede ver en las dos imágenes 5.1 y 5.2.



Imágen 5.1: Google Calendar, empresa.



Imágen 5.2: Google Calendar, comercial.

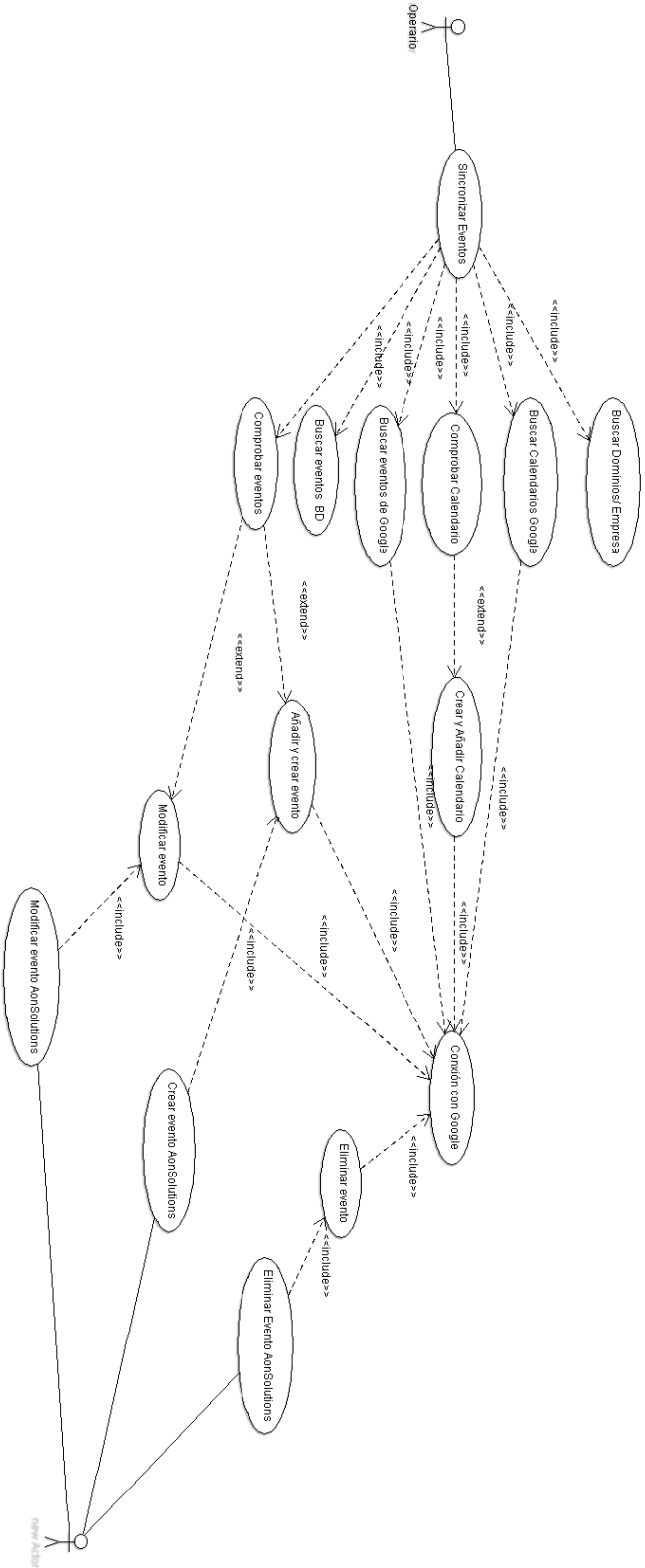
5.2. Captura de Requisitos

5.2.1. Requisitos

REFERENCIA	REQUISITOS
#1	Añadir calendario a Google Calendar
#2	Crear calendario de Google Calendar
#3	Modificar Calendario de Google Calendar
#4	Eliminar Calendario de Google Calendar
#5	Buscar todos los calendarios de Google Calendar
#6	Comprobar si la empresa (dominio) dispone de calendario
#7	Añadir evento a un calendario de Google Calendar
#8	Crear evento en un calendario de Google Calendar
#9	Modificar evento de un calendario de Google Calendar
#10	Eliminar evento de un calendario de Google Calendar.
#11	Buscar todos los eventos de un calendario de Google Calendar.
#12	Comprobar si un evento está en un calendario
#13	Sincronizar todos los eventos de la agenda comercial de AonSolutions en el calendario correspondiente de Google Calendar
#14	Buscar todos los eventos de la BD de cada Dominio/empresa.
#15	Buscar todos los dominios padre de la BD.
#16	Buscar todos los subdominios de un dominio padre.

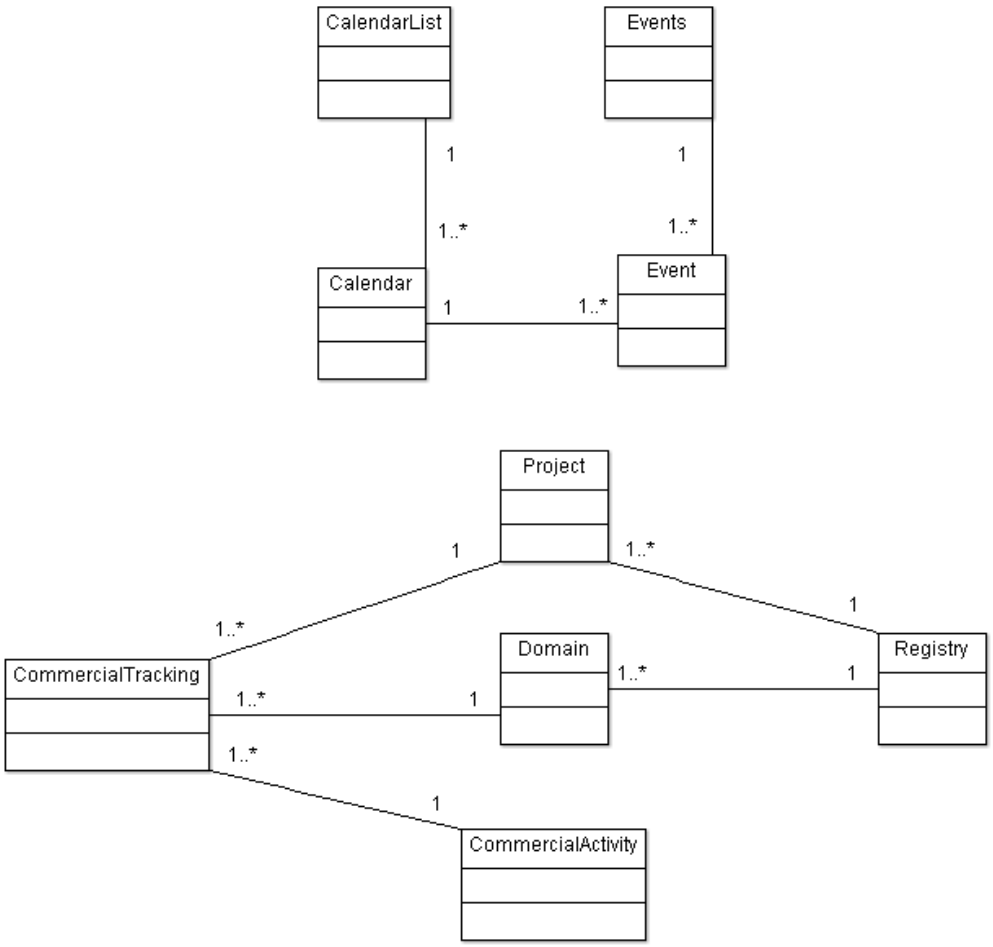
#17	Buscar datos de conexión de Google (Service Account)
#18	Buscar email de una empresa.
#19	Buscar email de un comercial.
#20	Buscar proyecto al que pertenece el evento.
#21	Buscar actividad del evento.
#22	Buscar cliente potencial del evento.
#23	Conectarse a Google (OAuth 2).

5.2.2. Modelo de Casos de uso



Imágen 5.3: Modelo de casos de uso iteración 1.

5.2.3. Modelo de Dominio



Imágen 5.4: Modelo de dominio de la iteración 1.

En la imagen 5.4 se puede observar el modelo de dominio de la segunda iteración, en la parte superior se encuentran las clases u objetos pertenecientes a Google Calendar API y en la parte inferior, las clases correspondientes a AonSolutions.

No se especifican ni los atributos ni las funciones de cada clase en el modelo de dominio de la iteración 1, para que no ocupe tanto espacio en la imagen y el diagrama se vea de manera más clara y sencilla. A continuación se especificarán tanto los atributos como las funciones de cada clase / objeto, únicamente las que se han utilizado para el desarrollo de la iteración, para no extenderse.

Calendar

Corresponde a un calendario de Google Calendar. En un calendario se almacenarán los eventos del tipo Event.

Atributos:

String **description**: Descripción del calendario.

String **id**: Identificador del calendario.

String **kind**: Tipo del recurso (“calendar#calendar”).

String **location**: Localización geográfica de la agenda como texto de formato libre.

String **summary**: Título del calendario.

String **timeZone**: Zona horaria del calendario.

Funciones:

getDescription*():* Devuelve descripción del calendario.

getId*():* Devuelve dentificador del calendario.

getKind*():* Devuelve el tipo del recurso (“calendar#calendar”).

getLocation*():* Devuelve la ubicación geográfica de la agenda como texto de formato libre.

getSummary*():* Devuelve el título del calendario.

getTimeZone*():* Devuelve la zona horaria del calendario.

set*(String fieldName, Object value)*

setDescription*(String description):* Inserta el valor de la descripción del calendario.

setId*(String id):* Inserta el valor del identificador del calendario.

setKind*(String kind):* Inserta el valor del tipo de recurso (“calendar#calendar”).

setLocation*(String location):* Inserta el valor de la ubicación geográfica de la agenda como texto de formato libre.

setSummary*(String summary):* Inserta el valor del título del calendario.

setTimeZone*(String timeZone):* Inserta el valor de la zona horaria del calendario.

CalendarList

Corresponde a una lista de calendarios de Google Calendar.

Atributos:

List<CalendarListEntry> **items**: Lista de las entradas a los calendarios.

String **kind**: Tipo de la colección (“calendar#calendarList”).

Funciones:

getItems*():* Devuelve la lista con las entradas a los calendarios, para poder acceder a ellos.

getKind*():* Devuelve el tipo de la colección (“calendar#calendarList”).

***setItems**(List<CalendarListEntry> items):* Inserta la lista de las entradas a los calendarios.

***setKind**(String kind):* Inserta el tipo de la colección (“calendar#calendarList”).

Event

Corresponde a eventos de Google Calendar, almacenados en un calendario.

Atributos:

List<EventAttendee> **attendees**: Lista de los asistentes del evento.

DateTime **created**: Fecha/hora de creación del evento.

String **description**: Descripcion del evento.

EventDateTime **end**: Fecha/hora de finalización del evento.

Event.ExtendedProperties **extendedProperties**: Propiedades adicionales del evento.

String **id**: Identificador del evento.

String **kind**: Tipo del recurso (“calendar#event”).

String **location**: Ubicación geográfica del evento como texto de formato libre.

EventDateTime **start**: Fecha/hora de inicio del evento.

String **status**: Estado del evento.

String **summary**: Título del evento.

DateTime **updated**: Fecha/hora en el que el evento se ha modificado por última vez.

Funciones:

***getAttendees**():* Devuelve los asistentes al evento.

***getCreated**():* Devuelve la fecha/hora de creación del evento.

***getDescription**():* Devuelve la descripción del evento.

***getEnd**():* Devuelve la fecha/hora de finalización del evento.

***getExtendedProperties**():* Devuelve las Propiedades adicionales del evento.

***getId**():* Devuelve el identificador del evento.

***getKind**():* Devuelve el tipo del recurso (“calendar#event”).

***getLocation**():* Devuelve la localización geográfica del evento en formato de texto libre.

getStart*():* Devuelve la fecha/hora de inicio del evento.

getStatus*():* Devuelve el estado del evento.

getSummary*():* Devuelve el título del evento.

getUpdated*():* Devuelve la fecha/hora en el que el evento se ha modificado por última vez.

set*(String fieldName,Object value).*

setAttendees*(List<EventAttendee> attendees)*: Inserta la lista de asistentes al evento.

setCreated*(DateTime created):* Inserta la fecha/hora de creación del evento.

setDescription*(String description):* Inserta la descripción del evento.

setEnd*(EventDateTime):* Inserta la fecha/hora de finalización del evento.

setExtendedProperties*(Event.ExtendedProperties extendedProperties):* Inserta las propiedades adicionales del evento.

setId*(String id):* Inserta el identificador del evento.

setKind*(String kind):* Inserta el tipo del recurso (“calendar#event”).

setLocation*(String location):* Inserta la ubicación geográfica del evento.

setStart*(EventDateTime start):* Inserta la fecha/hora de comienzo del evento.

setStatus*(String status):* Inserta el estado del evento.

setSummary*(String summary):* Inserta el título del evento.

setUpdated*(DateTime updated):* Inserta la fecha/hora en la que el evento ha sido modificado por última vez.

Events

Corresponde a una lista de eventos de Google Calendar.

Atributos:

List<Event> **items**: Lista de los eventos de un calendario.

String **kind**: Tipo de la colección (“calendar#events”).

Funciones:

getItems*():* Devuelve la lista con los eventos de un calendario, para poder acceder a ellos.

getKind*():* Devuelve el tipo de la colección (“calendar#events”).

setItems*(List<Event> items):* Inserta la lista de los eventos de un calendario.

setKind*(String kind):* Inserta el tipo de la colección (“calendar#events”).

CommercialTracking

Corresponde a los eventos de la agenda comercial de AonSolutions.

Atributos:

Integer **id**: Identificador del evento commercial tracking.

Timestamp **date**: La fecha y hora en la que comienza el evento de tipo commercial tracking.

String **comments**: Comentario del evento de tipo commercial tracking.

Timestamp **endDate**: La fecha y hora en la que finaliza el evento de tipo commercial tracking.

Boolean **allDay**: Si el evento se lleva a cabo durante todo el dia.

Funciones:

getId(): Devuelve el identificador del evento de tipo commercial tracking.

getDate(): Devuelve la fecha y hora en la que comienza el evento de tipo commercial tracking.

getComments(): Devuelve el comentario del evento de tipo commercial tracking.

getEndDate(): Devuelve la fecha y hora en la que finaliza el evento de tipo commercial tracking.

getAllDay(): Devuelve cierto sii el evento se lleva a cabo durante todo el dia.

setId(Integer id): Inserta el valor del identificador del evento de tipo commercial tracking.

setDate(Timestamp date): Inserta el valor de la fecha y hora en la que comienza el evento de tipo commercial tracking.

setComments(String comments): Inserta el valor del comentario del evento de tipo commercial tracking.

setEndDate(Timestamp endDate): Inserta el valor de la fecha y hora en la que finaliza el evento de tipo commercial tracking.

setAllDay(boolean allDay): Inserta el valor de del boolean que es cierto si el evento se lleva a cabo durante todo el dia.

CommercialActivity

Corresponde a la actividad comercial de un evento de la agenda comercial de AonSolutions.

Atributos:

Integer **id**: Identificador de la actividad del evento de tipo commercial tracking.

String **name**: Nombre de la actividad del evento de tipo commercial tracking.

Funciones:

getId(): Devuelve el identificador de la actividad del evento de tipo commercial tracking.

getName(): Devuelve el nombre de la actividad del evento de tipo commercial tracking.

setId(Integer id): Inserta el valor del identificador de la actividad del evento de tipo commercial tracking.

setName(String name): Inserta el valor del nombre de la actividad del evento de tipo commercial tracking.

Project

Corresponde al Proyecto al cuál pertenece el evento de la agenda comercial de AonSolutions.

Atributos:

Integer **id:** Identificador del proyecto.

String **name**: Nombre del proyecto.

String **alias**: Alias del proyecto.

Date **date**: Fecha del proyecto.

Funciones:

getId(): Devuelve el identificador del proyecto.

getName(): Devuelve el nombre del proyecto.

getAlias(): Devuelve el alias del proyecto.

getDate(): Devuelve la fecha del proyecto.

setId(Integer id): Inserta el valor del identificador del proyecto.

setName(String name): Inserta el valor del nombre del proyecto.

setAlias(String alias): Inserta el valor del alias del proyecto.

setDate(Date date): Inserta el valor de la fecha del proyecto.

Domain

Corresponde al dominio de la empresa a la cuál pertenece la agenda comercial de AonSolutions.

Atributos:

Integer **id**: Identificador del dominio.

String **name**: Nombre del dominio

String **description**: Descripción del dominio.

Funciones:

getId(): Devuelve el identificador del dominio.

getName(): Devuelve el nombre del dominio.

getDescription(): Devuelve la descripción del dominio.

setId(Integer id): Inserta el valor del identificador del dominio.

setName(String name): Inserta el valor del nombre del dominio.

setDescription(String description): Inserta el valor de la descripción del dominio.

Registry

Corresponde al registro de usuarios y empresas de la plataforma de gestión empresarial AonSolutions.

Atributos:

Integer **id**: Identificador de registry.

String **name**: Nombre de registry.

Funciones:

getId(): Devuelve el identificador de registry.

getName(): Devuelve el nombre de registry.

setId(Integer id): Inserta el valor del identificador de registry.

setName(String name): Inserta el valor del nombre de registry.

5.2.4. Casos de Uso

CU: Buscar dominios

Actores: Sistema

Descripción: El sistema consulta en la base de datos todos los dominios que dispone AonSolutions, es decir, las empresas que tienen un servicio contratado con AonSolutions.

Precondiciones: --

Post-condiciones: --

Referencias: #15, #22

Escenario Principal:
1. Sistema: Busca todos los dominios padre de la base de datos.
2. Sistema: Busca todos los subdominios para cada dominio padre.

CU: Buscar calendario

Actores: Sistema

Descripción: El sistema busca todos los calendarios de una cuenta de Google en Google Calendar.

Precondiciones: --

Post-condiciones: --

Referencias: #5, #23

Escenario Principal:
1. Sistema: Hace la conexión con Google.
2. Sistema: Buscar todos los calendarios de la cuenta Google Calendar.

CU: Comprobar calendarios

Actores: Sistema

Descripción: El sistema comprueba si un dominio dispone de calendario en Google Calendar en la cuenta de servicio.

Precondiciones: Disponer una lista de calendarios y el nombre del dominio a analizar.

Post-condiciones: --

Referencias: #6

Escenario Principal:
1. Ordena la lista de calendarios.
2. Comprueba si un dominio / empresa dispone de calendario.

CU: Crear y añadir calendario

Actores: Sistema

Descripción: El sistema se conecta con Google, crea un calendario para la empresa o dominio correspondiente en Google Calendar y lo añade a la cuenta.

Precondiciones: Disponer del dominio de la empresa.

Post-condiciones: El calendario correspondiente a una empresa ya esta creado en Google Calendar.

Referencias: #1, #2, #23

Escenario Principal:
1. Sistema: Crea un calendario de Google Calendar.
2. Sistema: Se conecta con Google.
3. Sistema: Añade el calendario creado a la cuenta de servicio correspondiente.

CU: Modificar calendario

Actores: Sistema

Descripción: El sistema se conecta con Google, crea un calendario para la empresa o dominio correspondiente en Google Calendar y lo actualiza a la cuenta.

Precondiciones: Disponer del dominio de la empresa y el identificar del calendario de Google Calendar que hay que actualizar.

Post-condiciones: El calendario correspondiente a una empresa ya está actualizado en Google Calendar.

Referencias: #1, #3, #23

Escenario Principal:
4. Sistema: Crea un calendario de Google Calendar.
5. Sistema: Se conecta con Google.
6. Sistema: Modifica el calendario creado a la cuenta de servicio correspondiente.

CU: Eliminar calendario

Actores: Sistema

Descripción: El sistema se conecta a Google y elimina el calendario correspondiente.

Precondiciones: Se dispone de un calendario de Google Calendar (el identificador).

Post-condiciones: --

Referencias: #11, #23

Escenario Principal:
1. El sistema se conecta a Google.
2. El sistema elimina el calendario, cuyo identificador es el dado.

CU: Buscar eventos Google

Actores: Sistema

Descripción: El sistema se conecta a Google y busca todos los eventos de un calendario de Google Calendar.

Precondiciones: Se dispone de un calendario de Google Calendar (del identificador).

Post-condiciones: --

Referencias: #11, #23

Escenario Principal:

3. El sistema busca todos los eventos de un calendario

CU: Buscar eventos BD

Actores: Sistema

Descripción: El sistema consulta en la base de datos todos los eventos (comercial Tracking) de un dominio de la agenda comercial de AonSolutions.

Precondiciones: Disponer del dominio de la empresa.

Post-condiciones: --

Referencias: #14

Escenario Principal:

1. El sistema busca todos los eventos de la agenda comercial.

CU: Comprobar eventos

Actores: Sistema

Descripción: El sistema comprueba si un evento se encuentra en su correspondiente calendario en Google Calendar.

Precondiciones: Lista de eventos de un calendario de Google y un evento Commercial Tracking.

Post-condiciones: --

Referencias: #12

Escenario Principal:

1. El sistema comprueba si un evento está en la lista de eventos.

CU: Añadir y Crear evento

Actores: Cliente, Sistema

Descripción: El sistema se conecta con Google, crea un evento de Google Calendar y lo añade al calendario correspondiente en Google Calendar.

Precondiciones: Un Evento DE AonSolutions (CommercialTracking), y un calendario.

Post-condiciones: --

Referencias: --

Escenario Principal:
1. El sistema crea un evento de Google Calendar.
2. El sistema se conecta
3. El sistema añade el evento a un calendario de Google Calendar.

CU: Modificar Evento

Actores: Sistema

Descripción: El sistema se conecta con Google, crea un evento de Google y modifica un evento del calendario correspondiente en Google Calendar.

Precondiciones: Disponer de un evento (Commercial Tracking) y del dominio de la empresa correspondiente

Post-condiciones: Modifica el evento.

Referencias: #9, #23

Escenario Principal:
1. El sistema crea un evento.
2. El sistema se conecta con Google
3. El sistema modifica el evento de Google Calendar.

CU: Eliminar Evento

Actores: Sistema

Descripción: El sistema elimina el evento del calendario de Google Calendar.

Precondiciones: Disponer del evento a eliminar y el identificador del calendario al que pertnece.

Post-condiciones: El evento ya no se encuentra en el calendario.

Referencias: #10, #23

Escenario Principal:
1. El sistema elimina el evento del calendario

CU: Sincronizar

Actores: Sistema

Descripción: El sistema ejecuta la función s sincronizar, para sincronizar la agenda comercial de AonSolutions del usuario con su cuenta de Google Calendar. Para ello el sistema tiene que realizar determinadas funciones que corresponde a la mayoría de los casos de uso detallados con anterioridad. *Buscar Dominios BD, Buscar calendarios BD, Buscar calendarios Google, Comprobar calendario, Buscar eventos Google, Comprobar eventos.*

Precondiciones: --

Post-condiciones: Todas los eventos y calendarios correspondientes ya están sincronizados con la cuenta de servicio de Google Calendar.

Referencias: #1, #2, #5, #6, #7, #8, #9, #11, #12, #13, #14, #15, #17, #18, #19, #20, #21, #22, #23

Escenario Principal:

1. El sistema busca todos los dominios padre de la base de datos.

Para todos los dominios

2. El sistema busca todos los calendarios que la cuenta de servicio tenga en Google Calendar.
3. El sistema comprueba si el calendario se encuentra o no en la acuenta de Google Calendar.
    a. El sistema crea y añade evento a Google Calendar.
4. El sistema busca todos los eventos de la base de datos.
5. El Sistema busca todos los eventos que disponga una determinada cuenta en Google Calendar.

Para todos los eventos

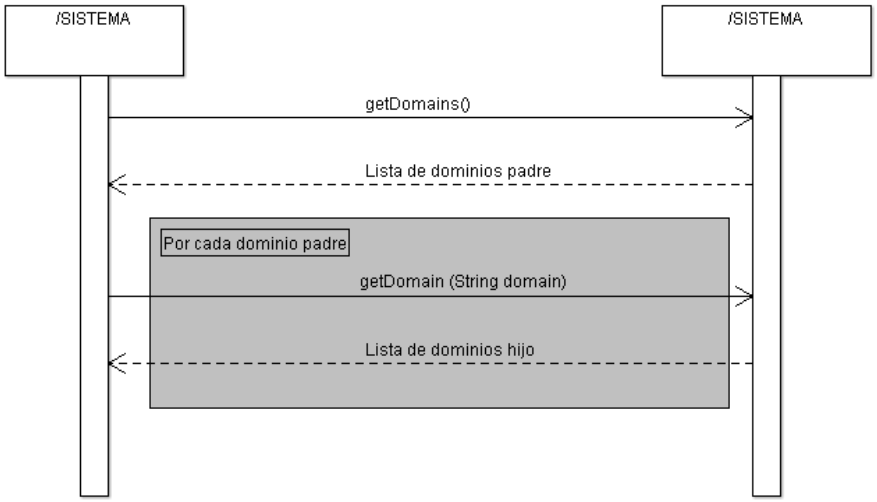
6. El sistema comprueba si un evento se encuentra en un claendario de Google Calendar.

Los casos de uso *Crear evento AonSolutions*, *Modificar evento AonSolutions* y *Eliminar evento AonSolutions*, no se detallan debido a que ya formaban parte de la plataforma, a los cuales se les añade un listener para ejecutarse los casos de uso *Añadir y crear evento, Modificar Evento* y *Eliminar evento* respectivamente.

5.3. Análisis

CU: Buscar dominios

Diagrama de secuencia del sistema



Imágen 5.5: Diagrama de secuencia del sistema del caso de uso buscar dominios

Contratos

Contrato 1:

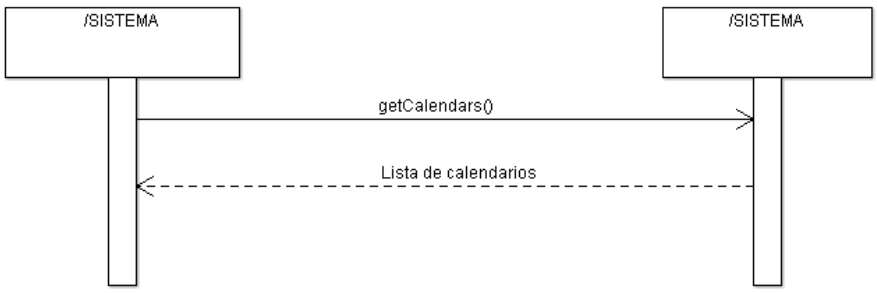
- Nombre: getDomains()
- Responsabilidades: Devuelve una lista con todos los dominios padre de la base de datos.
- Excepciones: --
- Precondiciones: --
- Post-condiciones: --
- Salida: Lista de dominios padre.

Contrato 2:

- Nombre: getDomain(String domain)
- Responsabilidades: Devuelve una lista con todos los subdominios del dominio *domain*.
- Excepciones: --
- Precondiciones: Disponer de un dominio para hacer la conexión con la Base de datos.
- Post-condiciones: --
- Salida: Lista de dominios hijo (subdominio).

CU: Buscar calendario

Diagrama de secuencia del sistema



Imágen 5.6: Diagrama de secuencia del sistema del caso de uso buscar calendarios

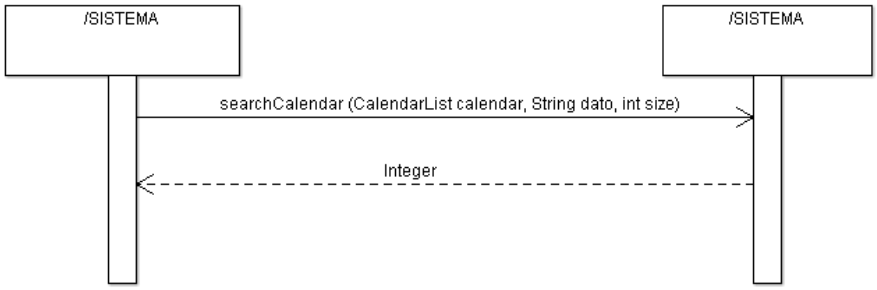
Contratos

Contrato 1

- Nombre: getCalendars
- Responsabilidades: Conectarse a Google y devolver todos los calendarios de Google Calendar.
- Excepciones: --
- Precondiciones: --
- Post-condiciones: --
- Salida: Lista de calendarios.

CU: Comprobar calendarios

Diagrama de secuencia del sistema



Imágen 5.7: Diagrama de secuencia del sistema del caso de uso comprobar calendarios

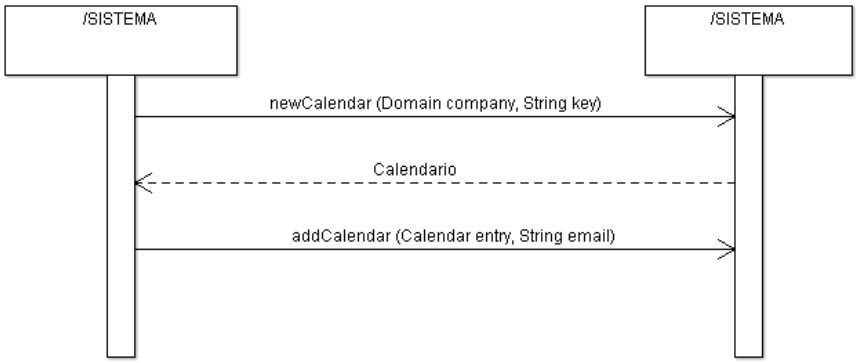
Contratos

Contrato 1

- Nombre: searchCalendar(CalendarList calendar,String dato,int size)
- Responsabilidades: Ordena la lista de calendarios y Devuelve un Integer con la posición del calendario en la lista o “-1” si el calendario no está en la lista de calendarios.
- Excepciones: --
- Precondiciones: Disponer de una lista de calendarios, el dato a buscar en la lista y el tamaño de la lista.
- Post-condiciones: --
- Salida: Integer

CU: Crear y añadir calendario

Diagrama de secuencia del sistema



Imágen 5.8: Diagrama de secuencia del sistema del caso de uso crear y añadir calendario

Contratos

Contrato 1

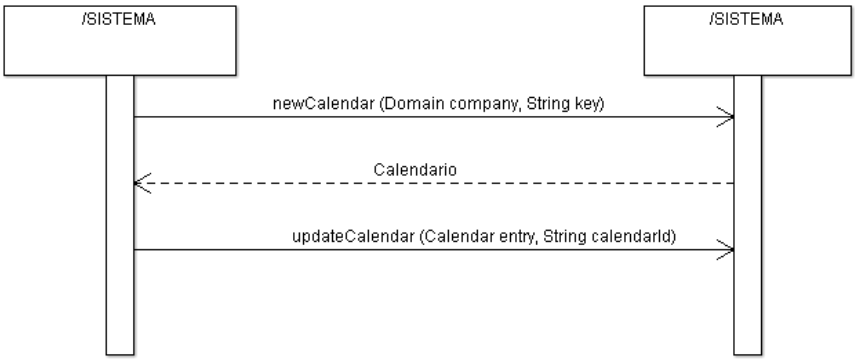
- Nombre: newCalendar(Domain company,String key)
- Responsabilidades: Obtener la información del dominio de la empresa, crear un nuevo calendario y lo devuelve.
- Excepciones: --
- Precondiciones: Disponer del dominio de la empresa.
- Post-condiciones: --
- Salida: Calendario

Contrato 2

- Nombre: addCalendar(Calendar entry, String email)
- Responsabilidades: Conectarse a Google y añadir un calendario a Google Calendar con la cuenta de servicio de la empresa, y compartir el calendario con el email de la empresa.
- Excepciones: --
- Precondiciones: Disponer del calendario y del email de la empresa.
- Post-condiciones: El calendario se encuentra en Google Calendar
- Salida: --

CU: Modificar calendario

Diagrama de secuencia del sistema



Imágen 5.9: Diagrama de secuencia del sistema, modificar calendario.

Contratos

Contrato 1

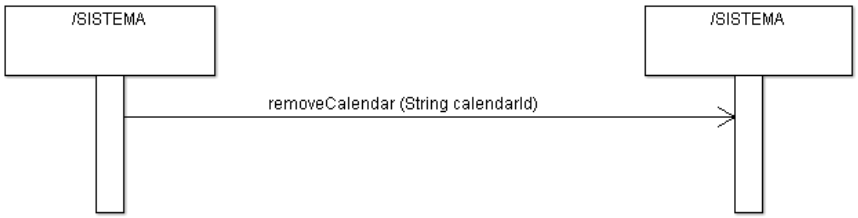
- Nombre: newCalendar(Domain company,String key)
- Responsabilidades: Obtener la información del dominio de la empresa, crear un nuevo calendario y lo devuelve.
- Excepciones: --
- Precondiciones: Disponer del dominio de la empresa.
- Post-condiciones: --
- Salida: Calendario

Contrato 2

- Nombre: updateCalendar(Calendar entry, String calendarId)
- Responsabilidades: Conectarse a Google y actualizar el calendario a Google Calendar con la cuenta de servicio de la empresa, y compartir el calendario con el email de la empresa.
- Excepciones: --
- Precondiciones: Disponer del calendario y del identificador del calendario a actualizar.
- Post-condiciones: El calendario se encuentra en Google Calendar
- Salida: --

CU: Eliminar calendario

Diagrama de secuencia del sistema



Imágen 5.10: Diagrama de secuencia del sistema, Eliminar calendario.

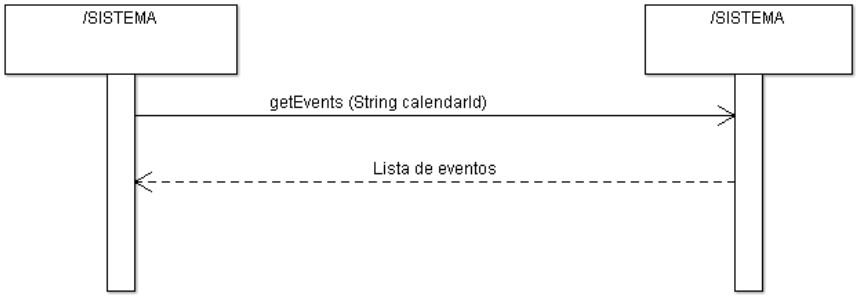
Contratos

Contrato 1

- Nombre: removeCalendar(String calendarId)
- Responsabilidades: Conectarse con Google y eliminar el calendario dado.
- Excepciones: --
- Precondiciones: Identificador del calendario
- Post-condiciones: El calendario correspondiente ya no existe en Google calendar.
- Salida: --

CU: Buscar eventos Google

Diagrama de secuencia del sistema



Imágen 5.11: Diagrama de secuencia del sistema del caso de uso buscar eventos Google

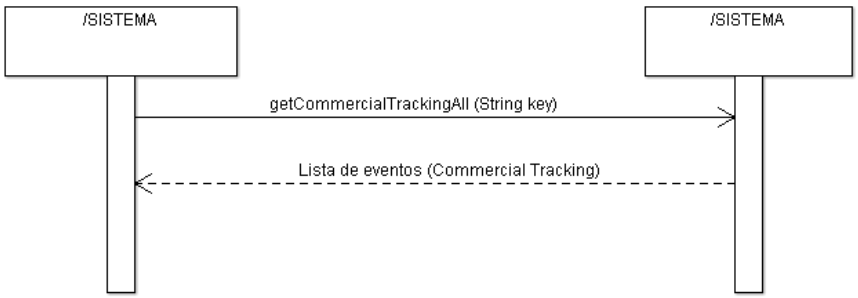
Contratos

Contrato 1

- Nombre: getEvents(String calendarId)
- Responsabilidades: Conectarse con Google y buscar todos los eventos del calendario dado.
- Excepciones: --
- Precondiciones: Identificador del calendario
- Post-condiciones: --
- Salida: Lista de eventos

CU: Buscar eventos BD

Diagrama de secuencia del sistema



Imágen 5.12: Diagrama de secuencia del sistema del caso de uso buscar eventos BD

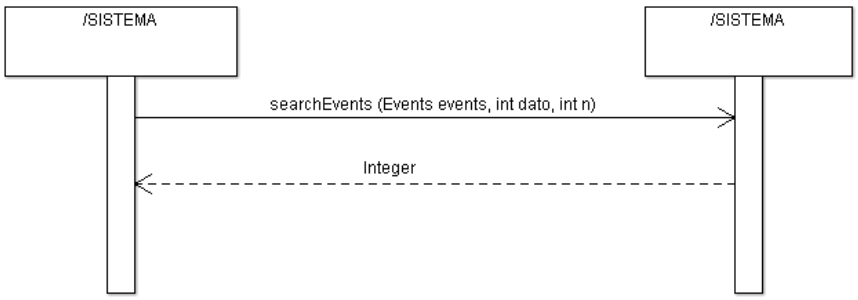
Contratos

Contrato 1

- Nombre: getCommercialTrackingAll(String key)
- Responsabilidades: Buscar todos los eventos (Commercial Tracking) de la agenda comercial.
- Excepciones: --
- Precondiciones: Dominio de la empresa
- Post-condiciones: --
- Salida: Lista de eventos (Commercial Tracking)

CU: Comprobar eventos

Diagrama de secuencia del sistema



Imágen 5.13: Diagrama de secuencia del sistema del caso de uso comprobar eventos

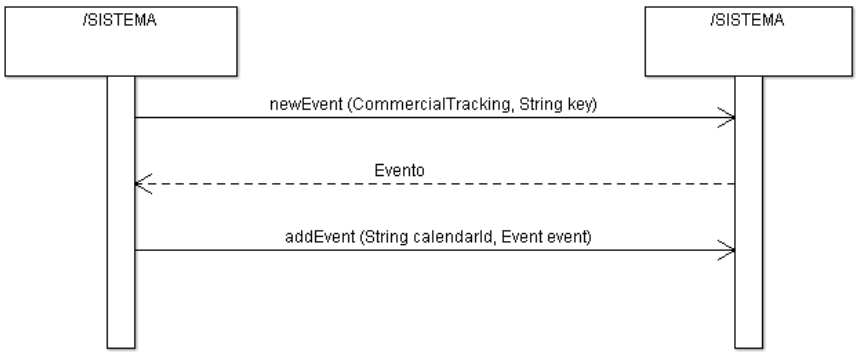
Contratos

Contrato 1

- Nombre:searchEvents(Events events,int dato,int n)
- Responsabilidades: Devuelve un integer con la posición del evento en la lista de eventos o “-1” si no está en la lista.
- Excepciones: --
- Precondiciones: --
- Post-condiciones: --
- Salida: Integer

CU: Añadir y Crear evento

Diagrama de secuencia del sistema



Imágen 5.14: Diagrama de secuencia del sistema del caso de uso crear y añadir evento

Contratos

Contrato 1

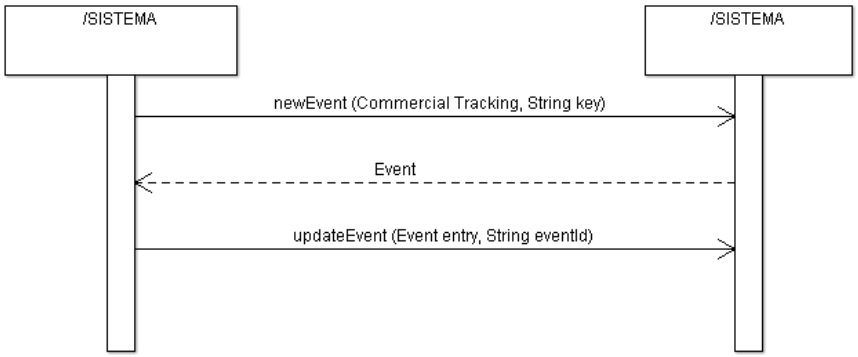
- Nombre: newEvent(CommercialTrackin commercialTracking,String key)
- Responsabilidades: Crea un evento de Google Calendar.
- Excepciones: --
- Precondiciones: Disponer de un evento (Commercial Tracking) y del dominio de la empresa.
- Post-condiciones: --
- Salida: Evento de Google Calendar

Contrato 2

- Nombre: addEvent(String calendarId, Event event)
- Responsabilidades: Se conecta con Google y añade el evento a su calendario correspondiente en Google Calendar.
- Excepciones: --
- Precondiciones: Disponer de un identificador del calendario y un evento a añadir.
- Post-condiciones: El evento ya se encuentra en su calendario correspondiente.
- Salida: --

CU: Modificar Evento

Diagrama de secuencia del sistema



Imágen 5.15: Diagrama de secuencia del sistema, Modificar evento

Contratos

Contrato 1

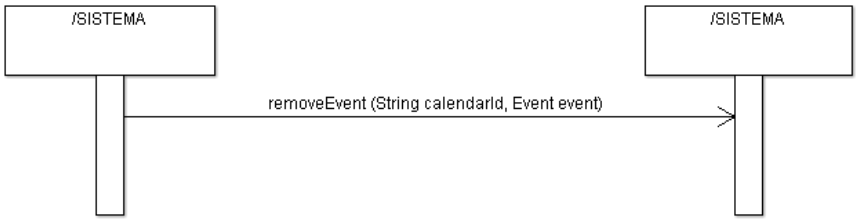
- Nombre: newEvent(CommercialTrackin commercialTracking,String key)
- Responsabilidades: Crea un evento de Google Calendar.
- Excepciones: --
- Precondiciones: Disponer de un evento (Commercial Tracking) y del dominio de la empresa.
- Post-condiciones: --
- Salida: Evento de Google Calendar

Contrato 2

- Nombre:updateEvent(Event entry, String eventId)
- Responsabilidades: Actualiza el evento correspondiente a eventId, en la cuenta de servicio de Google.
- Excepciones: --
- Precondiciones: Disponer de un evento de Google.
- Post-condiciones: Google Calendar ya tiene el evento correspondiente actualizado de forma correcta.
- Salida: --

CU: Eliminar Evento

Diagrama de secuencia del sistema



Imágen 5.16: Diagrama de secuencia del sistema del caso de uso eliminar evento

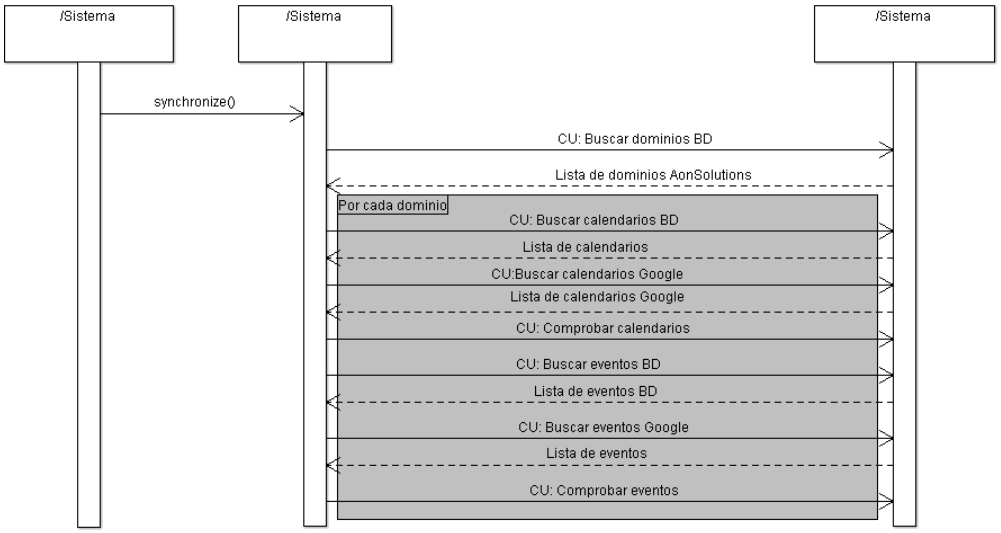
Contratos

Contrato 1

- Nombre: removeEvent (String CalendarId, Event event).
- Responsabilidades: El sistema eliminar el evento del calendario
- Excepciones: --
- Precondiciones: Evento a eliminar e identificador del calendario.
- Post-condiciones: El evento ya no está en el calendario.
- Salida: --

CU: Sincronizar

Diagrama de secuencia del sistema



Imágen 5.17: Diagrama de secuencia del sistema, Sincronizar.

Contratos

Contrato 1:

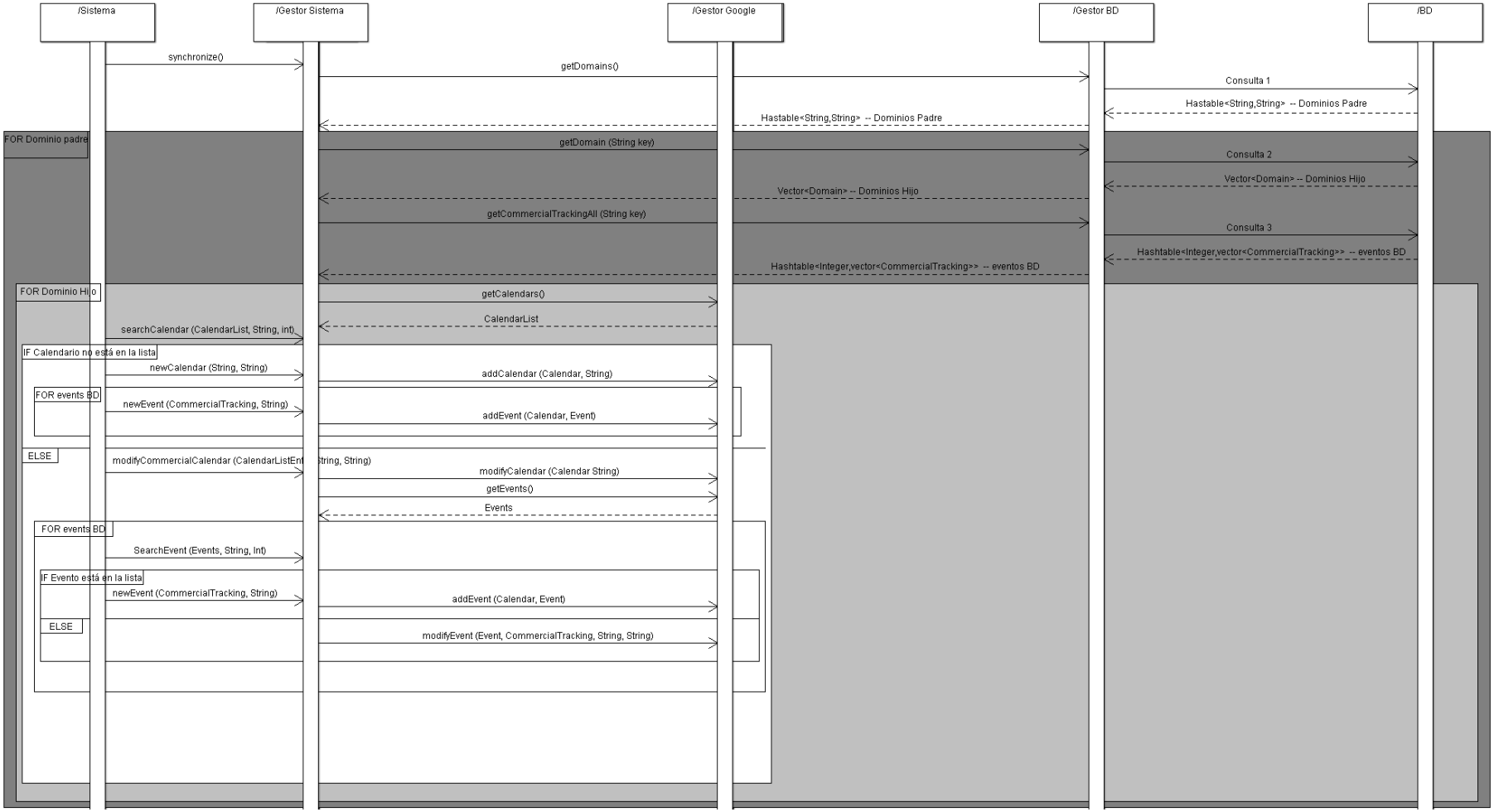
- Nombre: synchronize()
- Responsabilidades: El sistema sincroniza todos los eventos de la agenda comercial de AonSolutions con Google Calendar.
- Excepciones: --
- Precondiciones: --
- Post-condiciones: Todas los eventos de la agenda comercial de AonSolutions ya están sincronizadas con Google Calendar.
- Salida: --

5.4. Diseño

Para reducir el número de diagramas de secuencia en este documento se ha decidido realizar el diagrama de secuencia de la mayoría de los casos de uso en uno, precisamente el del caso de uso sincronizar, ya que abarca la gran mayoría de los casos de uso de ésta iteración.

Por otra parte se adjuntará el diagrama de secuencia de otros 6 casos de uso, Crear y Añadir calendario, Crear y Añadir evento, Modificar evento, Modificar calendario, Eliminar calendario y Eliminar evento, respectivamente.

CU: Sincronizar



Imágen 5.18: Diagrama de secuencia , sincronizar

Consulta 1:

```
ConnectionInfo connectionInfo = ConnectionInfo.getDefaultConnectionInfo ();
```

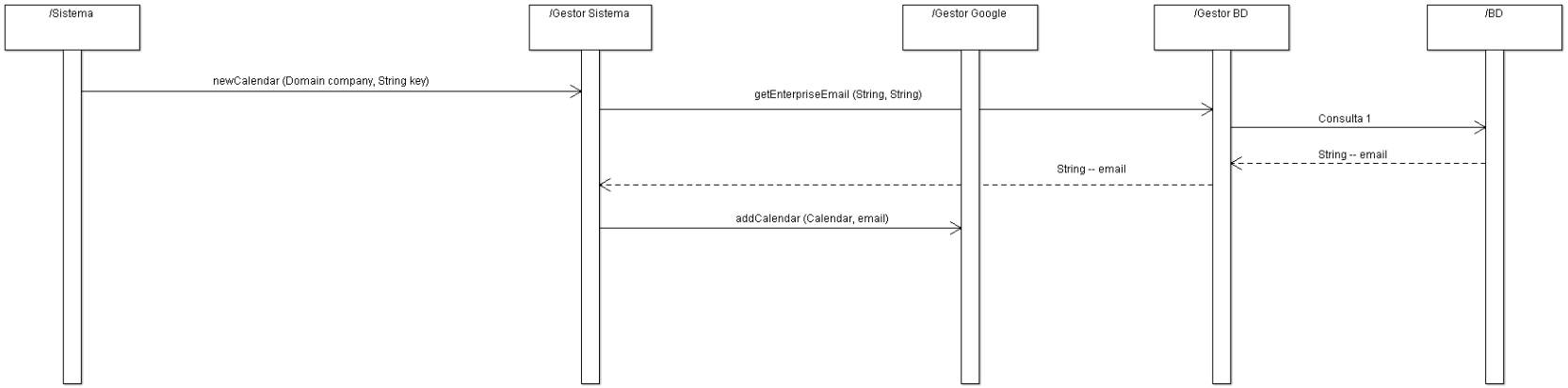
Consulta 2:

```
SELECT *
FROM domain
ORDER BY  name;
```

Consulta 3:

```
SELECT *
FROM commercialTracking;
```

CU: Crear y Añadir Calendario

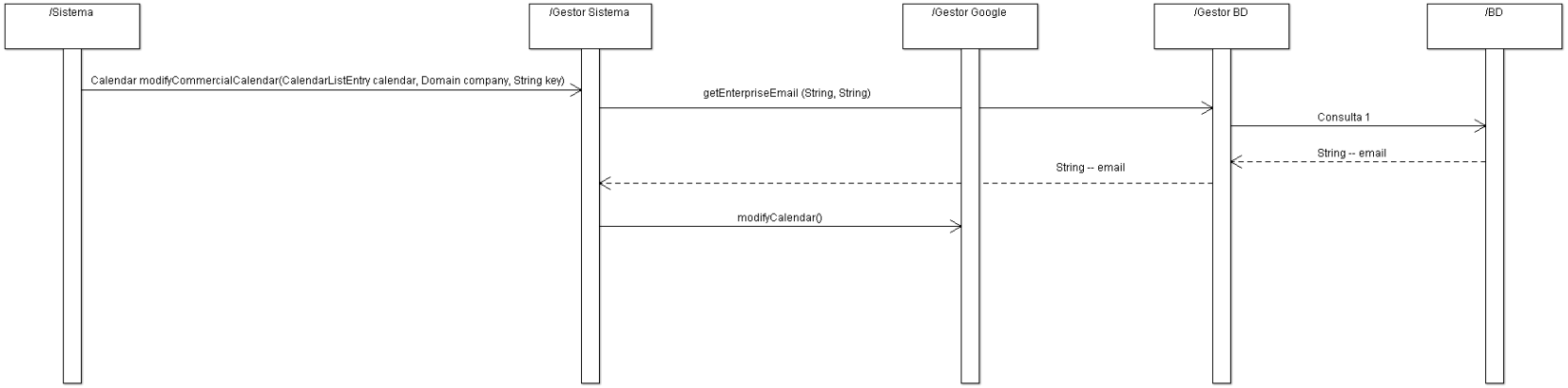


Imágen 5.19: Diagrama de secuencia , Crear y añadir calendario

Consulta 1

```
SELECT RM.value
FROM (domain AS D inner join enterprise AS E
        ON D.id = E.domain) inner join rmedia AS RM
        ON RM.registry=E.registry
WHERE RM.media=4 AND RM.domain= &domainID
```

CU: Modificar Calendario

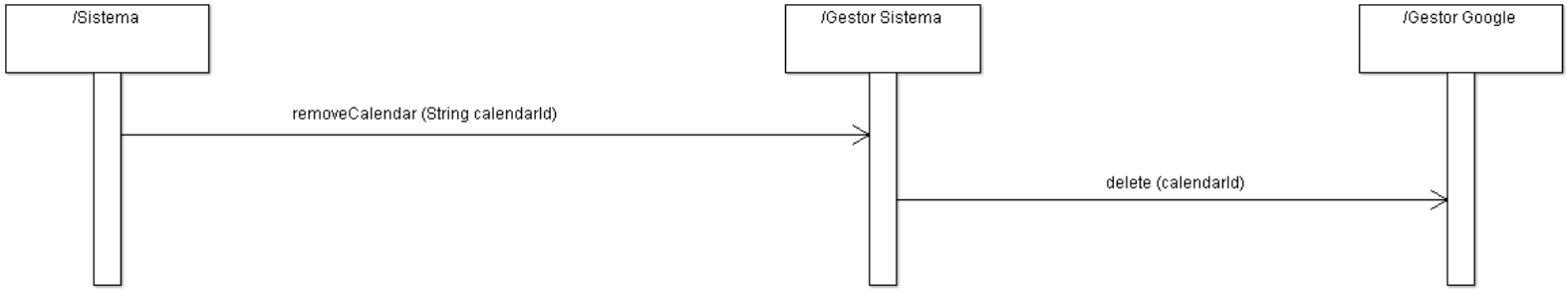


Imágen 5.20: Diagrama de secuencia , modificar calendario

Consulta 1

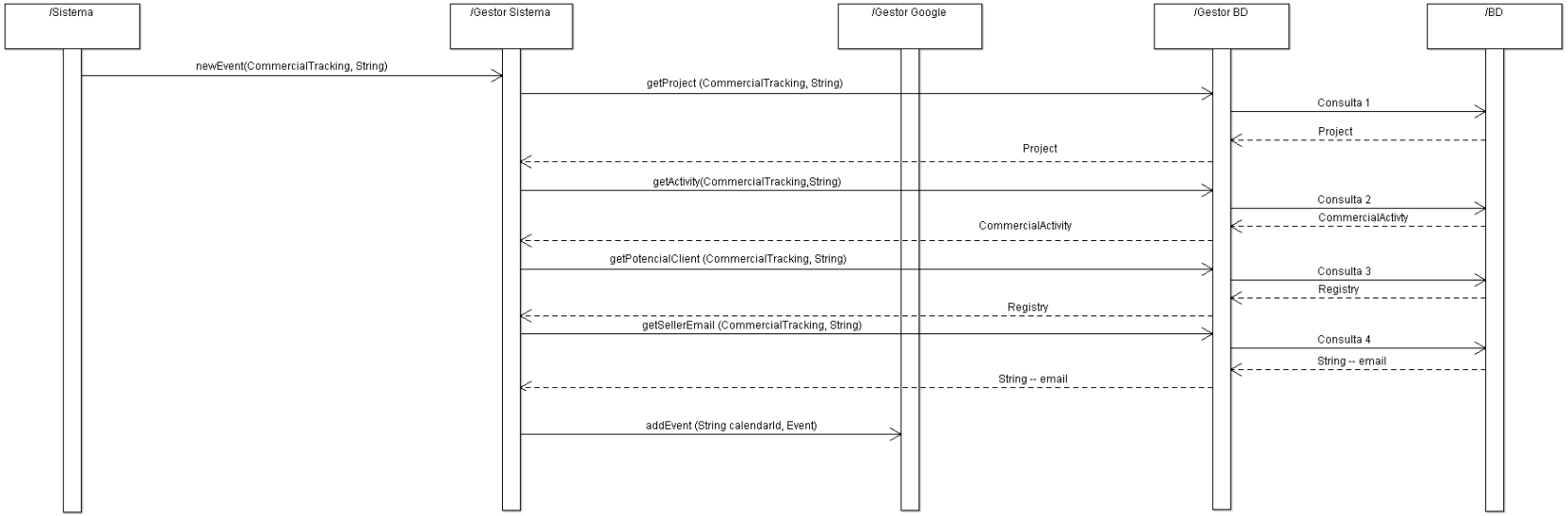
```
SELECT RM.value
FROM (domain AS D inner join enterprise AS E
        ON D.id = E.domain) inner join rmedia AS RM
        ON RM.registry=E.registry
WHERE RM.media=4 AND RM.domain= &domainId
```

CU: Eliminar Calendario



Imágen 5.21: Diagrama de secuencia , Eliminar calendario

CU: Crear y Añadir Evento



Imágen 5.22: Diagrama de secuencia , Crear y añadir evento

Consulta 1

```
SELECT P.*
FROM comercial_tracking AS CT inner join
        Project AS P ON P.id=CT.project_commercial
WHERE CT.id = & commercialId
```

Consulta 2

```
SELECT CA.*
FROM comercial_tracking AS CT inner join
        Comercial_activity AS CA ON CA.id CT.activity
WHERE CT.id= &commercialId
```

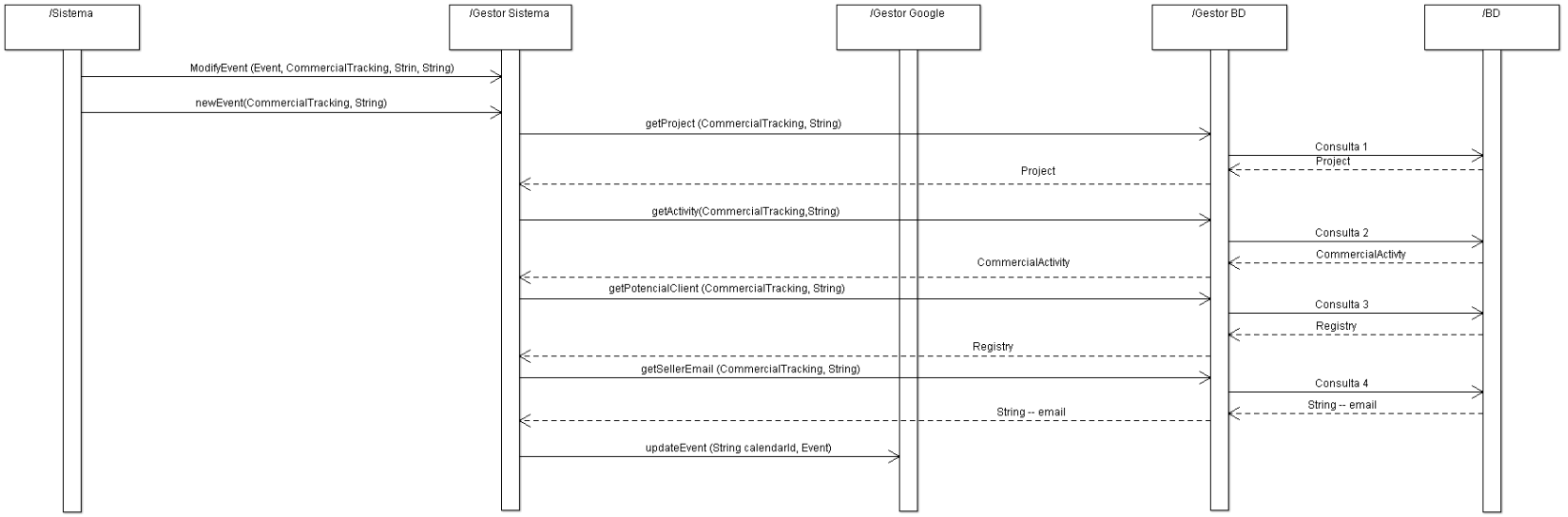
Consulta 3

```
SELECT R.*
FROM (comercial_traclomg AS CT inner join
        Project_commercial AS PC ON
        PC.project=CT.project_Commercial) inner join
        Registry AS R ON R.id= PC.target
WHERE CT.id= &commercialId
```

Consulta 4

```
SELECT RM.value
FROM rmedia AS RM inner join comercial_tracking AS CT
        ON RM.registry = CT.seller
WHERE RM.media = 4 AND CT.id = &commercialId
```

CU: Modificar Evento



Imágen 5.23: Diagrama de secuencia , Modificar evento

Consulta 1

```
SELECT P.*
FROM comercial_tracking AS CT inner join
        Project AS P ON P.id=CT.project_commercial
WHERE CT.id = & commercialId
```

Consulta 2

```
SELECT CA.*
FROM comercial_tracking AS CT inner join
        Comercial_activity AS CA ON CA.id CT.activity
WHERE CT.id= &commercialId
```

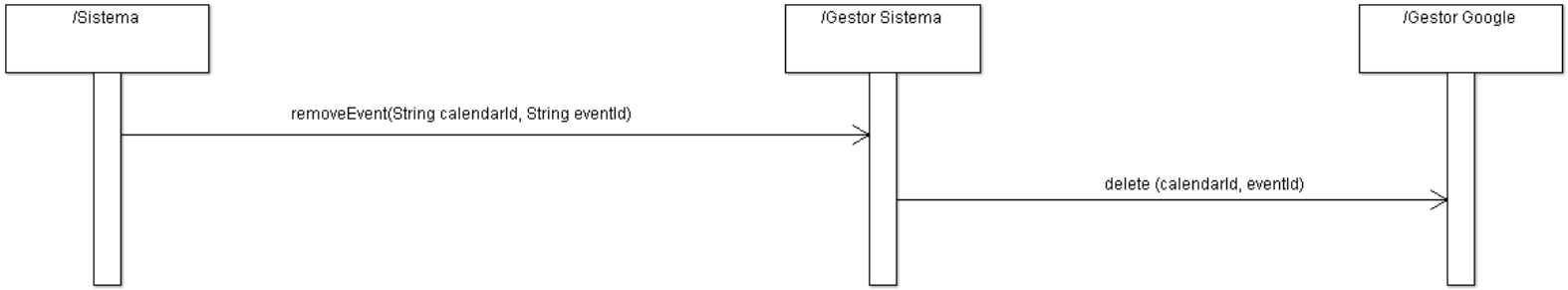
Consulta 3

```
SELECT R.*
FROM (comercial_traclomg AS CT inner join
        Project_commercial AS PC ON
        PC.project=CT.project_Commercial) inner join
        Registry AS R ON R.id= PC.target
WHERE CT.id= &commercialId
```

Consulta 4

```
SELECT RM.value
FROM rmedia AS RM inner join comercial_tracking AS CT
        ON RM.registry = CT.seller
WHERE RM.media = 4 AND CT.id = &commercialId
```

Eliminar Evento



Imágen 5.24: Diagrama de secuencia , Eliminar evento

Las fases de implementación, pruebas e implantación de ésta iteración se detallan en los apartados 8, 9 y 10 respectivamente. Donde se tratarán dentro de cada uno de los apartados, cada una de las fases de las cuatro iteraciones del proyecto.

6. ITERACIÓN 3: GOOGLE TASK API

6.1. Introducción

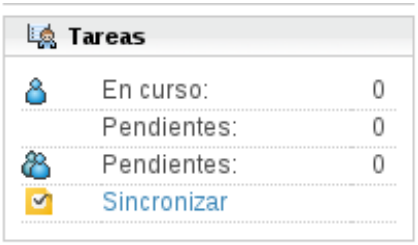
El objetivo de ésta tercera iteración del proyecto, es realizar la sincronización de las tareas de AonSolutions con Google Task.

Para ello se crea una lista de tareas en Google Calendar que corresponden a un proyecto de AonSolutions. Ésta lista incluirá todas las tareas del proyecto a la que corresponde dicha lista de tareas.

6.1.1. Ejecución Visual

En la página de inicio se pueden observar los dos portlet[1] implementados en el proyecto, el de Google Task y el de Google Drive, lo cuales no aperecen en la pantalla de inicio de la aplicación si el usuario no se ha conectado con su cuenta de Google correspondiente.

El widget de **Google Task** y las tareas de AonSolutions, el cuál fue modificado, para sincronizar las tareas del usuario a Google Task, añadiendo dicha opción al portlet (imagen 6.1).



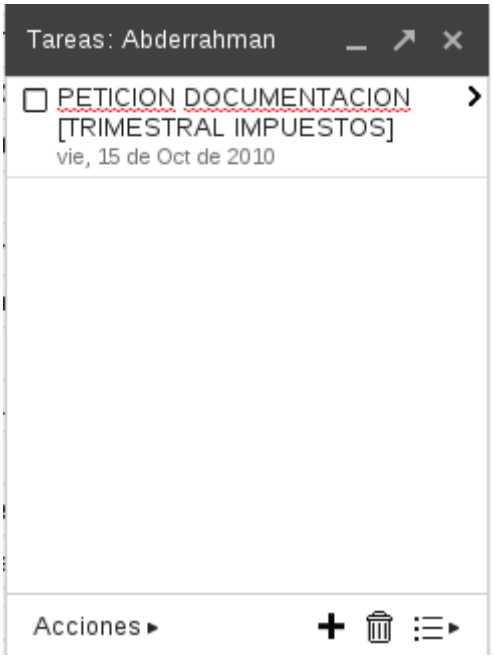
Imágen 6.1: Portlet tareas.

Para sincronizar todos las tareas que el usuario dispone en su cuenta de AonSolutions, simplemente hay que pulsar en sincronizar, y se ejecuta la función synchronize() correspondiente a TaskUtils.

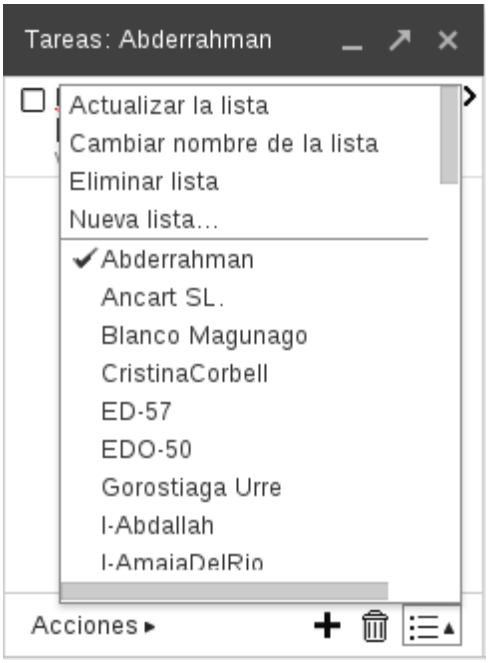
Para comprobar que las tareas se sincronizan correctamente, se ha utilizado una cuenta de google, con la que se pueden ver los resultados, en la interfaz gráfica que Google dispone para su servicio de Google Task.

Tal y como se puede observar en la siguiente imagen (6.2 y 6.3), las tareas y los proyectos que el usuario dispone en la plataforma de AonSolutions, se han sincronizado correctamente con Google Task.

[1] Portlet: Son componentes modulares de las interfaces de usuario gestionadas y visualizadas en un portal web



Imágen 6.2: Tarea de un proyecto Google Task



Imágen 6.3: Lista de proyectos Google Task

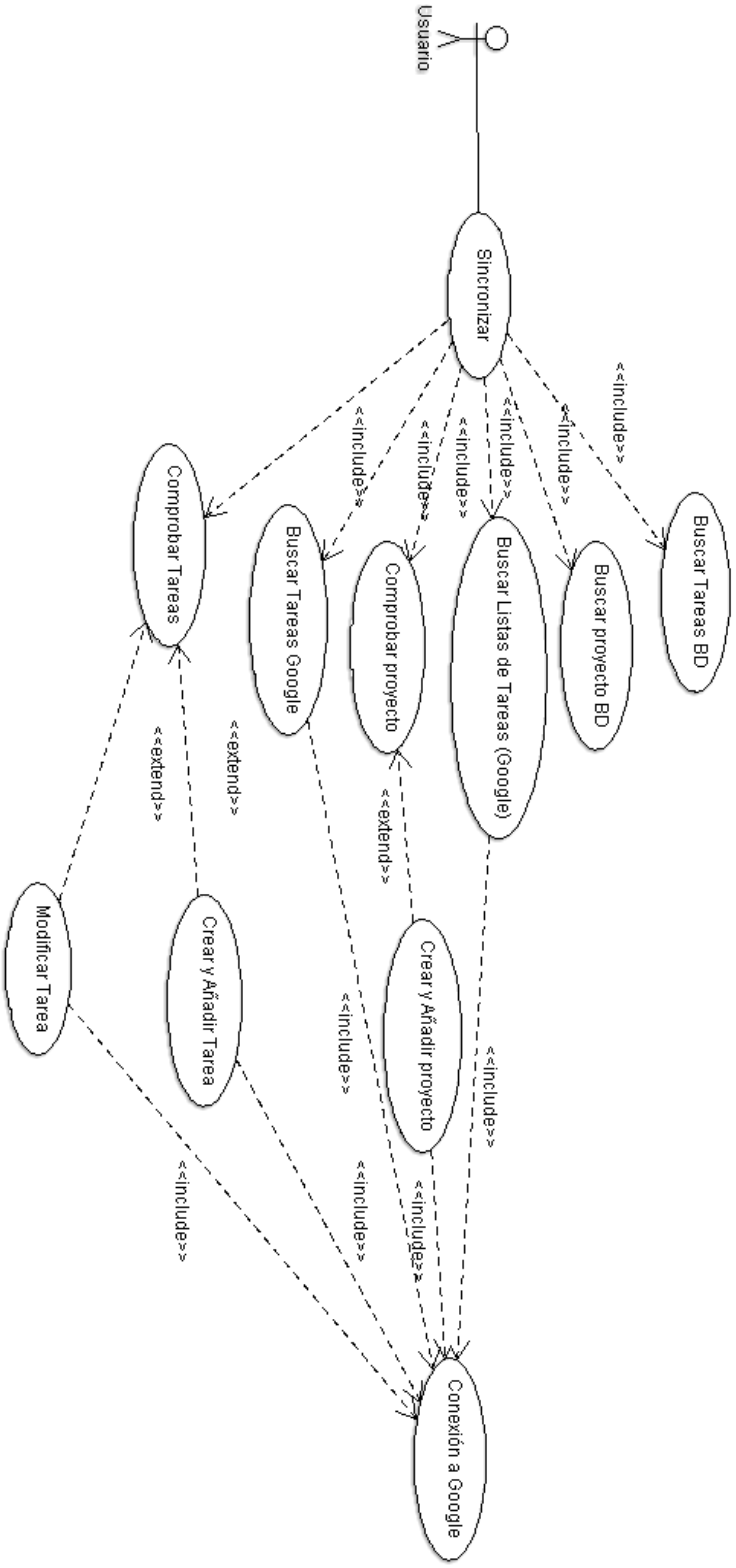
6.2 Captura de Requisitos

6.2.1. Requisitos

REFERENCIA	REQUISITOS
#1	Buscar todas las tareas del base de datos
#2	Buscar el proyecto de una tarea en la base de datos
#3	Crear una lista de tareas de Google Task
#4	Añadir una lista de tareas a Google Task
#5	Modificar una lista de tareas de Google Task.
#6	Eliminar una lista de tareas de Google Task
#7	Crear una tarea de Google Task
#8	Añadit una tarea a Google Task
#9	Modificar una tarea de Google Task
#10	Eliminar una tarea de Google Task
#11	Sincronizar todas las tareas de AonSolutions (BD) a Google Task

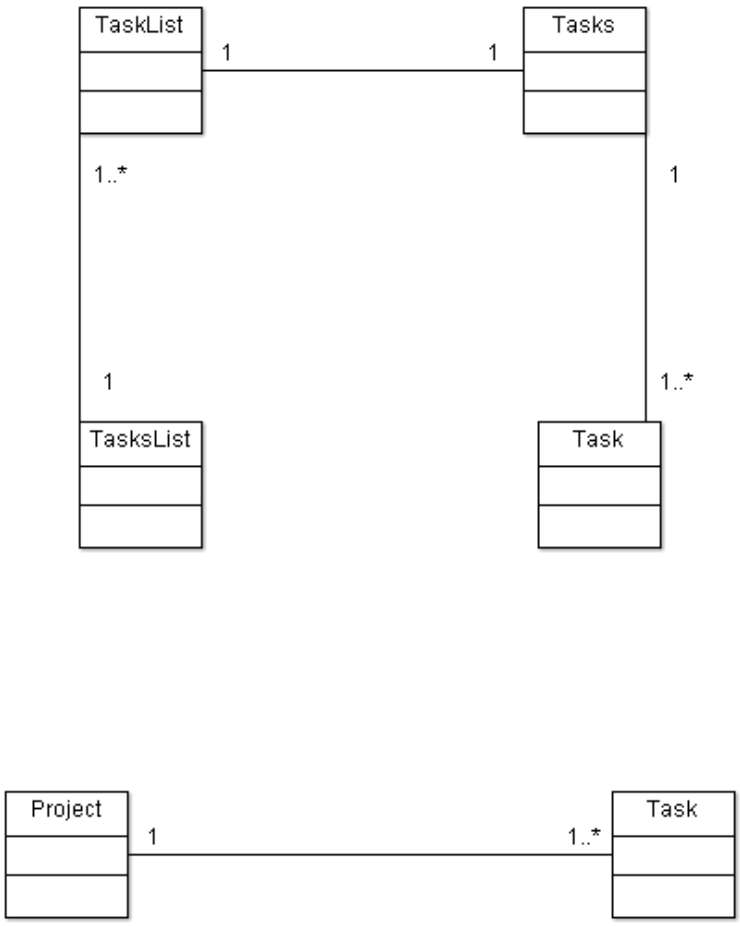
#12	Buscar Todas las listas de tareas (Proyecto) de Google Task.
#13	Buscar todas las tareas de una lista de tareas (Proyecto)
#14	Comprobar si un proyecto (lista de tareas) esta en Google Task.
#15	Comprobar si una tarea esta en una lista de tareas de Google Task.
#16	Conectarse a Google (OpenId).
#17	Ordenar lista de Proyectos de Google Task (TaskLists)
#18	Ordenar lista de tareas de Google Task (Tasks).

6.2.1. Modelo de Casos de Uso



Imágen 6.4: Diagrama de casos de uso de la iteración 2.

6.2.3. Modelo de Dominio



Imágen 6.5: Modelo de dominio de la iteración 2.

En la imagen 6.5 se puede observar el modelo de dominio de la tercera iteración, en la parte superior se encuentran las clases u objetos pertenecientes a Google Task API y en la parte inferior, las clases correspondientes a AonSolutions.

No se especifican ni los atributos ni las funciones de cada clase en el modelo de dominio de la iteración 2, para que no ocupe tanto espacio en la imagen y el diagrama se vea de manera más clara y sencilla. A continuación se especificarán tanto los atributos como las funciones de cada clase / objeto, únicamente las que se han utilizado para el desarrollo de la iteración, para no extenderse.

Task

Corresponde a una tarea de Google Task.

Atributos:

String **Id**: Identificador de la tarea

DateTime **Due**: Fecha de vencimiento de la tarea.

String **Notes**: Notas de describen la tarea.

String **Title**: Titulo o nombre de la tarea.

DateTime **Updated**: Fecha en la que la tarea se ha actualizado por última vez.

Funciones:

getId(): Devuelve el identificar de la tarea.

getDue():Devuelve la fecha de vencimiento de la tarea.

getNotes(): Devuelve las notas que describen la tarea.

getTitle(): Devuelve el título de la tarea.

getUpdated(): Devuelve la fecha de la última modificación de la tarea.

setId(String id): Inserta el valor del identificador de la tarea.

setDue(DateTime due): Inserta el valor de la fecha de vencimiento de la tarea.

setNotes(String notes): Inserta el valor de las notas que describen la tarea.

setTitle(String title): Inserta el valor del título de la tarea.

setUpdated(DateTime): Inserta el valor de la última fecha de modificación de la tarea.

TaskList

Corresponde a una lista de tareas de Google Calendar. Corresponde a un proyecto de AonSolutions.

Atributos:

String **Id**: Identificador de la lista de tareas.

String **Title**: Título de la lista de tareas.

String **Updated**: Última fecha de mofication de la lista de tareas.

Funciones:

getId(): Devuelve el identificador de la lista de tareas.

getTitle(): Devuelve el título de la lista de tareas.

getUpdated(): Devuelve la última fecha de modificación de la lista de tareas.

setId(String id): Inserta el valor del identificador de la lista de tareas.

setTitle(String title): Inserta el valor del título de la lista de tareas.

setUpdated(DateTime Datetime): Inserta el valor de la última fecha de modificación de la lista de tareas.

Tasks

Lista de tareas de Google Calendar.

Atributos:

List<Task> **Items**: Lista de Tareas.

Funciones:

getItems(): Devuelve la lista de tareas, para poder acceder a cada una de las tareas que contenga dicha lista.

setItems(List<Task>): Insertar la lista de tareas.

TasksList

Corresponde a una lista de TaskList de Google calendar, es decir, una lista de proyectos.

Atributos:

List<TaskList> i**tems**: Lista de taskList.

Funciones:

getItems(): Devuelve una lista de taskList.

setItems(List<TaskList> ítems): Inserta el valor de una lista de taskList.

Project

Corresponde a un proyecto de AonSolutions.

Atributos:

Integer **id**: Identificador del proyecto.

String **name**: Nombre del proyecto.

String **alias**: Alias del proyecto.

Date **date**: Fecha de inicio del proyecto.

Funciones:

getId(): Devuelve el identificador del proyecto.

getName(): Devuelve el nombre del proyecto.

getAlias(): Devuelve el alias del proyecto.

getDate(): Devuelve la fechade inicio del proyecto.

setId(Integer id): Inserta el valor del identificador del proyecto.

setName(String name): Inserta el valor del nombre del proyecto.

setAlias(String alias): Inserta el valor del alias del proyecto.

setDate(Date date): Inserta el valor de la fech inicial del proyecto.

Task

Corresponde a una tarea de AonSolutions, correspondiente a un proyecto determinado.

Atributos:

Integer **id**: Identificador de la tarea.

String **description**: Descripción detallada de la tarea.

Date **startDate**: Fecha de inicio de la tarea.

Date **endDate**: Fecha de finalización de la tarea.

Date **dueDate**: Fecha de vencimiento de la tarea.

String **comments**: Comentarios de la tarea.

Funciones:

getId(): Devuelve el identificador de la tarea.

getDescription(): Devuelve una descripción detallada de la tarea.

getStartDate(): Devuelve la fecha de inicio correspondiente a la tarea.

getEndDate(): Devuelve la fecha de finalización correspondiente a la tarea.

getDueDate(): Devuelve la fecha de vencimiento correspondiente a la tarea.

getComments(): Devuelve los comentarios de la tarea.

setId(Integer id): Inserta el valor del identificador de la tarea.

setDescription(String description): Inserta el valor de una descripción detallada de la tarea.

setStartDate(Date startDate): Inserta el valor de la fecha inicial de la tarea.

setEndDate(Date endDate): Inserta el valor de la fecha de finalización de la tarea.

setDueDate(Date dueDate): Inserta el valor de la fecha de venicimiento de la tarea.

setComments(String comments): Inserta el valor de los comentarios de una tarea.

6.2.4. Casos de Uso

CU: Buscar Tareas BD

Actores: Sistema

Descripción: El sistema busca en la base de datos todas las tareas que haya del usuario en la cual se encuentra la sesión actual.

Precondiciones: --

Post-condiciones: --

Referencias: #1

Escenario Principal:
1. El sistema busca todas las tareas en la base de datos.

CU: Buscar Proyecto BD

Actores: Sistema

Descripción: El sistema busca el proyecto correspondiente a una tarea específica.

Precondiciones: Disponer de una tarea (Task).

Post-condiciones: --

Referencias: #2

Escenario Principal:
1. El sistema busca el proyecto de una tarea determinada.

CU: Buscar Listas de Tareas (Proyectos) Google

Actores: Sistema

Descripción: El sistema se conecta con Google y busca todas las listas de tareas de Google Tasks.

Precondiciones: --

Post-condiciones: --

Referencias: #12, #16

Escenario Principal:
1. El sistema se conecta a Google Task.
2. El sistema busca todas las listas de tareas de Google Task

CU: Comprobar Proyecto

Actores: Sistema

Descripción: El sistema ordena la lista de lista de tareas (lista de proyectos) y comprueba si un proyecto se encuentra en Google Task como una lista de Tareas.

Precondiciones: Disponer de una lista de lista de tareas y un proyecto de AonSolutions.

Post-condiciones: --

Referencias: #17, #14

Escenario Principal:
1. El sistema ordena la lista de proyectos de Google Task
2. El sistema comprueba si la lista de lista de tareas contiene un proyecto dado.

CU: Buscar Tareas Google

Actores: Sistema

Descripción: El sistema se conecta a Google Task y luego busca todas las tareas que hay en la cuenta del usuario de Google Task en una lista de tareas determinada (TaskList).

Precondiciones: Disponer del identificador de la lista de tareas correspondiente.

Post-condiciones: --

Referencias: #16, #13

Escenario Principal:
1. El sistema se conecta a Google Task.
2. El sistema busca la tareas de una lista de Tareas (tasks)

CU: Comprobar Tareas

Actores: Sistema

Descripción: El sistema ordena una lista de tareas y luego comprueba si una determinada tarea se encuentra en dicha lista ordenada o no.

Precondiciones: Disponer de una lista de tareas y de una tarea.

Post-condiciones: --

Referencias: #15, #18

Escenario Principal:
1. El sistema ordena una lista de tareas de Google Task
2. El sistema comprueba si una tarea se encuentra en la lista de tareas de Google Task.

CU: Crear y Añadir Proyecto

Actores: Sistema

Descripción: El sistema se conecta a Google, crea una lista de tareas (proyecto) y la añade a Google Task.

Precondiciones: Disponer de un proyecto.

Post-condiciones: El proyecto se encuentra en Google Task.

Referencias: #3, #4,#16

Escenario Principal:
1. El sistema crea una lista de Tareas (TaskList) como proyecto.
2. El sistema se conecta a Google Task.
3. El sistema añade dicha lista de Tareas a Google Task.

CU: Modificar Proyecto

Actores: Sistema

Descripción: El sistema se conecta a Google, crea una lista de tareas (proyecto) y actualiza la lista de tareas de Google Task.

Precondiciones: Disponer de un proyecto actualizado y el identificador de la lista de tareas a actualizar.

Post-condiciones: El proyecto se encuentra en Google Task.

Referencias: #5, #16

Escenario Principal:
1. El sistema crea una lista de Tareas (TaskList) como proyecto.
2. El sistema se conecta a Google Task.
3. El sistema actualiza dicha lista de Tareas a Google Task.

CU: Eliminar Proyecto

Actores: Sistema

Descripción: El sistema elimina una lista tareas de Google Task.

Precondiciones: Disponer del identificador de la lista de tareas a eliminar.

Post-condiciones: La lista de tareas ya no está disponible en Google Task.

Referencias: #6, #16

Escenario Principal:
1. El sistema se conecta a Google.
2. El sistema elimina la lista de tareas correspondiente.

CU: Crear y Añadir Tarea

Actores: Sistema

Descripción: El sistema se conecta a Google, crea y añade una tarea en la cuenta del usuario de Google Task.

Precondiciones: Disponer de una Tarea a añadir a Google Task.

Post-condiciones: La tarea ya se encuentra accesible en Google Task.

Referencias: #7, #8, #16

Escenario Principal:
1. El sistema crea una tarea de Google Task.
2. El sistema se conecta con Google.
3. El sistema añade la tarea creada a Google Task.

CU: Modificar Tarea

Actores: Sistema

Descripción: El sistema crea una tarea de Google Task, se conecta con Google y Modifica o actualiza la tarea correspondiente.

Precondiciones: Disponer de la tarea actualizada y del identificador de la tarea a actualizar de Google Task.

Post-condiciones: La tarea ya se encuentra actualizada en Google Task.

Referencias: #9, #16

Escenario Principal:
1. El sistema crea una tarea de Google Task.
2. El sistema se conecta con Google.
3. El sistema modifica la tarea correspondiente.

CU: Eliminar Tarea

Actores: Sistema

Descripción: El sistema elimina una tarea de Google Task.

Precondiciones: Disponer del identificador de la tarea a eliminar.

Post-condiciones: La tarea ya no está disponible en Google Task.

Referencias: #10,#16

Escenario Principal:
1. El sistema se conecta a Google.
2. El sistema elimina la tarea correspondiente.

CU: Sincronizar

Actores: Sistema, Usuario

Descripción: El usuario pulsa el botón de sincronizar, para sincronizar las tareas de AonSolutions del usuario con su cuenta de Google Task. Para ello el sistema tiene que realizar determinadas funciones que corresponde a la mayoría de los casos de uso detallados con anterioridad. *Buscar tareas BD, Buscar proyecto BD, Buscar lista de tareas Google, Comprobar proyecto, Buscar tareas Google, Crear y Añadir proyecto, Crear y Añadir tarea* y *Modificar tarea.*

Precondiciones: --

Post-condiciones: Todas las tareas del usuario correspondientes ya están sincronizadas con Google Task.

Referencias: #1,#2,#3,#4,#7,#8,#9,#11,#12,#13,#14,#15,#16,#17,#18

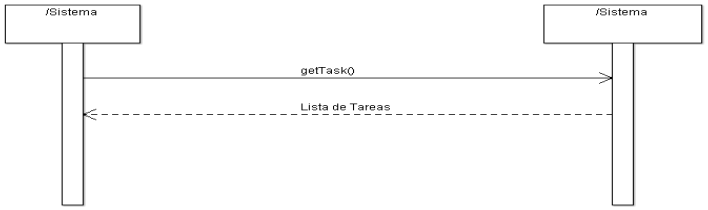
Escenario Principal:
1. El usuario pulsa el botón de sincronizar.
2. El sistema busca las tareas del usuario en la base de datos.

Para todas las tareas:
3. El sistema busca el proyecto de una tarea.
4. El sistema busca las listas de tareas de Google.
5. El sistema comprueba que el proyecto se encuentre en la lista de tareas.
    a. Si no está el proyecto en la lista de lista de tareas, crea y añade la lista de tareas (proyecto) a la lista de lista tareas.
6. El sistema busca las tareas de una lista de tareas.
7. El sistema comprueba que que una tarea está en la lista de tareas.
    a. Si no está la tarea en la lista de tareas, crea y añade la tarea a la lista de tareas.
    b. Si está la tarea en la lista de tareas, actualiza la tarea si es necesario.

6.3. Análisis

CU: Buscar Tareas BD

Diagrama de secuencia del sistema



Imágen 6.6: Diagrama de secuencia del sistema, buscar tareas BD.

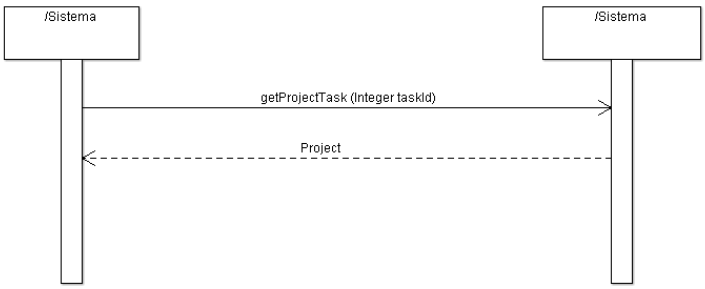
Contratos

Contrato 1:

- Nombre: getTask()
- Responsabilidades: busca en la base de datos todas las tareas del usuario.
- Excepciones: --
- Precondiciones: --
- Post-condiciones: --
- Salida: Lista de Tareas.

CU: Buscar Proyecto BD

Diagrama de secuencia del sistema



Imágen 6.7: Diagrama de secuencia del sistema, buscar proyecto BD.

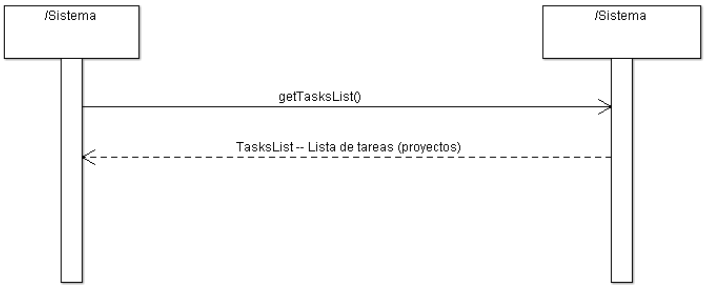
Contratos

Contrato 1:

- Nombre: getProjectTask(Integer taskId)
- Responsabilidades: El sistema busca el proyecto de una tarea en la base de datos.
- Excepciones: --
- Precondiciones: Disponer del identificador de una tarea.
- Post-condiciones: --
- Salida: Lista de Proyectos.

CU: Buscar Listas de Tareas (Proyectos) Google

Diagrama de secuencia del sistema



Imágen 6.8: Diagrama de secuencia del sistema, buscar listas de tareas.

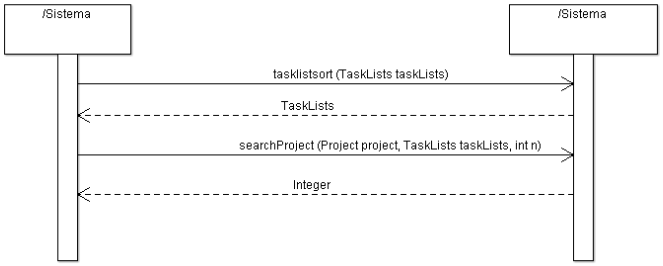
Contratos

Contrato 1:

- Nombre:getTasksList()
- Responsabilidades: El sistema se conecta a Google y busca en Google Task la lista de proyectos (lista de lista de tareas).
- Excepciones: --
- Precondiciones: --
- Post-condiciones: --
- Salida: Listas de Tareas , TasksList

CU: Comprobar Proyecto

Diagrama de secuencia del sistema



Imágen 6.9: Diagrama de secuencia del sistema, comprobar proyecto.

Contratos

Contrato 1:

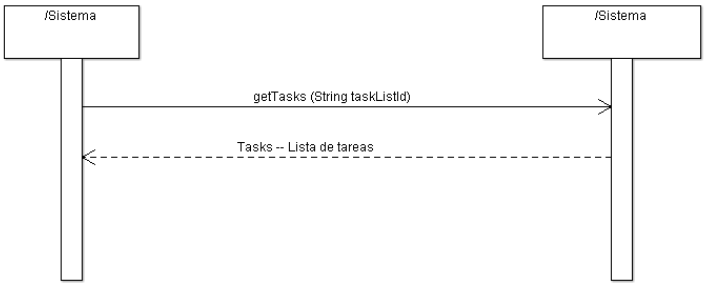
- Nombre: tasklistsort(TaskLists taskLists)
- Responsabilidades: Devuelve la lista de proyectos de Google Task (Tasklists) ordenada, mediante el algoritmo quicksort.
- Excepciones: --
- Precondiciones: Disponer de una lista de listas de tareas.
- Post-condiciones: --
- Salida: TaskLists

Contrato 2:

- Nombre: searchProject(Project project, TaskLists taskLists , int n)
- Responsabilidades: Devuelve el número de la posición en la que se encuentra el proyecto en la lista, y si éste no está en ella, devuelve “-1”.
- Excepciones: --
- Precondiciones: Disponer de un proyecto y de una lista de listas de tareas.
- Post-condiciones: --
- Salida: Integer

CU: Buscar Tareas Google

Diagrama de secuencia del sistema



Imágen 6.10: Diagrama de secuencia del sistema, buscar tareas Google.

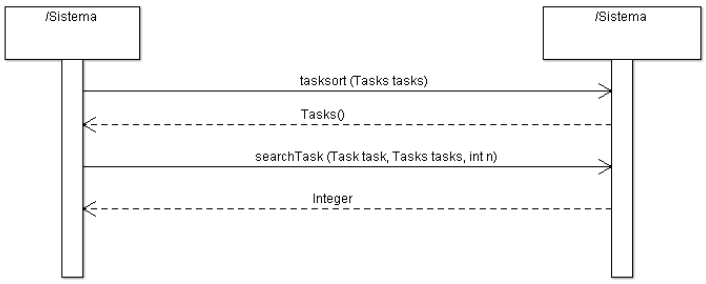
Contratos

Contrato 1:

- Nombre: getTasks(String taskListId)
- Responsabilidades: El sistema se conecta a Google y devuelve una lista de tareas.
- Excepciones: --
- Precondiciones: Disponer del identificador de TaskList
- Post-condiciones: --
- Salida: Tasks (Lista de tareas).

CU: Comprobar Tareas

Diagrama de secuencia del sistema



Imágen 6.11: Diagrama de secuencia del sistema, comprobar tareas.

Contratos

Contrato 1:

- Nombre: tasksort(Tasks tasks)
- Responsabilidades: Devuelve una lista de tareas de Google Task ordenada.
- Excepciones: --
- Precondiciones: Disponer de una lista de tareas de Google Task (Tasks)
- Post-condiciones: --
- Salida: Tasks (lista de tareas)

Contrato 1:

- Nombre: searchTask(Task task, Tasks tasks,int n)
- Responsabilidades: Devuelve el número de la posición en la que se encuentra la tarea en la lista, y si éste no está en ella, devuelve “-1”.
- Excepciones: --
- Precondiciones: Disponer de una tarea y de una lista de tareas de Google Task.
- Post-condiciones: --
- Salida: Integer.

CU: Crear y Añadir Proyecto

Diagrama de secuencia del sistema

Imágen 6.12: Diagrama de secuencia del sistema, crear y añadir proyecto.

Contratos

Contrato 1:

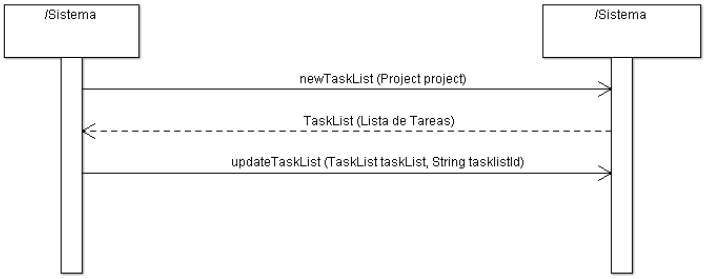
- Nombre: newTaskList(Project Project)
- Responsabilidades: El sistema crea una lista de tareas a partir del Proyecto dado y la devuelve.
- Excepciones: --
- Precondiciones: Disponer de un Proyecto.
- Post-condiciones: --
- Salida: TaskList (lista de Tareas).

Contrato 2:

- Nombre: addTaskList(TaskList taskList)
- Responsabilidades: El sistema se conecta a Google y añade la lista de tareas a Google Task.
- Excepciones: --
- Precondiciones: Disponer de una lista de tareas.
- Post-condiciones: La lista de tareas ya está disponible en Google Task
- Salida: --

CU: Modificar Proyecto

Diagrama de secuencia del sistema



Imágen 6.13: Diagrama de secuencia del sistema,modificar proyecto.

Contratos

Contrato 1:

- Nombre: newTaskList(Project Project)
- Responsabilidades: El sistema crea una lista de tareas a partir del Proyecto dado y la devuelve.
- Excepciones: --
- Precondiciones: Disponer de un Proyecto.
- Post-condiciones: --
- Salida: TaskList (lista de Tareas).

Contrato 1:

- Nombre: updateTaskList(TaskList taskList, String taskListId)
- Responsabilidades: El sistema se conecta a Google y actualiza la lista de tareas de Google Task correspondiente
- Excepciones: --
- Precondiciones: Disponer de una lista de tareas actualizada y el identificado de la lista de tareas a actualizar en Google Task.
- Post-condiciones: La lista de tareas ya se puede encontrar actualizada en Google Task.
- Salida: --

CU: Eliminar Proyecto

Diagrama de secuencia del sistema

Imágen 6.14: Diagrama de secuencia del sistema, eliminar .

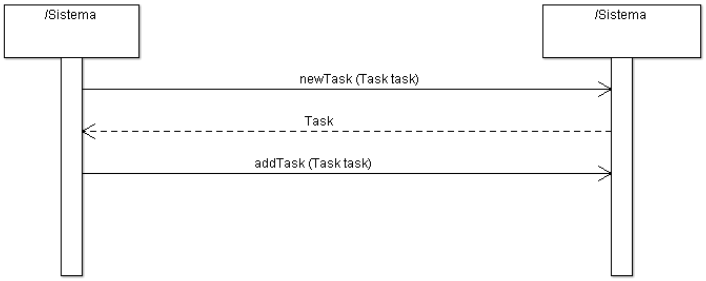
Contratos

Contrato 1:

- Nombre: deleteTaskList(String taskListId)
- Responsabilidades: El sistema se conecta a Google y borra la lista de tareas correspondiente al identificador dado.
- Excepciones: --
- Precondiciones: Identificador de una lista de tareas de Google Task
- Post-condiciones: La lista de tareas a eliminar ya no se encuentra en Google Task.
- Salida: --

CU: Crear y Añadir Tarea

Diagrama de secuencia del sistema



Imágen 6.15: Diagrama de secuencia del sistema, Crear y añadir tarea.

Contratos

Contrato 1:

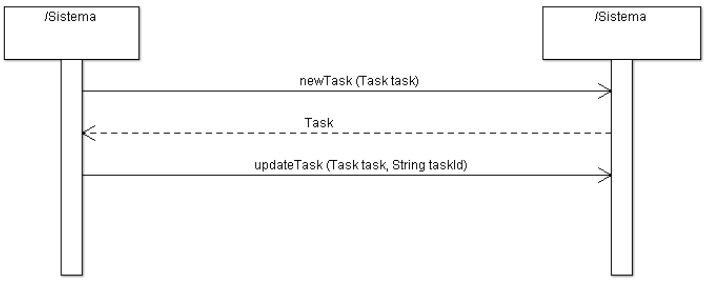
- Nombre: newTask(Task task)
- Responsabilidades: El sistema Crea una tarea de Google Task.
- Excepciones: --
- Precondiciones: Disponer de una tarea de AonSolutions.
- Post-condiciones: --
- Salida: Task (de Google Task)

Contrato 2:

- Nombre: addTask(Task task)
- Responsabilidades: El sistema se conecta a Google Task y añade a la tarea correspondiente.
- Excepciones: --
- Precondiciones: Disponer de una tarea.
- Post-condiciones: La tarea ya está disponible en Google Task.
- Salida: --

CU: Modificar Tarea

Diagrama de secuencia del sistema



Imágen 6.16: Diagrama de secuencia del sistema, modificar tarea

Contratos

Contrato 1:

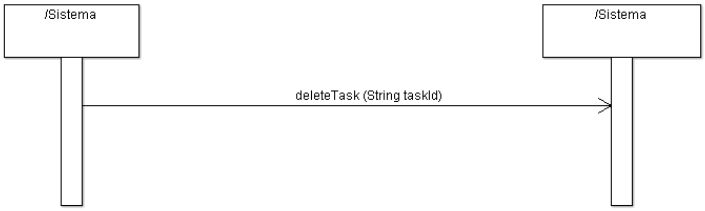
- Nombre: newTask(Task task)
- Responsabilidades: El sistema crea una tarea de Google Task con la tarea de AonSolutions y actualizada.
- Excepciones: --
- Precondiciones: Disponer de una tarea de AonSolutions actualizada.
- Post-condiciones: --
- Salida: Task (tarea de Google).

Contrato 2:

- Nombre: updateTask(Task task, String TaskId)
- Responsabilidades: El sistema se conecta a Google y actualiza la tarea de Google Task,
- Excepciones: --
- Precondiciones: Disponer de una tarea actualiza de Google Task y del identificador de la tarea de Google Task a actualizar.
- Post-condiciones: La tarea ya está disponible en Google Task.
- Salida: --

CU: Eliminar Tarea

Diagrama de secuencia del sistema



Imágen 6.17: Diagrama de secuencia del sistema, Eliminar tarea.

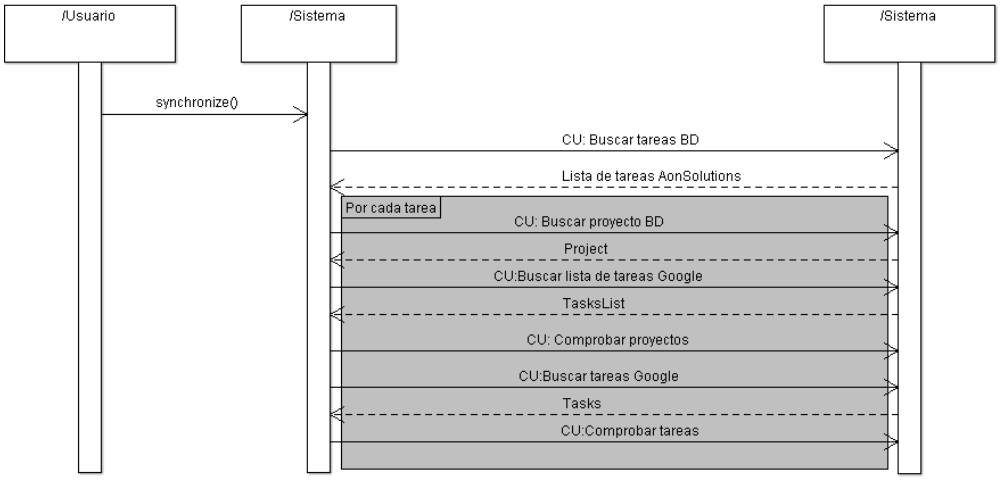
Contratos

Contrato 1:

- Nombre: deleteTask(String taskId)
- Responsabilidades: El sistema se conecta a Google y borra la tarea correspondiente al identificador dado.
- Excepciones: --
- Precondiciones: Identificado de una tarea de Google Task
- Post-condiciones: La tarea a eliminar ya no se encuentra en Google Task.
- Salida: --

CU: Sincronizar

Diagrama de secuencia del sistema



Imágen 6.18: Diagrama de secuencia del sistema, sincronizar.

Contratos

Contrato 1:

- Nombre: synchronize()
- Responsabilidades: El sistema sincroniza todas las tareas de AonSolutions del usuario con Google Task.
- Excepciones: --
- Precondiciones: --
- Post-condiciones: Todas las tareas de AonSolutions de un usuario ya están sincronizadas con Google Task.
- Salida: --

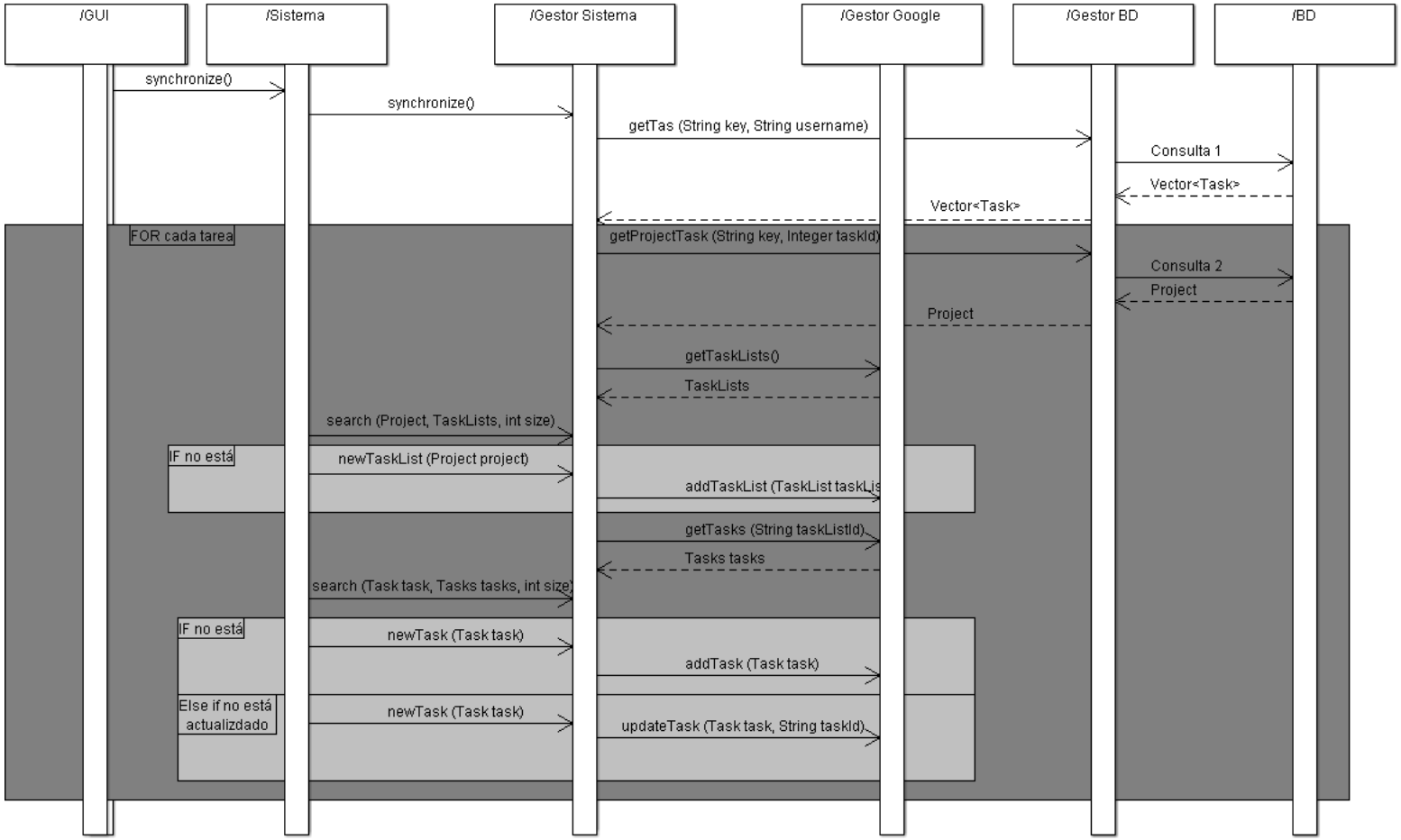
6.4. Diseño

Para reducir el número de diagramas de secuencia en este documento se ha decidido realizar el diagrama de secuencia de la mayoría de los casos de uso en uno, precisamente el del caso de uso sincronizar, ya que abarca la gran mayoría de los casos de uso de ésta iteración.

Por otra parte tampoco se añadirán los diagramas de secuencia de los casos de uso *Crear y añadir proyecto, Crear y añadir tarea* y *Modificar tarea ,* tal y como se han añadido en el diseño de Google calendar, debido a que en el caso de este modulo se especifican claramente en diagrama de secuencia Google Task.

Algunos de estos casos de uso no se utilizan dutante el proceso de sincronización, exactamente *Modificar proyecto*, *Eliminar proyecto* y *Eliminar tarea* que se desarrollarán con vistas al futuro pudiendo necesitarlos en algún otro momento. En el caso de estos tres casos de uso si qu ese añadirá su diagrama de secuencia correspondiente.

CU: Sincronizar



Imágen 6.19: Diagrama de secuencia,sincronizar.

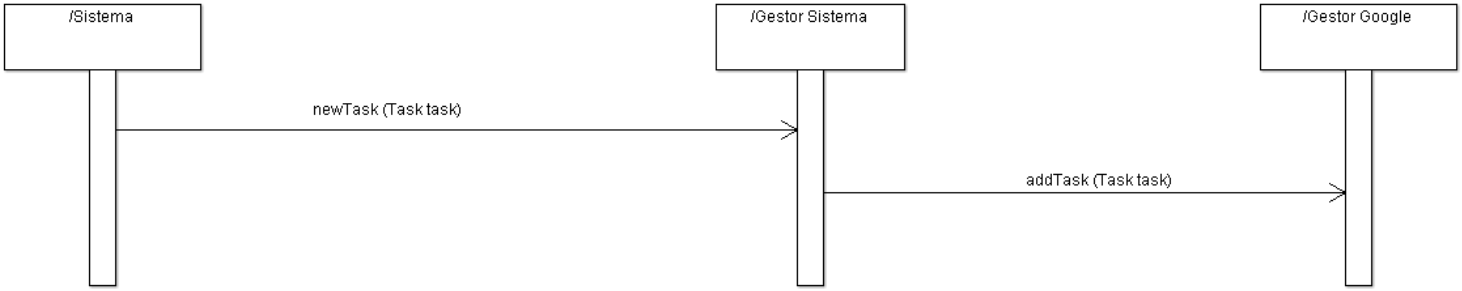
Consulta 1

```
SELECT T.*

FROM task AS T INNER JOIN user AS U
        ON T.task_holder = U.registry

WHERE U.login = username
```

Consulta 2

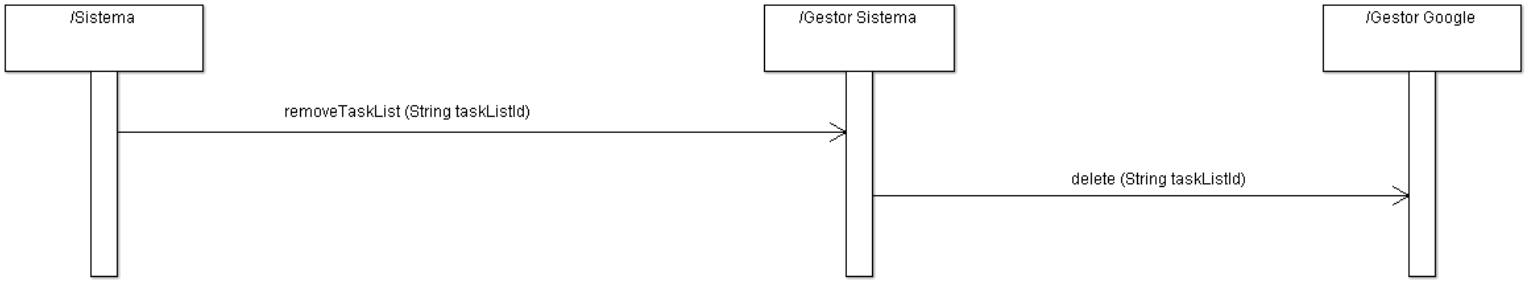
```
SELECT P.*

FROM task AS T INNER JOIN Project AS P
        ON T.projecto = P.id

WHERE T.id=taskId
```

CU: Modificar proyecto



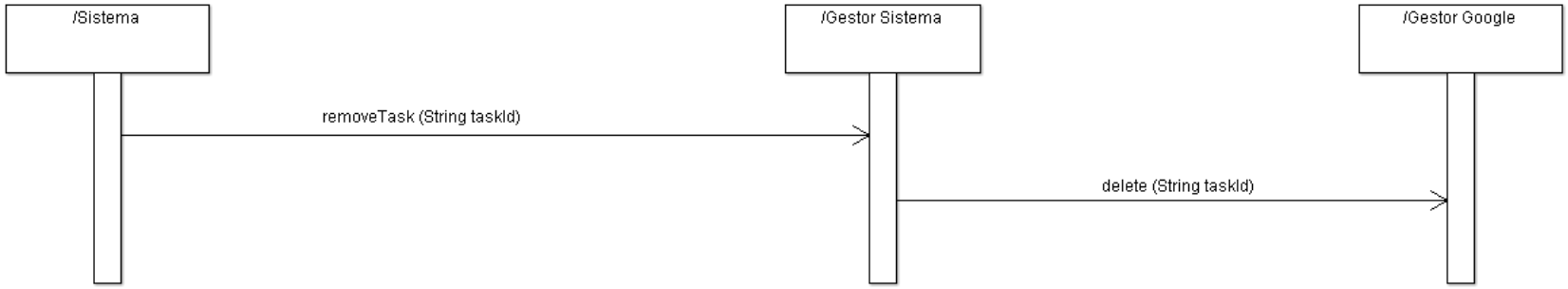
Imágen 6.20: Diagrama de secuencia,modificar proyecto.

CU: Eliminar Proyecto



Imágen 6.21: Diagrama de secuencia,eliminar proyecto.

CU: Eliminar Task



Imágen 6.22: Diagrama de secuencia,Eliminar tarea.

Las fases de implementación, pruebas e implantación de ésta iteración se detallan en los apartados 8, 9 y 10 respectivamente. Donde se tratarán dentro de cada uno de los apartados, cada una de las fases de las cuatro iteraciones del proyecto.

7. ITERACIÓN 4: GOOGLE DRIVE API

7.1. Introducción

El objetivo de ésta última iteración del proyecto, es realizar la sincronización de los documentos de AonSolutions con Google Drive.

Por una parte se subirán todos los archivos, de los tipos de fichero de AonSolutions especificados, con una cuenta de servicio de Google, liberando espacio de la base de datos de Aonsolutions.

A la hora de subir o descargar un archivo desde la aplicación lo realizará desde Google Drive.

El sistema también muestra en la pantalla de inicio del usuario, un widget con los documentos de su cuenta personal de Google Drive. Desde el cuál se podrá subir, borrar o descargar un archivo.

7.1.1. Ejecución Visual

En la página de inicio se pueden observar los dos portlet implementados en el proyecto, el de Google Task y el de Google Drive, lo cuales no aperecen en la pantalla de inicio de la aplicación si el usuario no se ha conectado con su cuenta de Google correspondiente.

Para el widget de **Google Drive**, el cuál imprime los últimos archivos que el usuario ha modificado en su cuenta de Google Drive, se dispone de las opciones de subida de archivos, descarga de archivos y eliminación de archivos, únicamente de los archivos de su propia cuenta de Drive.

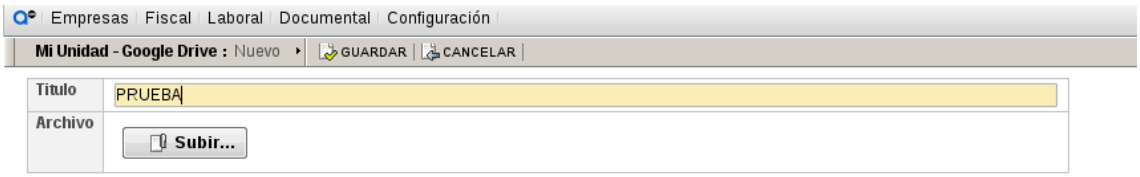
Si se pulsa en la cabecera del portlet, se actualizará la lista de archivos, y al pulsar sobre el nombre del archivo de la lista, se abrirá un anueva ventana, para poder visualizar dicho archivo desde la interfaz gráfica de Google Drive.

Por último si el archivo correspondiente que aparece en la lista, es una carpeta o un archivo de Google docs, no se podrán descargar desde aon, pero si eliminar. Para visualizarlo se puede pulsar sobre el nombre tal y como se ha comentado en el anterior párrafo.

Imágen 7.1: Portlet Google Drive.

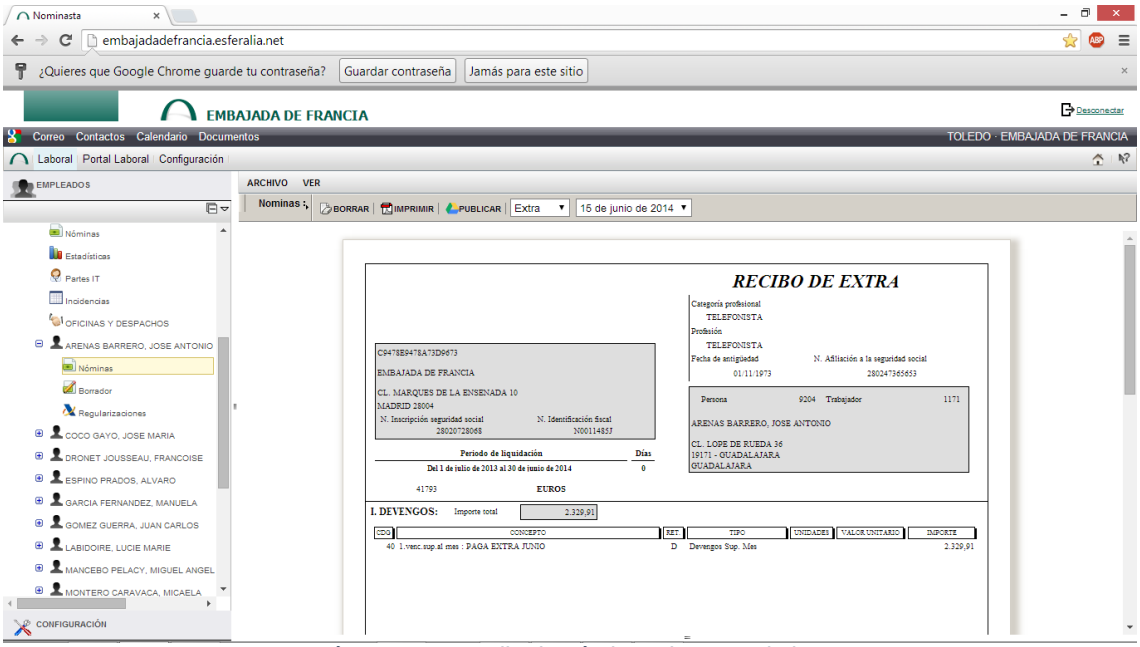
En el portlet de la imagen 7.1 se pueden observar todas las opciones descritas anteriormente en los párrafos superiores. Tal y como se ha dicho las carpetas y los archivos de Google docs, no se podrán descargar, para verlo de forma clara en el portlet, el icono de descargar tiene un tono grisáceo, indicando que esa opción no esposible.

Por último está la opción de subida de archivos, en la esquina derecha del portlet, la cuál redirigirá a otra pantalla de la aplicación de AonSolutions (imagen 7.2), para realizar la subida del archivo que el usuario desee.



Imágen 7.2: Subir un archivo a Google Drive.

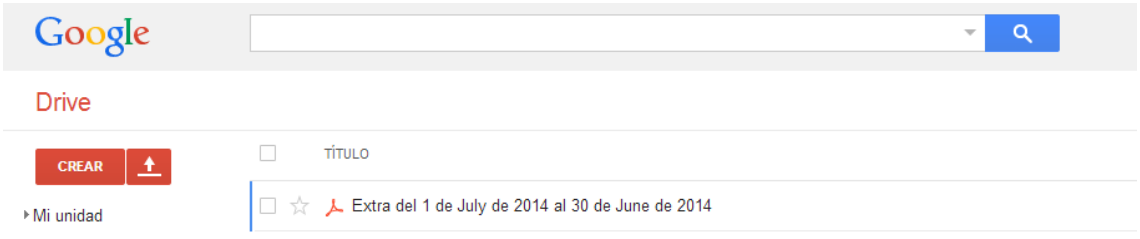
En la parte donde se tratan la nóminas de los empleados (Laboral – integral de nominas), se añadirá un botón donde se podrá publicar la nómina en la cuenta de Google Drive del empleado. De esta manera podrá disponer de todas sus nóminas fácilmente. (Imagen 7.3).



Imágen 7.3: Pantalla de nóminas de AonSolutions.

Para poder disponer de este servicio es necesario que el empleado tenga una cuenta de Google en la aplicación.

Una vez se pulsa el botón de publicar en Google Drive, y se confirma que la operación se ha llevado a cabo con éxito, al empleado le llegará un correo electrónico con en link al archivo de Google drive. También se podrá acceder a la nomina desde la interfaz gráfica de Google Drive (Imagen 7.4), en el apartado de archivos compartidos, tal y como se puede observar en la siguiente imagen.



Imágen 7.4: Nómina subida a Google Drive.

7.1.2. Ejecución Interna

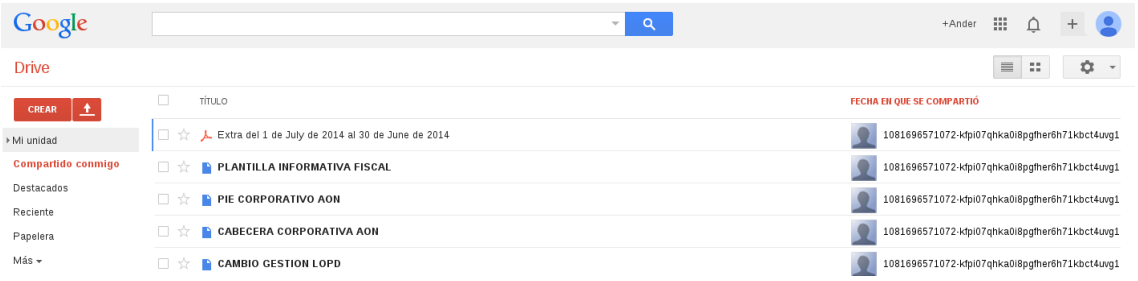
Ésta es la parte donde la funcionalidad no se puede observar desde la interfaz gráfica de la plataforma de gestión empresarial AonSolutions. Por ello se han diferenciado dos apartados en esta parte de la documentación.

Se dispone de una función main, la cuál llama a las funciones necesarias para poder realizar la sincronización correctamente. Esta función main, será ejecutada cada cierto tiempo, normalmente en horario nocturno, para asegurarse de que el sistema sigue sincronizado de forma correcta. Pero lo importante de ésta parte es la primera ejecución, ya que después de esta primera ejecución se podría realizar un seguimiento sin necesidad de volver a ejecutarlo, aunque si que se vuelve ejecutar, eso si con el objetivo ya descrito del aseguramente de la sincronización.

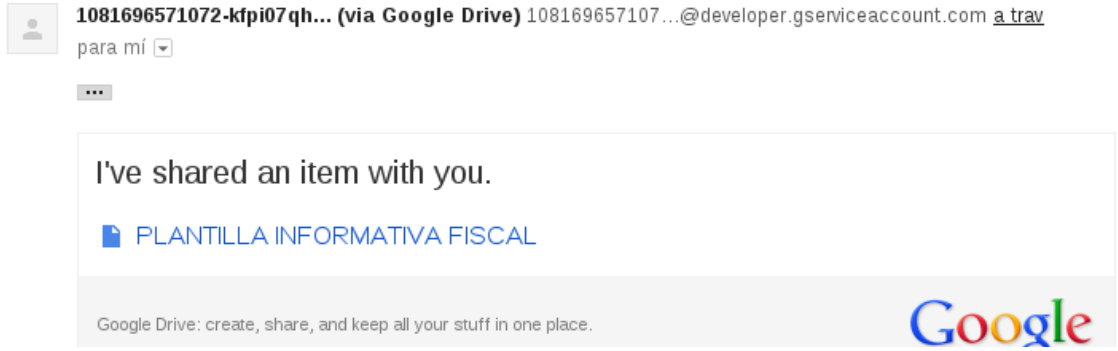
Se ejecutará la sincronización de la cuarta iteración, es decir, la sincronización de los archivos de AonSolutions con Google Drive. En esta ocasión para realizar la prueba se utilizará una cuenta de Google que simulará ser un usuario de la aplicación el cuál tenga acceso al susodicho archivo.

Para realizar la sincronización es necesario especificar que tipos de archivos se quieren sincronizar con Google Drive, y de forma opcional el dominio del que se subirán los archivos. En caso de que no se le pase ningún dominio, se sincronizarán todos los dominios los cuales dispongan del servicio que ofrece la empresa AonSolutions, para la sincronización de archivos.

Una vez realizado el proceso de sincronización se ha llevado a cabo, se puede observar tanto en la cuenta del usuario de Google drive en el apartado de archivos compartidos, o en la bandeja de entrada de Gmail, donde se envía una notificación de que un archivo ha sido compartido junto con el link correspondiente al archivo sincronizado. Todo esto se puede observar en las siguientes imágenes (7.5 y 7.6).



Imágen 7.5: Archivos compartido Google Drive.



Imágen 7.6: Notificación con el link del archivo compartido.

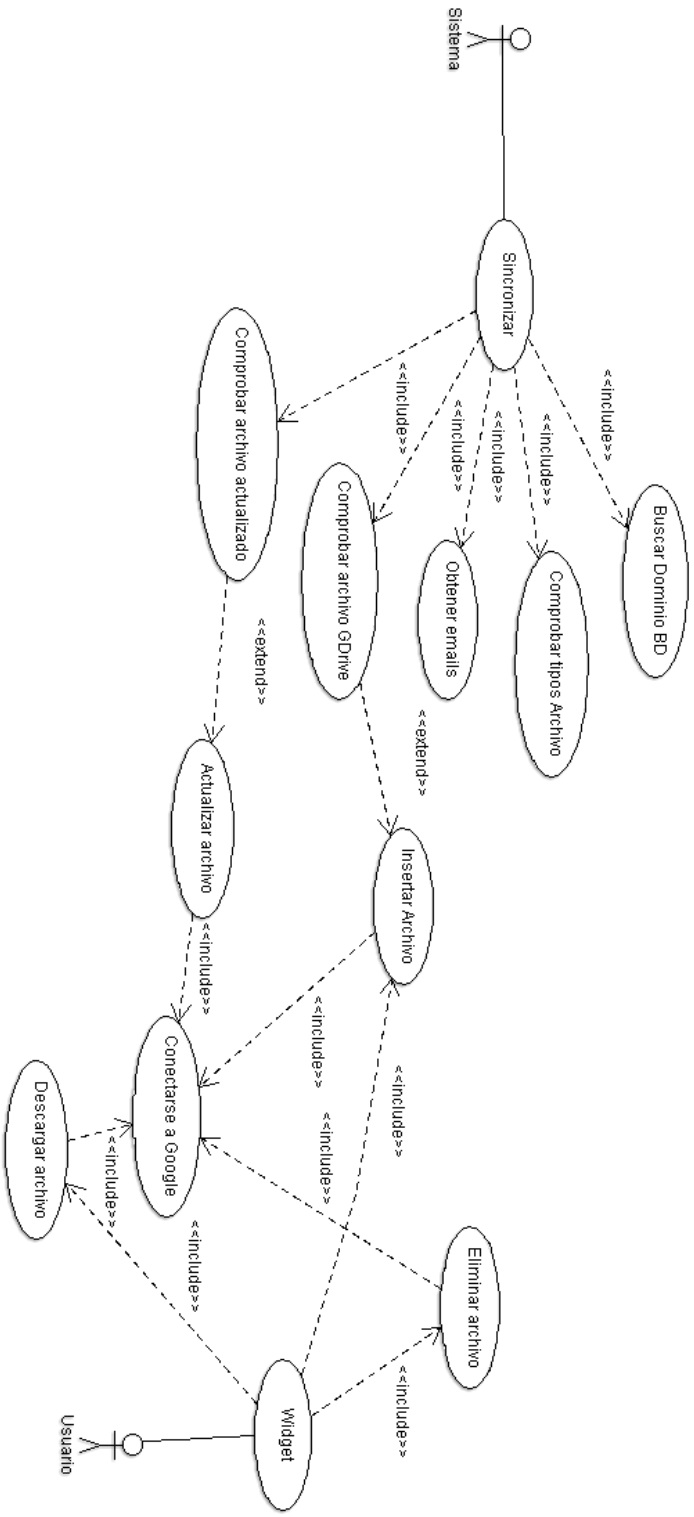
7.2. Captura de Requisitos

7.2.1. Requisitos

REFERENCIA	REQUISITOS
#1	Buscar todos los archivos de Google Drive.
#2	Buscar un archivo de Google Drive.
#3	Buscar todos los archivos del directorio raíz de Google Drive.
#4	Crear un archivo de Google Drive.
#5	Insertar un fichero en Google Drive.
#6	Actualizar un fichero de Google Drive.
#7	Dar los permisos pertinentes a un archivo subido a Google Drive.
#8	Eliminar un archivo de Google Drive.
#9	Sincronizar todos los archivos de AonSolutions a Google Drive.
#10	Buscar todos los dominios de la Base de datos.
#11	Buscar todos los ficheros del dominio dado, y también los ficheros de sus dominios hijo.
#12	Buscar los datos de conexión a Google (Service Account)
#13	Buscar un fichero específico de la Base de Datos.
#14	Buscar todos los emails, los cuales tengan acceso al archivo correspondiente. (por empresa)
#15	Buscar todos los emails de las personas dueñas del archivo correspondientes.(por persona)
#16	Añadir el identificador del archivo en Google Drive en la Base de datos.
#17	Obtener la palabra MD5 del archivo.
#18	Comprobar que el archivo de Google Drive y de la Base de datos está actualizado.
#19	Comprobar que tipos de archivos de la Base de datos hay que sincronizar con Google Drive.
#20	Generar un widget (interfaz gráfica) para Google Drive.
#21	Descargar un archivo de Google Drive.

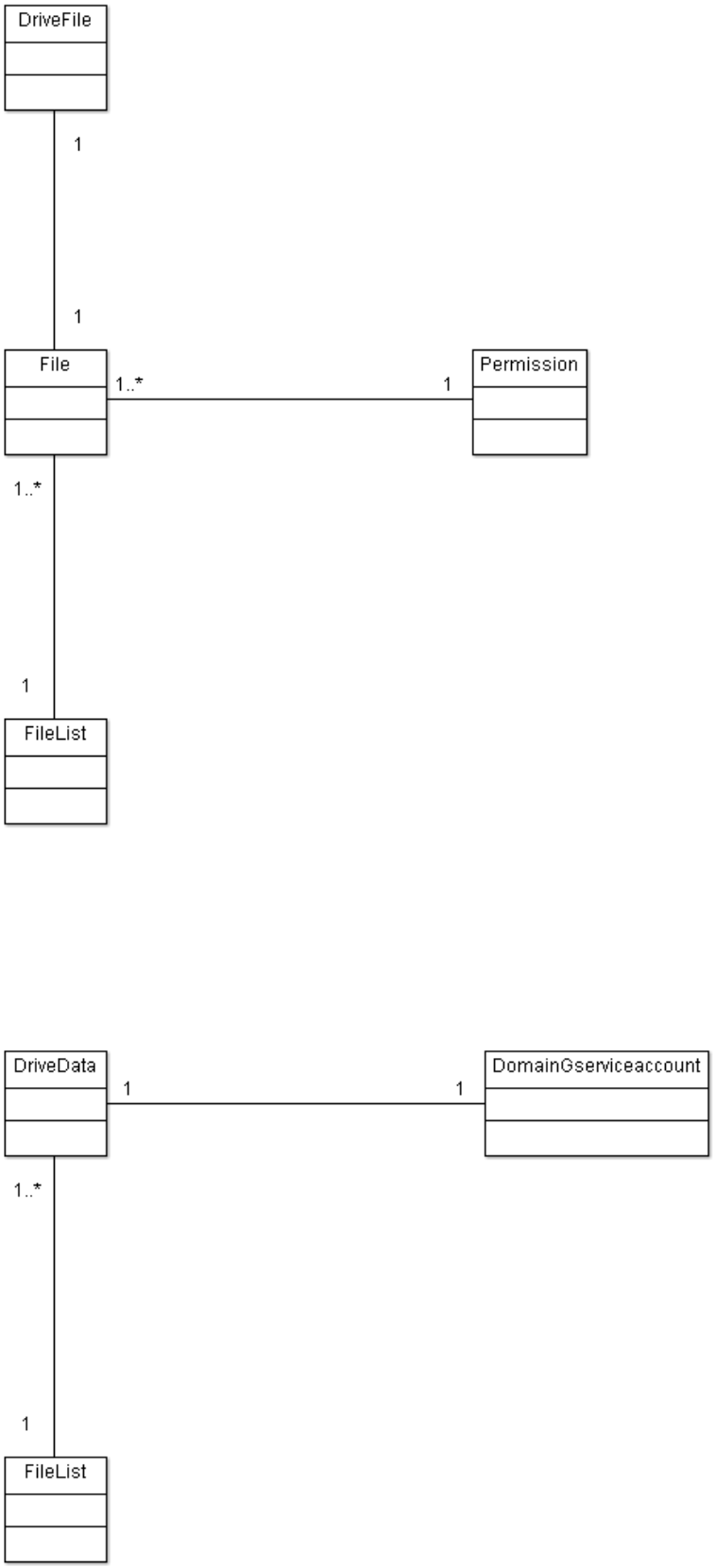
#22	Realizar la conexión con Google (OAuth 2)
#23	Comprobar si un archivo ya está en Google Drive.
#24	Eliminar el InputStrean del archivo de la Base de datos.
#25	Eliminar el identificador del archivo de Google Drive de la Base de datos.

7.2.2. Modelo de Casos de Uso



Imágen 7.7: Modelo de casos de uso de la iteración 3

7.2.3. Modelo de Dominio



Imágen 7.8: Modelo de dominio de la iteración 3

En la imagen 7.8 se puede observar el modelo de dominio de la cuarta iteración, en la parte superior se encuentran las clases u objetos pertenecientes a Google Drive API y en la parte inferior, las clases correspondientes a AonSolutions.

No se especifican ni los atributos ni las funciones de cada clase en el modelo de dominio de la iteración 4, para que no ocupe tanto espacio en la imagen y el diagrama se vea de manera más clara y sencilla. A continuación se especificarán tanto los atributos como las funciones de cada clase / objeto, únicamente las que se han utilizado para el desarrollo de la iteración, para no extenderse.

DriveFile

Clase auxiliar que contiene, la url del archivo en Google Drive, la descripción del archivo, el Mimetype y el identificador del archivo de Google Drive.

Atributos:

String **url**: La url para poder visualizar o descargar el archivo desde el navegador con Google Drive.

String **description**: Título del archivo.

String **mimetype**: Extensión MIME del tipo del archivo correspondiente.

String **id**: Identificador del archivo de Google Drive.

Funciones:

DriveFile(String id,String url, String description, String mimetype): Constructor de la clase DriveFile.

getUrl(): Devuelve la Url del archivo en Google Drive.

getDescription():Devuelve el título / descripción del archivo de Google Drive.

getMimetype(): Devuelve la extensión MIME del tipo del archivo de Google Drive.

getId(): Devuelve el identificador del archivo de Google Drive.

setUrl(String url): Inserta el valo de la Url del archivo en Google Drive.

setDescription(String description): Inserta el valor del título o descripción del archivo de Google Drive.

setMimetype(String mimetype): Inserta el valor de la ectensión del tipo del archivo de Google Drive.

setId(String id): Inserta el valor del identificador del archivo de Google Drive.

DriveData

Clase auxiliar que contiene los datos de conexión con Google y un vector de archivos, de la base de datos.

Atributos:

DomainGserviceaccount **Gservice**: Datos de Conexión para la cuenta de servicio de Google.

Vector<Rattach> **rattachs**: Vector de archivos de la base de datos.

String **domain**: Dominio, al que pertenecen los datos acceso a Google (conexión) y los archivos de la base de datos.

Funciones:

DriveData(DomainGserviceaccount Gservice,Vector<Rattach> rattachs): Constructor de la clase DriveData.

getGservice(): Devuelve los datos de conexión de la cuenta de servicio de Google.

getRattachs(): Devuelve el vector con los archivos de la base de datos.

getDomain(): Devuelve el dominio correspondiente a los datos de conexión y los archivos de la base de datos.

setGservice(DomainGserviceaccount Gservice): Inserta el valor de los datos de conexión de la cuenta de servicio de Google.

setRattachs(Vector<Rattach> rattachs): Inserta el valor del vector de archivos de la base de datos.

setDomain(String domain): Inserta el valor del dominio correspondiente a los datos de conexión de Google y de los archivos de la base de datos.

FileList

Corresponde una lista de archivos de Google Drive.

Atributos:

List<File> **ítems**: Lista de Ficheros.

Funciones:

FileList(): Constructor de la clase FileList.

getItems(): Devuelve una lista de Ficheros de Google Drive.

setItems(List<File>): Inserta una lista de ficheros de Google Drive.

File

Corresponde a un Fichero de Google Drive.

Atributos:

String **alternateLink**: Link para acceder a un archivo desde la interfaz gráfica de Google Drive.

String **description**: Descripción del archivo de Google Drive.

String **downloadUrl**: Url de descargar del archivo de Google Drive.

String **id**: Identificador del archivo.

String **md5Checksum**: Palabra de la reducción criptográfica de un archivo de Google Drive.

String **mimetype:** Extensión MIME del tipo de datos del archivo de Google Drive.

List<Permission> **permissions**: List de los permiso que dispone un archivo de Google Drive.

Boolean **shared**: Se puede compartir un archivo de Google Drive o no.

String **Title**: Titulo de un archivo de Google Drive.

Funciones:

File(): Constructor de la clase File.

getAlternateLink(): Devuelve el link para acceder a un archivo desde la interfaz gráfica de GooglDirve

getDescription(): Devuelve la descripción de un archivo de Google Drive.

getDownloadUrl(): Devuelve la url de descarga de un archivo de Google Drive.

getId(): Devuelve el identificador del archivo de Google Drive.

getMd5Checksum(): Devuelve la palabra de la reducción criptográfica del archivo de Gooogle Drive.

getMimetype(): Devuelve la extensión MIME del tipo de datos del archivo de Google Drive.

getPermissions(): Devuelve la lista de permisos que tiene un archivo de Google Drive.

getShared(): Devuelve true sii el archivo de Google Drive se puede compartir. False en caso contrario.

getTitle(): Devuelve el título del archivo de Google Drive.

setAlternateLink(String alternateLink): Inserta el valor del link para acceder a un archivo desde la interfaz gráfica de Google Drive.

setDescription(String description): Inserta el valor de la descripción de un archivo de Google Drive.

setDownloadUrl(String downloadUrl): Inserta el valor de la url de descar de un archivo de Google Drive.

setId(String id): Inserta el valor del identificador del archivo de Google Drive.

setMd5Checksum(String md5Checksum): Inserta el valor de la palabra de la reducción criptográfica del archivo de Googel Drive.

setMimetype(String mimetype): Inserta el valor de la extensión MIME del tipo de datos del archivo de Google Drive.

setPermissions(List<Permission> permissions): Inserta el valor de una lista de permisos para acceder al archivo de Google Drive.

setShared(boolean shared): Inserta el valor del booleano shared.

setTitle(String title): Inserta el valor del titulo del archivo de Google Drive.

Permission

Corresponde a los permisos que se le da a un archivo de Google Drive.

Atributos:

String **id**: Identificador del permiso.

String **role**: Rol que tendrá el usuario respecto al archivo.

String **type**: Tipo de la cuenta a la que le da permisos. (user , group , domain o anyone)

String **value**: Valor, dirección de correo electrónico o el dominio de la entidad. (owner , reader , writer , commenter).

Funciones:

permission(): Constructor de la clase Permission

getId(): Devuelve el identificador del permiso.

getRole(): Devuelve el role que desempeña el usuario en los permisos al archivo correspondiente de Google Drive.

getType(): Devuelve el tipo de la cuenta a la que se le da permisos para el archivo.

getValue(): Devuelve el valor de l permiso, ya sea el correo electrónico o el dominio de la entidad correspondiente.

setId(String id): Inserta el valor del identificador del permiso.

setRole(String role): Inserta el valor del rol que desempeña el usuario en los permisos al archivo correspondiente.

setType(String type): Inserta el valor del tipo de la cuenta a la que se le da permisos para el acceso al archivo.

setValue(String value): Inserta el valor del permiso, ya sea el correo electrónico o el dominio de la entidad correspondiente.

Rattach

Corresponde a los Ficheros de AonSolutions, de la base de datos.

Atributos:

Integer **id**: Identificador del archivo.

Short **mimetype**: Extensión MIME del tipo de datos del archivo.

String **description**: Descripción del archivo.

InputStream **data**: bytes del archivo.
Short **type**: Tipo del archivo en AonSolutions.

Integer **scope**: Ámbito del archivo.

String **driveId**: Identificador del archivo en Google Drive.

Funciones:

getId(): Devuelve el identificador del archivo.

getMimetype(): Devuelve la extensión MIME del tipo de datos del archivo.

getDescription(): Devuelve la descripción del archivo.

getData(): Devuelve los bytes del archivo.

getType(): Devuelve el tipo del archivo en AonSolutions.

getScope(): Devuelve el ámbito del archivo.

getDriveId(): Devuelve el identificador del archivo en Google Drive.

setId(Integer id): Inserta el valor del identificador del archivo.

setMimetype(Short mimetype): Inserta el valor de la extensión MIME del tipo de datos del archivo.

setDescription(String description): Inserta el valor de la descripción del archivo.

setData(InputStream data): Inserta el valor de los bytes del archivo.

setType(Short type): Inserta el valor del tipo del archivo en AonSolutions.

setScope(Integer scope): Inserta el valor del ámbito del archivo.

setDriveId(String driveId): Inserta el valor del identificador del archivo en Google Drive.

DomainGserviceaccount

Datos de conexión con Google, para una cuenta de servicio de la aplicación.

Atributos:

String **clientId**: Identificador del cliente de una cuenta de servicio de Google.

String **emailAdress**: Correo electrónico de una cuenta de servicio de Google.

String **publicKey**: Clave pública de una cuenta de servicio de Google.

InputStream **privateKey**: Clave privada de una cuenta de servicio de Google.

InputStream **clientSecret**: Archivo client_Secrets.

Funciones:

getClientId(): Devuelve el identificador del cliente de una cuenta de servicio de Google.

getEmailAdress(): Devuelve el correo electrónico de una cuenta de servicio de Google.

getPublicKey(): Devuelve la clave pública de una cuenta de servicio de Google.

getPrivateKet(): Devuelve la clave privada de una cuenta de servicio de Google.

getClientSecret(): Devuelve el archivo client_Secrets.

setClientId(String clientId): Inserta el valor el identificador del cliente de una cuenta de servicio de Google.

setEmailAdress(String emailAdress): Inserta el valor del correo electrónico de una cuenta de servicio de Google.

setPublicKey(String publicKey): Inserta el valor de la clave pública de una cuenta de servicio de Google.

setPrivateKey(InputStream privateKey): Inserta el valor de la clave privada de una cuenta de servicio de Google.

setClientSecret(InputStream clientSecret): Inserta el valor del archivo client_Secrets.

7.2.4. Casos de Uso

CU: Buscar dominio BD

Actores: Sistema

Descripción: El sistema busca todos los dominios existentes en la base de datos.

Precondiciones: --

Post-condiciones: --

Referencias: #10

Escenario Principal:
1. El sistema busca todos los dominios de la base de datos.

CU: Buscar archivos BD

Actores: Sistema

Descripción: El sistema busca todos los archivos de un dominio específico y los de sus dominios hijo, en el caso que el dominio disponga de una cuenta de servicio para la conexión con Google.

Precondiciones: Disponer de un dominio

Post-condiciones: --

Referencias: #11, #12

Escenario Principal:
1. El sistema busca los archivos de un dominio y sus hijos en la base de datos.
2. El sistema busca los datos de conexión de la cuenta de servicio de Google.

CU: Comprobar tipos de archivo

Actores: Sistema

Descripción: El sistema comprueba si el tipo del archivo es del tipo que se quiere sincronizar con Google Drive, dependiendo de las necesidades o preferencias del cliente correspondiente.

Precondiciones: Disponer de un vector de RegistryAttachmentType y un archivo de la base de datos.

Post-condiciones: --

Referencias: #19

Escenario Principal:
1. El cliente o el sistema establece un vector de RegistryAttachmentType.
2. El sistema comprueba si el archivo es de alguno de los tipos a sincronizar.

CU: Obtener emails

Actores: Sistema

Descripción: El sistema busca todos los correos electrónicos que estén relacionado con un archivo de la base de datos, para posteriormente darle permisos al archivo de Google Drive.

Precondiciones: Disponer del identificador del archivo y del dominio al que pertenezca el archivo.

Post-condiciones: --

Referencias: #14, #15

Escenario Principal:
1. El sistema busca todos los emails, los cuales tengan acceso al archivo correspondiente. (por empresa).
2. El sistema busca todos los emails de las personas dueñas del archivo correspondientes (por persona).

CU: Comprobar archivo en Google Drive

Actores: Sistema

Descripción: El sistema comprueba si el archivo está en Google drive, mediante el atributo drive_id del archivo de la base de datos que es nulo si no está y sino si que ha sido sincronizado con Google Drive.

Precondiciones: Disponer de un archivo de la base de datos.

Post-condiciones: --

Referencias: #23

Escenario Principal:
1. El sistema comprueba si el archivo de la base de datos está en Google.

CU: Insertar archivo

Actores: Sistema

Descripción: El sistema crea un nuevo archivo de Google Drive, partiendo del archivo de la base de datos, se conecta a Google, e inserta el archivo en la cuenta de Google Drive. Luego en el caso de que se trate de una cuenta de servicio da los permisos pertinentes a los usuarios y añade a la base de datos, en la tabla de archivos el identificador del archivo en Google Drive. Al final borra el campo data de la tabla rattach de la base de datos, para liberar espacio en la base de datos.

Precondiciones: Disponer de un archivo de la base de datos y de la lista de emails que están relacionados con el archivo en cuestión.

Post-condiciones: El archivo ya esta disponible en Google Drive (, ha sido compartido con los usuarios que tenían acceso al archivo, en la tabla rattach de la base de datos se ha añadido el campo drive_id y el campo data (Inputstream) se ha eliminado).

Referencias: #4, #5, #7, #16, #20, #24

Escenario Principal:
1. El sistema crea un nuevo archivo de Google Drive a partir del archivo de la base de datos.
2. El sistema realiza la conexión con Google.
3. El sistema inserta el archivo en la cuenta correspondiente de Google Drive.

(Si es una cuenta de servicio)
4. El sistema crea e inserta los permisos necesarios para el archivo correspondiente en Google Drive.
5. Inserta el identificador del archivo en Google Drive en el parámetro drive_id de la tabla rattach de la base de datos.
6. Elimina de la base de datos, el campo data de la tabla rattach.

CU: Comprobar archivo actualizado

Actores: Sistema

Descripción: El sistema comprueba si un archivo esta actualizado, es decir, si coincide el archivo de la base de datos con el que está en Google Drive, para ello se utiliza la palabra Md5.

Precondiciones: El archivo esta en ambos sitios, la base de datos y Google Drive. Disponer del archivo de la base de datos

Post-condiciones: --

Referencias: #2, #17, #18

Escenario Principal:
1. El sistema busca el archivo en Google Drive.
2. El sistema genera la palabra md5 del archivo de la base de datos.
3. El sistema compara las palabras md5 de los dos archivos.

CU: Actualizar archivo

Actores: Sistema

Descripción: El sistema crea un archivo de Google Drive a partir del archivo de la base de datos, se conecta con Google y Actualiza el archivo correspondiente en Google Drive. Al final borra el campo data de la tabla rattach de la base de datos, para liberar espacio en la base de datos.

Precondiciones: Disponer de un archivo de la base de datos.

Post-condiciones: El archivo se queda actualizado en Google Drive, y el campo data de la tabla rattach de la base de datos se pone a null.

Referencias: #4, #6, #20, #24

Escenario Principal:

1. El sistema crea un archivo de Google Drive a partir del archivo dado.
2. El sistema realiza la conexión con Google.
3. El sistema actualiza el archivo en Google Drive.
4. El sistema borra el campo data de la tabla rattach de la base de datos y lo poner a null.

CU: Eliminar archivo

Actores: Sistema

Descripción: El sistema se conecta con Google y elimina el archivo de Google Drive correspondiente (el especificado con su identificador).Al final elimina el campo drive_id de la tabla rattach de la base de datos.

Precondiciones: Disponer del identificador del archivo de Google Drive a eliminar.

Post-condiciones: El archivo ya no se encuentra accesible desde Google Calendar y el campo drive_id esta como null en la tabla rattach de la base de datos.

Referencias: #8, #20

Escenario Principal:

1. El sistema se conecta con Google.
2. El sistema elimina el archivo de Google Drive.

Si la cuenta es de servicio

3. El sistema elimina de la base de datos el campo drive_id de la tabla rattach.

CU: Descargar archivo

Actores: Sistema

Descripción: El sistema se conecta a Google con la intención de descargar el archivo (InputStream) de Google Drive.

Precondiciones: Disponer del Fichero de Google Drive.

Post-condiciones: --

Referencias: #20, #22

Escenario Principal:

1. El sistema se conecta a Google.
2. El sistema descarga el archivo en InputStream.

CU: Sincronizar

Actores: Sistema

Descripción: El sistema sincroniza los archivos de AonSolutions del dominio correspondiente con la cuenta de servicio de la aplicación. Para ello el sistema tiene que realizar determinadas funciones que corresponde a la mayoría de los casos de uso detallados con anterioridad. *Buscar dominios BD, Buscar archivos BD, Comprobar tipos, Obtener emails, Comprobar archivo en Google, Insertar archivo, Comprobar archivo actualizado y Actualizar archivo.*

Precondiciones: Necesita los tipos de archivo de AonSolutions que hay que sincronizar con Google Drive (y el dominio).

Post-condiciones: Todas las archivos que hay que subir ya están sincronizados con Google Drive.

Referencias: #2, #4, #5, #6, #7, #9, #10, #11, #12, #13, #14, #15, #16, #17, #18, #19, #20, #23, #24, #25

Escenario Principal:

1. El sistema sistema busca todos los dominios de la base de datos (Si disponemos del dominio, se salta al paso 2).

Para todos los dominios:

2. El sistema busca todos los ficheros a sincronizar con Google Drive.

Para todos los ficheros:

3. El sistema comprueba si el fichero es de alguno de los tipos de los que hay que sincronizar. Si no hay que sincronizarlo, no hace más pasos.
4. El sistema busca todos los emails que estén relacionados con el susodicho fichero.
5. El sistema comprueba si el fichero esta o no en Google Drive.
    a. Si no esta en Google Drive Inserta el archivo correspondiente en Google Drive.
    b. Si está, comprueba que está actualizado y si no lo esta actualiza el archivo en Google Drive.

CU: Widget

Actores: Sistema

Descripción: En este widget aparecerán los archivos correspondientes a la cuenta personal del usuario. Con los botones de subir un nuevo archivo a Google Drive, descargar un archivo de Google Drive y borrar un archivo de Google Drive.

Precondiciones: --

Post-condiciones: --

Referencias: #3, #5, #8, #16, #20, #22

Escenario Principal:

1. El usuario pulsa el botón de subir un nuevo archivo.
2. El sistema se conecta a Google.
3. El sistema inserta el nuevo archivo a Google Drive.

Escenario Alternativo 1:

1. El usuario pulsa el botón de descargar un Archivo.
2. El sistema se conecta a Google.
3. El sistema se descarga el archivo correspondiente.

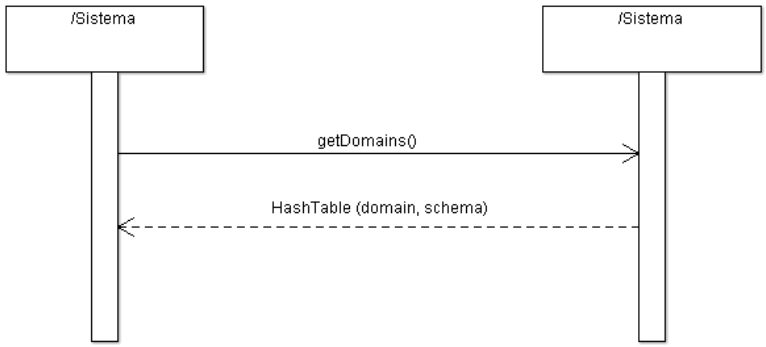
Escenario Alternativo 2:

1. El usuario pulsa el botón de borrar un archivo.
2. El sistema se conecta a Google.
3. El sistema borra el archivo de Google Drive.

7.3. Análisis

CU: Buscar dominio BD

Diagrama de secuencia del sistema



Imágen 7.9: Diagrama de secuencia del sistema, buscar dominio BD

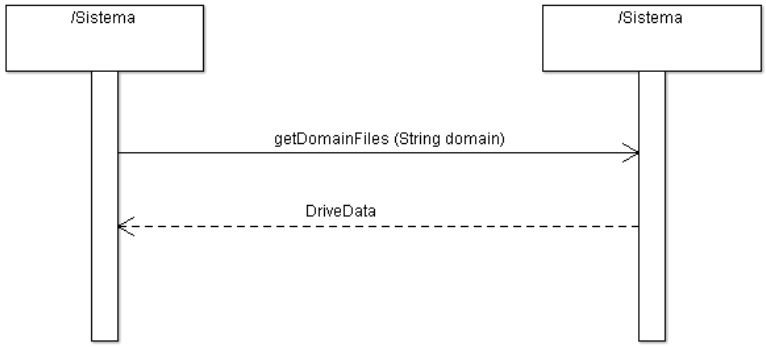
Contratos

Contrato 1:

- Nombre: getDomains()
- Responsabilidades: El sistema busca todos los dominios que dispone la base de datos.
- Excepciones: --
- Precondiciones: --
- Post-condiciones: --
- Salida: HashTable (domain, schema)

CU: Buscar archivos BD

Diagrama de secuencia del sistema



Imágen 7.10: Diagrama de secuencia del sistema, buscar archivos BD

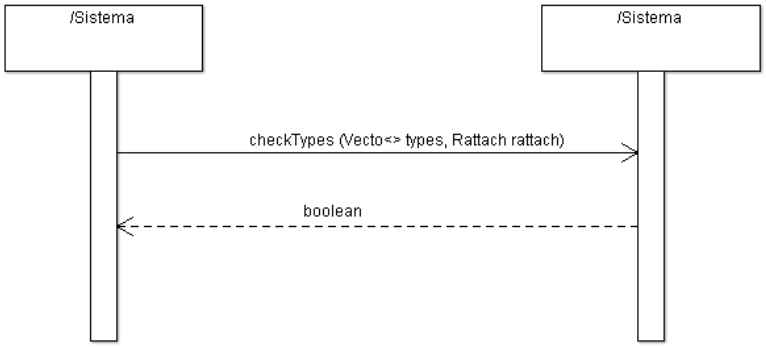
Contratos

Contrato 1:

- Nombre: getDomainFiles(String domain)
- Responsabilidades: Busca todos los archivos de la base de datos, correspondientes al dominio indicado y si dicho dominio dispone de cuenta de servicio para conectarse con Google.
- Excepciones: --
- Precondiciones: Disponer de un dominio.
- Post-condiciones: --
- Salida: DriveData.

CU: Comprobar tipos de archivo

Diagrama de secuencia del sistema



Imágen 7.11: Diagrama de secuencia del sistema, comprobar tipos

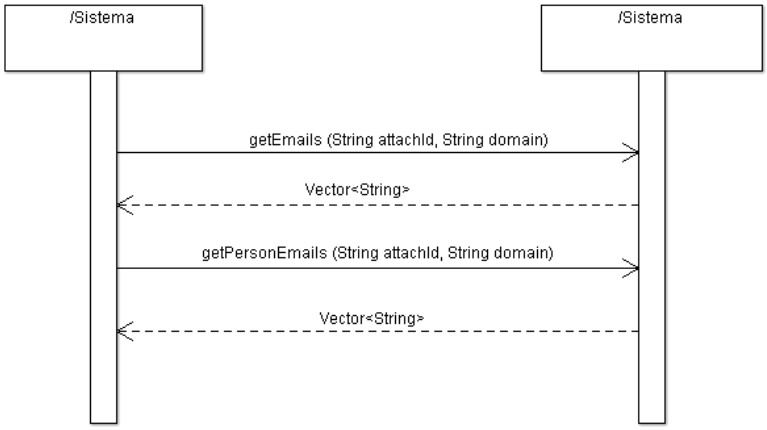
Contratos

Contrato 1:

- Nombre: checkTypes(Vector<> types, Rattach rattach)
- Responsabilidades: Comprueba si el archivo es del tipo correspondiente a los tipos que se tienen que sincronizar con Google Drive o no.
- Excepciones: --
- Precondiciones: Disponer del vecto de RegistryAttachmentType y de un archivo.
- Post-condiciones: --
- Salida: boolean

CU: Obtener emails

Diagrama de secuencia del sistema



Imágen 7.11: Diagrama de secuencia del sistema, obtener emails

Contratos

Contrato 1:

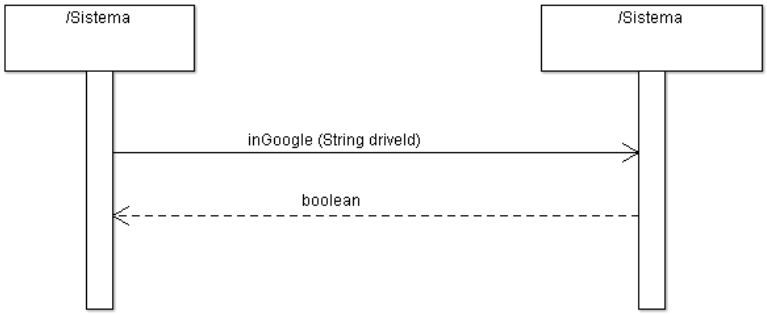
- Nombre: getEmails(String attachId,String domain)
- Responsabilidades: Busca todos los emails que estén relacionados con el archivo por empresa.
- Excepciones: --
- Precondiciones: Disponer del identificador del archivo y del dominio al que pertenece.
- Post-condiciones: --
- Salida: Vector<String>

Contrato 2:

- Nombre: getPersonEmails(String attachId,String domain)
- Responsabilidades: Buscar todos los emails que estén relacionados con el archivo por persona.
- Excepciones: --
- Precondiciones: Disponer del identificador del archivo y del dominio al que pertenece.
- Post-condiciones: --
- Salida: Vector<String>

CU: Comprobar archivo en Google Drive

Diagrama de secuencia del sistema



Imágen 7.12: Diagrama de secuencia del sistema, comprobar archivos Google

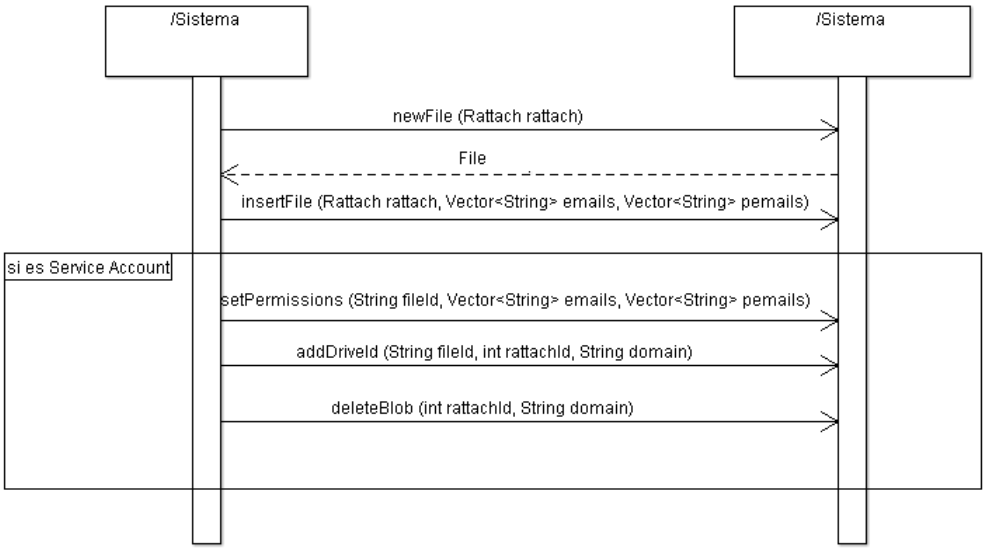
Contratos

Contrato 1:

- Nombre: inGoogle()
- Responsabilidades:
- Excepciones: --
- Precondiciones: Disponer del identificador del archivo de Google Drive.
- Post-condiciones: --
- Salida: Boolean

CU: Insertar archivo

Diagrama de secuencia del sistema



Imágen 7.13: Diagrama de secuencia del sistema, insertar archivo

Contratos

Contrato 1:

- Nombre: newFile(Rattach rattach)
- Responsabilidades: Crea un nuevo archivo de Google Drive a partir del archivo de la base de datos dado.
- Excepciones: --
- Precondiciones: Disponer de un archivo de la base de datos.
- Post-condiciones: --
- Salida: File

Contrato 2:

- Nombre: insertFile(Rattach rattach | Vector<String> emails, Vector<Sting> pemails)
- Responsabilidades: Se conecta a Google e inserta el archivo dado a Google Drive. Llama a las funciones, newFile, y setPermissions.
- Excepciones: --
- Precondiciones: Disponer de un archivo
- Post-condiciones: El archivo ya se encuentra disponible en en la cuenta correspondiente de Google Drive.
- Salida: --

Contrato 3:

- Nombre: setPerminssions(String fileId, Vector<String> emails, Vector<Sting> pemails)
- Responsabilidades: Se encarga de crear e insertar los permisos de las cuentas dadas en los vectores de emails al archivo correspondiente.
- Excepciones: --

- Precondiciones: Es necesario disponer del identificador del archivo de Google Drive y 2 vectores de emails.
- Post-condiciones: El archivo ya dispone de los permisos.
- Salida: --

Contrato 4:

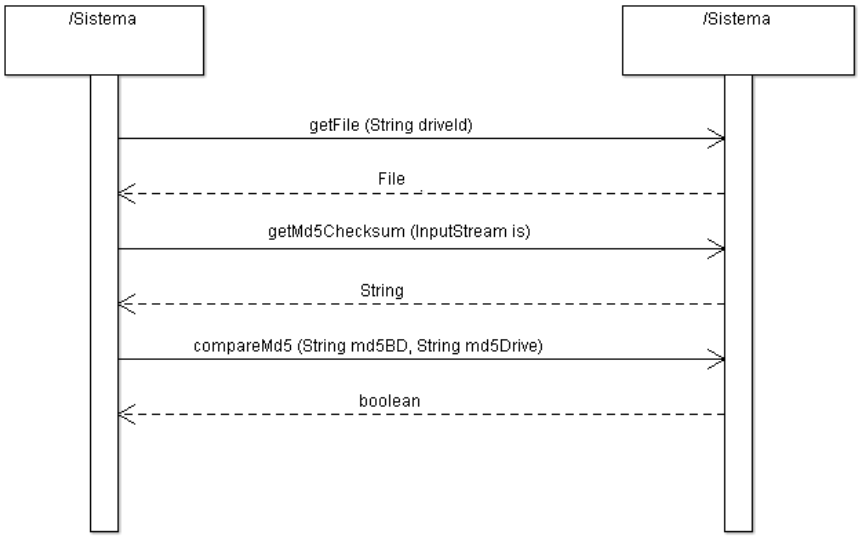
- Nombre: addDriveId(String fileId,int rattachId, String domain)
- Responsabilidades: Se añade el atributo drive_id a la tabla rattach de la base de datos.
- Excepciones: --
- Precondiciones: Disponer del identificador del fichero de la base de datos, el identificador del archivo de Google Drive u el dominio al que pertenece el archivo.
- Post-condiciones: Ya se encuentra en el campo drive_id de la tabla rattach de la base de datos con el valor del identificador del archivo de Google Drive.
- Salida: --

Contrato 5:

- Nombre: deleteBlob(int rattachId, String domain
- Responsabilidades: Borra el campo data de la tabla rattach de la base de datos.
- Excepciones: --
- Precondiciones: Disponer del identificador del archivo de la base de datos y su dominio correspondiente.
- Post-condiciones: El campo data de la tabla rattach de la base de datos es null.
- Salida: --

CU: Comprobar archivo actualizado

Diagrama de secuencia del sistema



Imágen 7.14: Diagrama de secuencia del sistema, comprobar archivos actualizado

Contratos

Contrato 1:

- Nombre: getFile(String driveId)
- Responsabilidades: Busca el archivo en Google Drive, el cual coincida con el driveId dado.
- Excepciones: --
- Precondiciones: Disponer del identificador del archivo de Google Drive
- Post-condiciones: --
- Salida: File

Contrato 2:

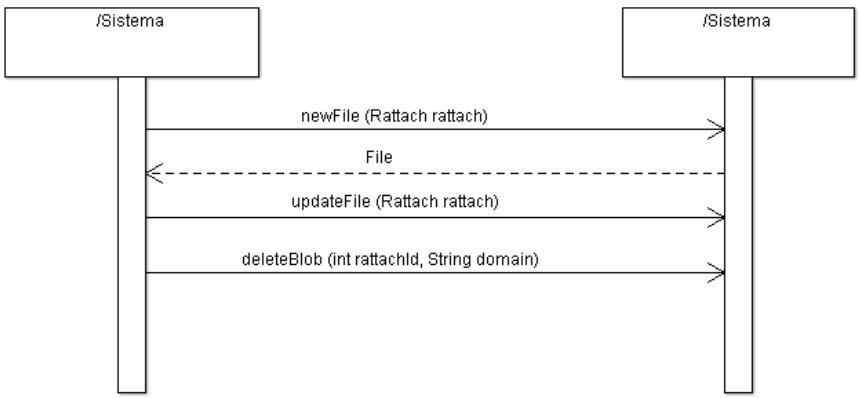
- Nombre: getMd5Checksum(InputStream is)
- Responsabilidades: Obtiene la palabra md5 del archivo introducido is.
- Excepciones: --
- Precondiciones: Disponer de un archivo en tipo InputStream.
- Post-condiciones: --
- Salida: String

Contrato 3:

- Nombre: compareMd5(String md5BD, md5Drive)
- Responsabilidades: Compara la palabra md5 del archivo de Google Drive con l palabra md5 del archivo de la base de datos.
- Excepciones: --
- Precondiciones: Disponer de las dos palabras md5.
- Post-condiciones: --
- Salida: boolean

CU: Actualizar archivo

Diagrama de secuencia del sistema



Imágen 7.15: Diagrama de secuencia del sistema, actualizar archivo

Contratos

Contrato 1:

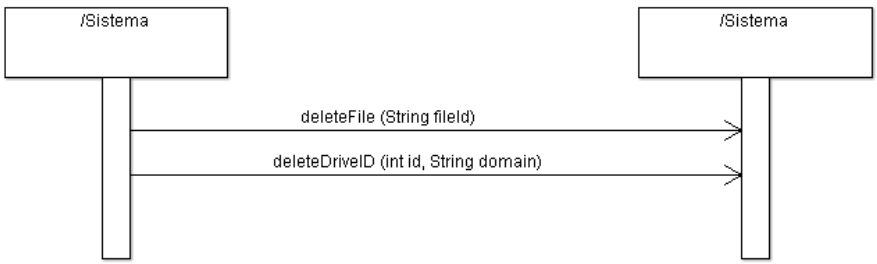
- Nombre: newFile(Rattach rattach)
- Responsabilidades: Crea un nuevo archivo de Google Drive a partir del archivo de la base de datos dado.
- Excepciones: --
- Precondiciones: Disponer de un archivo de la base de datos.
- Post-condiciones: --
- Salida: File

Contrato 2:

- Nombre: updateFile(Rattach rattach)
- Responsabilidades: Ejecuta la función newFile(rattach) y con la archivo que le devuelve actualiza el archivo en Google Drive.
- Excepciones: --
- Precondiciones: Disponer de un archivo de la base de datos.
- Post-condiciones: El archivo ya se encuentra actualizado en Google Drive.
- Salida: --

CU: Eliminar archivo

Diagrama de secuencia del sistema



Imágen 7.16: Diagrama de secuencia del sistema, Eliminar archivo

Contratos

Contrato 1:

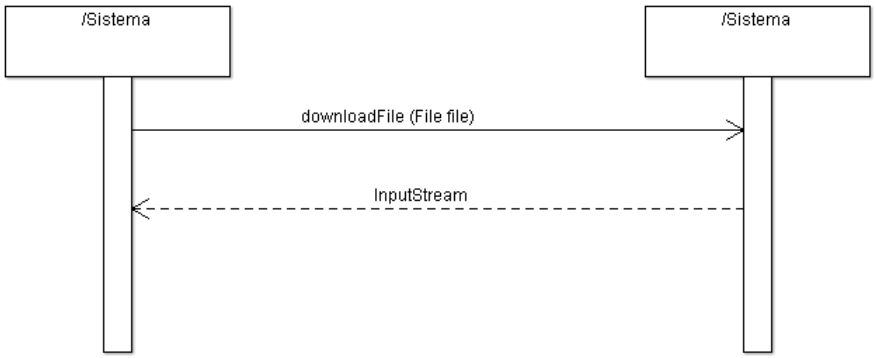
- Nombre: deleteFile(String fileId)
- Responsabilidades: Se conecta a Google y elimina el archivo correspondiente de Google Drive.
- Excepciones: --
- Precondiciones: Disponer del identificador del archivo de Google Drive.
- Post-condiciones: El archivo ya está eliminado en Google Drive y no se puede acceder a él.
- Salida: --

Contrato 2:

- Nombre: deleteDriveID(int id, String domain)
- Responsabilidades: Elimina o pone a null el campo de la tabla rattach de la base de datos para dicho archivo.
- Excepciones: --
- Precondiciones: Disponer del identificador del archivo de la base de datos y el dominio al que pertenece el archivo.
- Post-condiciones: El el campo drive_id de la tabla rattach de la base de datos es nul.
- Salida: --

CU: Descargar archivo

Diagrama de secuencia del sistema



Imágen 7.17: Diagrama de secuencia del sistema, Descargar archivo

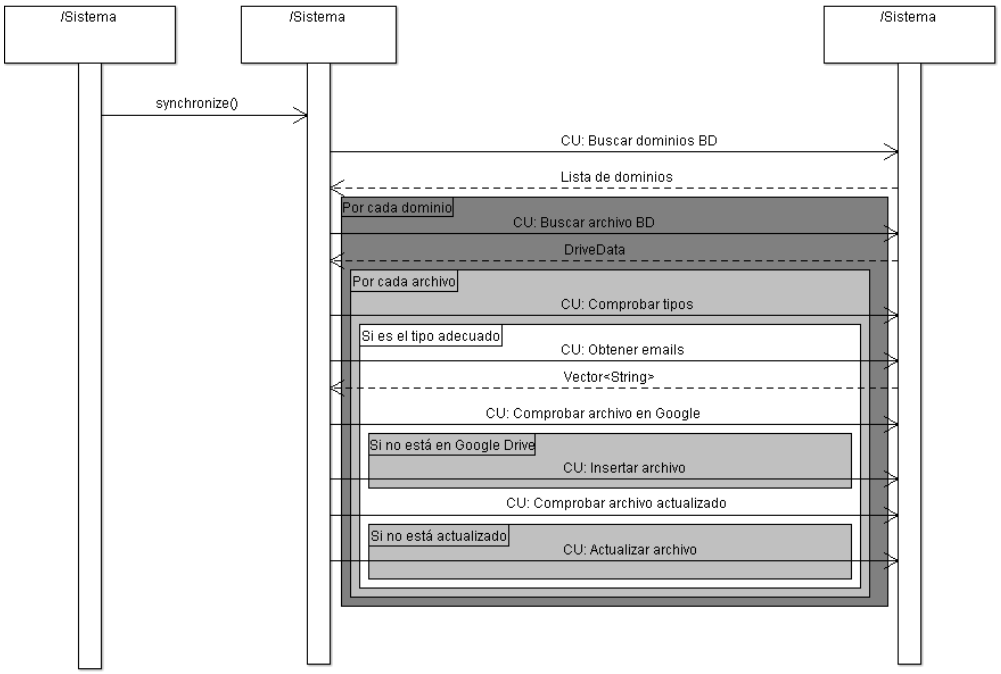
Contratos

Contrato 1:

- Nombre: downloadFile(File file)
- Responsabilidades: Se conecta a Google y descarga el archivo en InputStrem y lo devuelve.
- Excepciones: --
- Precondiciones: Disponer de un archivo de Google Calendar.
- Post-condiciones: --
- Salida: InputStream

CU: Sincronizar

Diagrama de secuencia del sistema



Imágen 7.18: Diagrama de secuencia del sistema, sincronizar

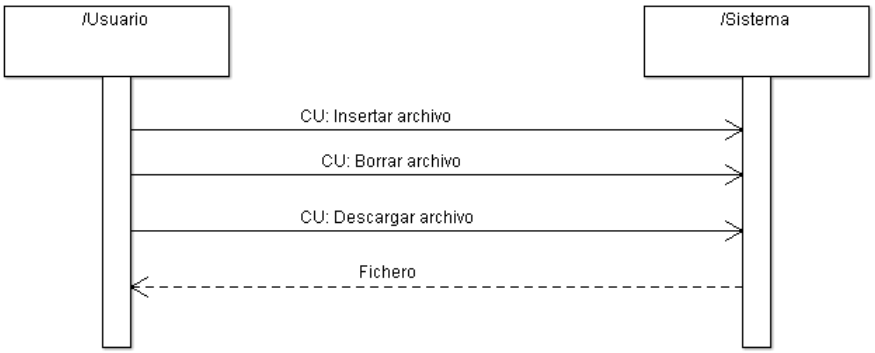
Contratos

Contrato 1:

- Nombre: synchronize()
- Responsabilidades: El sistema sincroniza todos los archivos de AonSolutions del dominio con Google Drive.
- Excepciones: --
- Precondiciones: --
- Post-condiciones: Todos los archivos de AonSolutions de un dominio ya están sincronizados con Google Drive.
- Salida: --

CU: Widget

Diagrama de secuencia del sistema



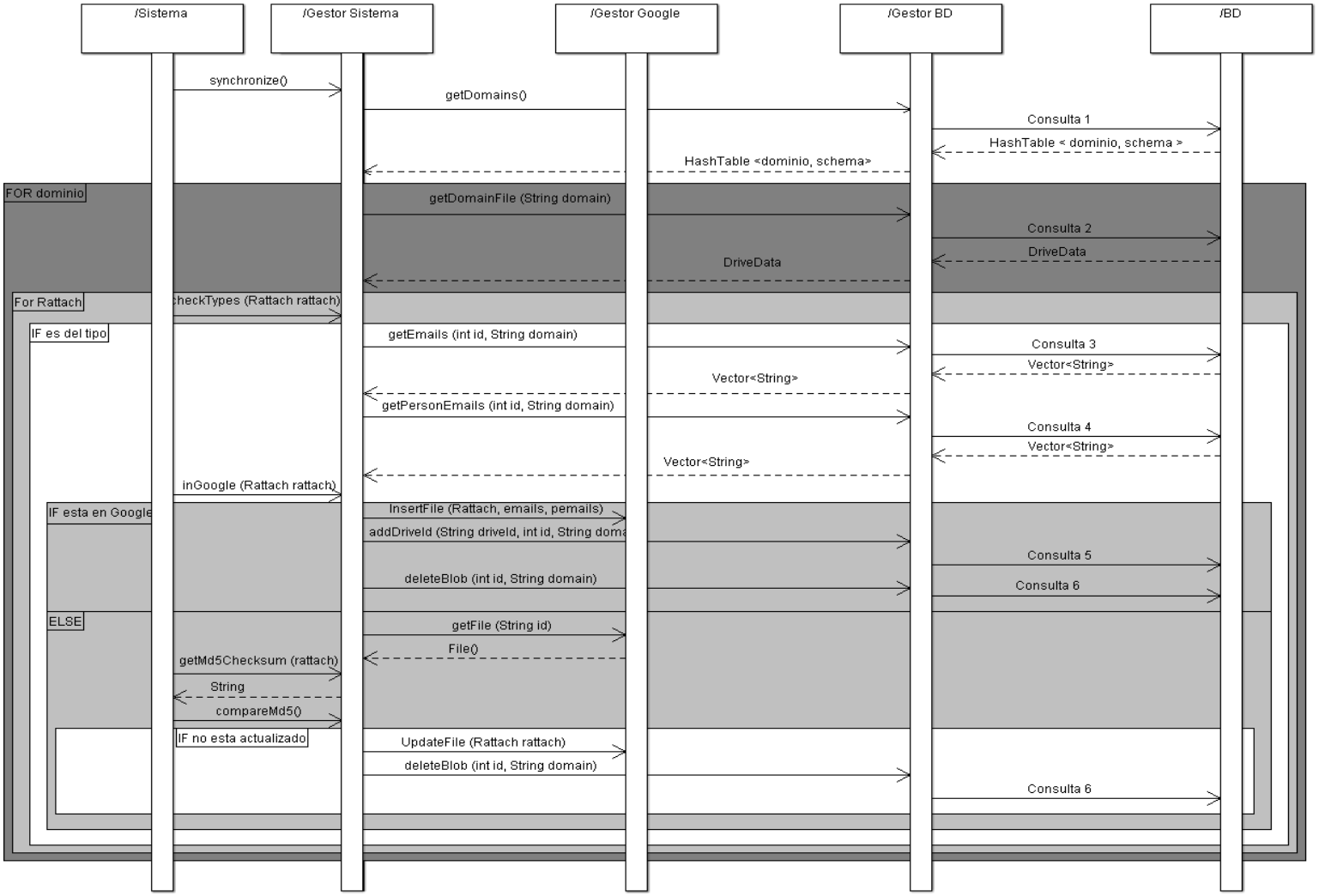
Imágen 7.19: Diagrama de secuencia del sistema, widget

7.4. Diseño

Para reducir el número de diagramas de secuencia en este documento se ha decidido realizar el diagrama de secuencia de la mayoría de los casos de uso en uno, precisamente el del caso de uso sincronizar, ya que abarca la gran mayoría de los casos de uso de ésta iteración.

Por otra parte también se añadirán los diagramas de secuencia de los casos de uso *Insertar archivo, Actualizar archivo, Eliminar archivo, Descargar archivo* y *Widget*.

CU: Sincronizar



Imágen 7.20: Diagrama de secuencia, sincronizar

Consulta 1

ConnectionInfo connectionInfo = ConnectionInfo.getDefaultConnectionInfo().getDomains().

Consulta 2

```
SELECT RA.id, RA.mimetype, RA.description, RA.type, RA.drive_id, G.*, D.name

FROM (rattach AS RA INNER  JOIN domain AS D ON RA.domain = D.id) INNER  JOIN domain_Gserviceaccount AS G ON (G.domain = D.id
        OR  D.parent = G.domain)

WHERE D.name = &DOMAIN OR D.parent IN (SELECT id

                                                FROM domain

                                                WHERE name = &DOMAIN
```

Consulta 3

```
SELECT DISTINCT MA.email

FROM rattach AS R INNER  JOIN enterprise AS E ON (E.registry = R.registry) INNER  JOIN mail_account  AS MA ON (MA.domain =
        E.domain) INNER  JOIN user_scope AS US ON (US.user_id =MA.user_id)

WHERE R.id = &ID  AND (R.scope  IS NULL OR R.scope = US.scope)
```

Consulta 4

```
SELECT DISTINCT RM.value

FROM rattach AS R INNER  JOIN person AS P ON ( P.registry = R.registry) INNER  JOIN rmedia  AS RM ON ( RM.registry = P.registry)

WHERE R.id = &ID AND RM.media = 4
```

Consulta 5

UPDATE rattach

SET drive_id = &DRIVEID

WHERE id = &ID

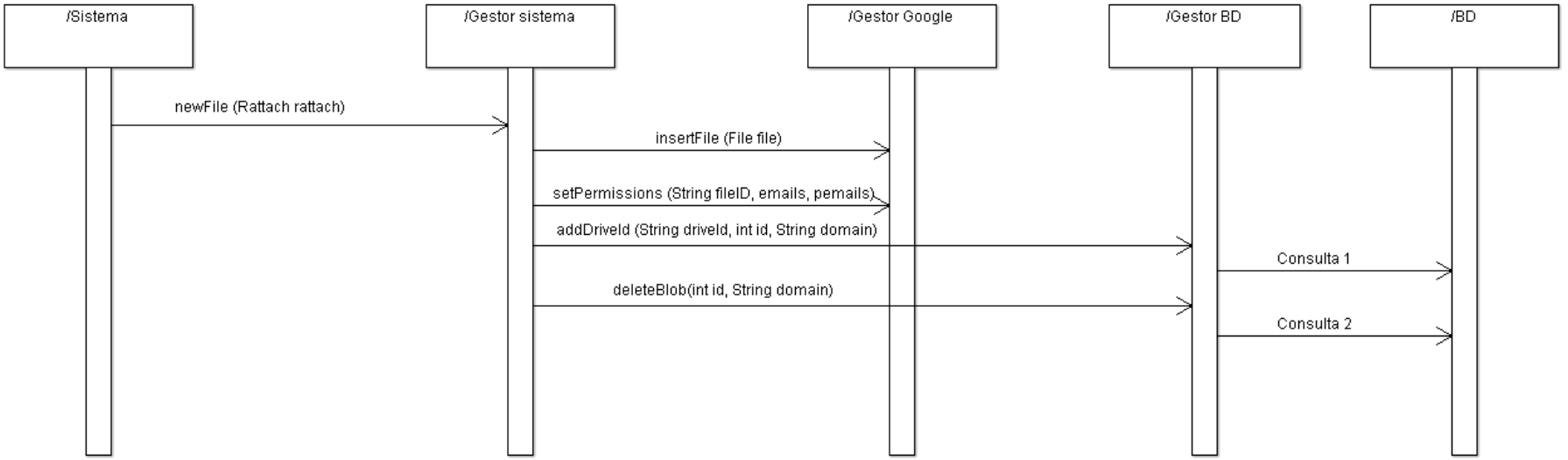
Consulta 6

UPDATE rattach

SET data = null

WHERE id = &ID

CU: Insertar archivo



Imágen 7.21: Diagrama de secuencia, insertar archivo

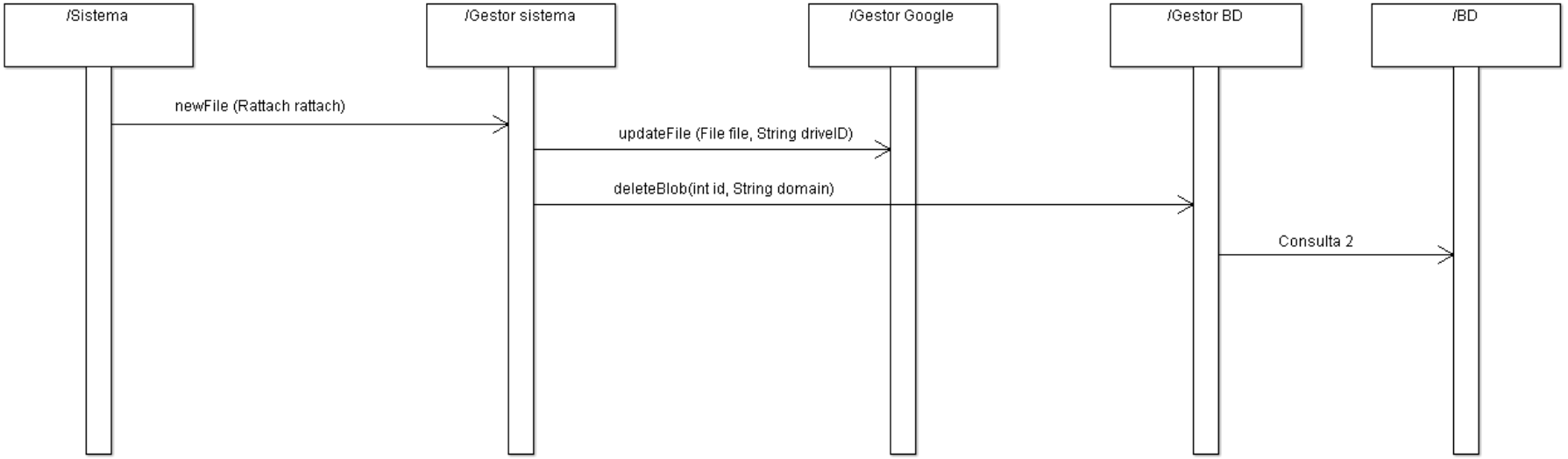
Consulta 1

Corresponde a la consulta 5 del anterior diagrama de secuencia (sincronizar)

Consulta 2

Corresponde a la consulta 6 del anterior diagrama de secuencia (sincronizar)

CU: Actualizar archivo

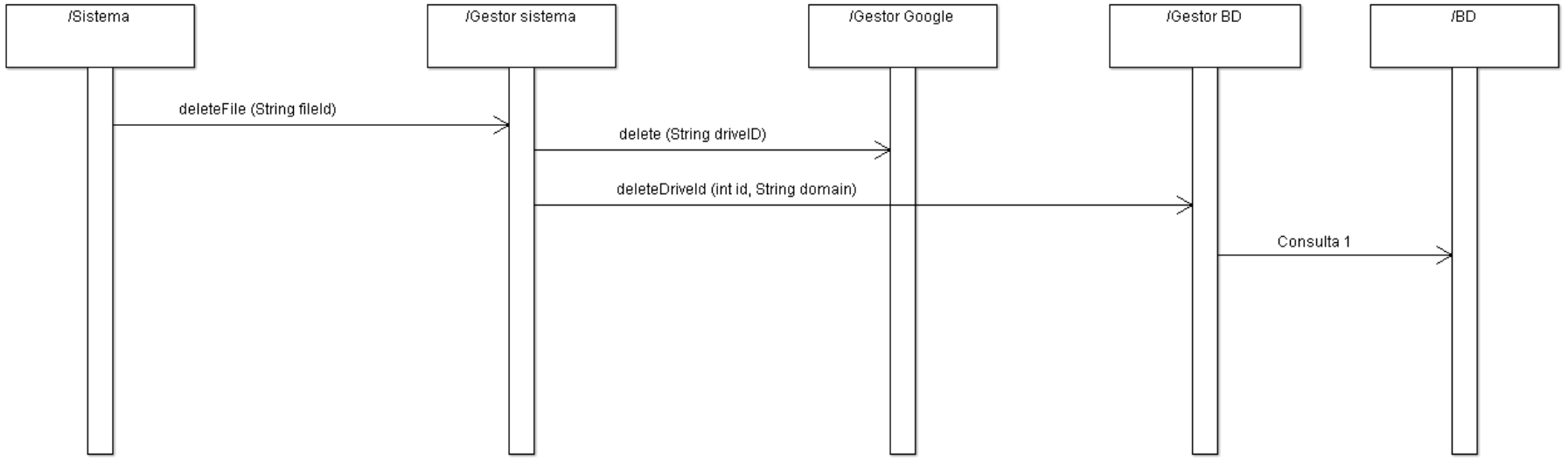


Imágen 7.22: Diagrama de secuencia, actualizar archivo

Consulta 2

Corresponde a la consulta 6 del diagrama de secuencia del caso de uso sincronizar.

CU: Eliminar archivo



Imágen 7.23: Diagrama de secuencia, Eliminar archivo

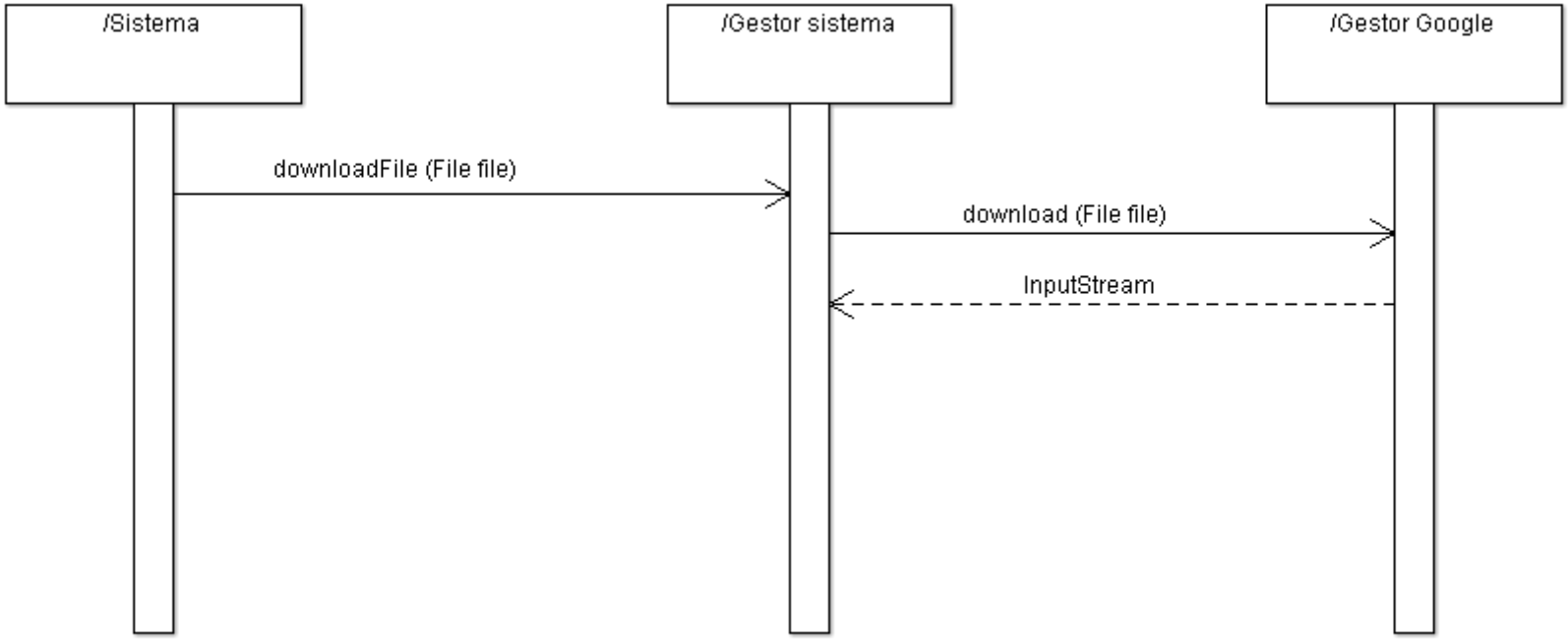
Consulta 1

UPDATE rattach

SET drive_id = null

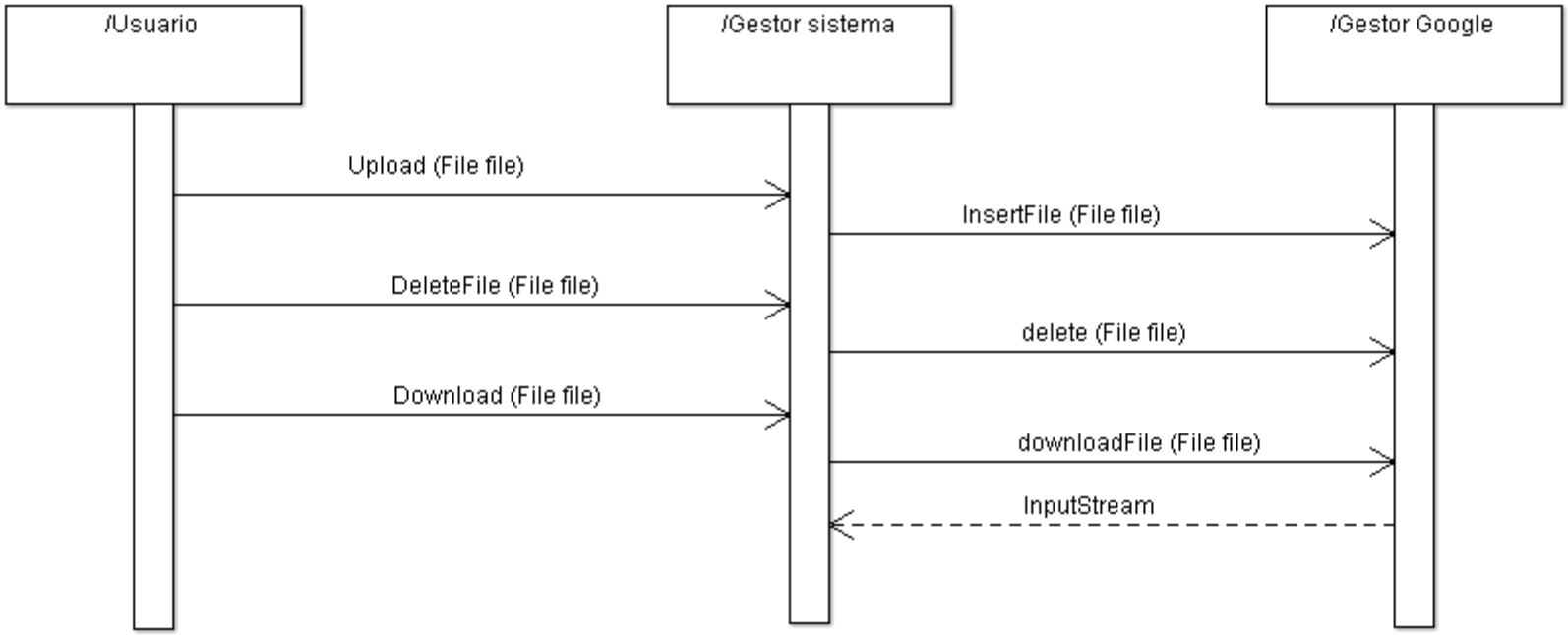
WHERE id = &ID

CU: Descargar archivo



Imágen 7.24: Diagrama de secuencia, descargar archivo

CU: Widget



Imágen 7.25: Diagrama de secuencia, widget

Las fases de implementación, pruebas e implantación de ésta iteración se detallan en los apartados 8, 9 y 10 respectivamente. Donde se tratarán dentro de cada uno de los apartados, cada una de las fases de las cuatro iteraciones del proyecto.

8. IMPLEMENTACIÓN

Para la realización del proyecto se ha creado un nuevo proyecto que será incluido en la plataforma de gestión empresarial AonSolutions. Este proyecto es aon-google-apis. Donde se implementarán las funciones fundamentales de las aplicaciones de Google que se desarrollarán para el proyecto (Calendar, Task, Drive y Sign in with Google).

Además se utilizará el proyecto aon-google-sql, que se genera automáticamente con las tablas de la base de datos que se necesiten para cada aplicación. Donde cada tabla utilizada se convierte en una clase. El proyecto que genera automáticamente las clases, ya se disponía en la empresa.

Por último, a lo largo de las 4 iteraciones también se implementará algo de código en otros proyectos ya existentes en la plataforma de gestión empresarial AonSolutions. Dichos proyectos, se irán especificando en cada una de las iteraciones de la aplicación, junto con los archivos donde se desarrollará el código en cada proyecto.

Los proyectos utilizados a lo largo de todo el proceso de implementación en la aplicación son los siguientes:

- Aon-google-apis: Código principal de las cuatro iteraciones del proyecto.
- Aon-sql-google: Código, que contiene las clases de la base de datos.
- Aon-web-aio: Proyecto principal de la aplicación AonSolutions1.
- Aon-ui-commercial: Código correspondiente a la agenda comercial de AonSolutions.
- Aon-ui-resources: Contiene los iconos que se necesitan para lanzar la interfaz gráfica de la aplicación.
- Aon-jaas: Contiene el código correspondiente para el Login de AonSolutions.
- Aon-ui-registry: Contiene el código correspondiente a los archivos de AonSolutions.

Para la conexión con Google se utilizarán el proyecto de principal de éste de proyecto de fin de carrera, aon-google-apis. Los archivos correspondientes para dicha conexión con Google se encuentran en el paquete aon-google-apis-servlets del proyecto.

Los siguientes archivos son los correspondientes para el primer método de conexión con Google utilizada, es decir, mediante la identificación de un usuario, con su propia cuenta de Google.

Aon-google-apis-servlets

Los archivos de este paquete se corresponden a los servlets para poder realizar la conexión con Google.

GoogleAuthorizationCodeCallback.java: Callback del servlet para la conexión con Google.

Funciones:

- **public static** String getUsername()
- **public static** String getPassword()
- **protected void** onSuccess(HttpServletRequest req, HttpServletResponse resp, Credential credential)

- **protected** String getRedirectUri(HttpServletRequest req)
- **protected** String getUserId(HttpServletRequest req)
- **protected** AuthorizationCodeFlow initializeFlow()

GoogleAuthorizationCodeServlet.java: Código del servlet para la conexión con Google.

Funciones:
- **protected** String getRedirectUri(HttpServletRequest req)
- **protected** String getUserId(HttpServletRequest req)
- **protected** AuthorizationCodeFlow initializeFlow()

*GoogleAuthorizationServletUtils.java***:** En este archivo se generan los datos correspondientes a la conexión con Google.

Funciones:
- **public static** JsonFactory getJsonFactory()
- **public static** HttpTransport getHttpTransport()
- **public static** AuthorizationCodeFlow newFlow()
- **public static** GoogleClientSecrets getClientCredential()
- **public static** String getAuth2CallbackUri(HttpServletRequest req)
- **public static** String getPrincipalShortName(HttpServletRequest req)

A continuación se especificarán los archivos necesarios para realizar la conexión con Google utilizando el segundo método descrito anteriormente, es decir, la conexión mediante una cuenta de servicio (Service Account) de la apliación.

Los datos de conexión a la cuenta de servicio (Service Account) están almacenados en la base de datos de la AonSolutions.

Aon-google-apis

CalendarUtils.java: Donde se implementarán todas las funciones que tengan que ver con el funcionamiento de Google calendar.

Funciones:
- **public static void** serviceInitialize(String domain)

DriveUtils.java: Donde se implementarán todas las funciones que tengan que ver con el funcionamiento de Google Drive.

Funciones:
- **public static void** serviceInitialize(DomainGserviceaccount d)

DatabaseSync.java: Donde se desarrollarán todas las llamadas a la base de datos con sus correspondientes consultas.

Funciones:
- **public static** DomainGserviceaccount getServiceAccount(String key)

8.1. Iteración 1: Sign-in with Google

8.1.1. Aon-google-apis

En esta iteración únicamente se utilizarán archivos del proyecto aon-google-apis, para realizar lo conexión con Google.

→ aon-google-apis-servlets

GoogleAuthorizationCodeCallback.java: Callback del servlet para la conexión con Google.

GoogleAuthorizationCodeServlet.java: Código del servlet para la conexión con Google.

GoogleAuthorizationServletUtils.java: En este archivo se generan los datos correspondientes a la conexión con Google.

8.1.2. Aon-web-aio

Aon-web-aio es el proyecto principal de la plataforma de gestión empresarial AonSolutions, Donde se añadirá un nuevo icono para iniciar sesión mediante Google.

→ Webapp-login

errorlogin.jsp: Escribe los detalles de un error si ha ocurrido algún fallo durante la conexión con Google.

formredirection.jsp: Redirecciónа del inicio de sesión de Google a la plataforma de AonSolutions.

loginLayout.jsp: Archivo jsp principal de la pantalla de login, donde se añade el icono de inicio de sesión con Google.

popupclose.jsp: Se encarga de cerrar el Pop Up abierto para la autenticación con la cuenta de Google.

popupclosecancel.jsp: Se encarga de cerrar el Pop Up abierto para la autenticación con la cuenta de Google, en caso de que se cancele el inicio con Google.

8.1.3. Aon-jaas

Aon-jaas es el proyecto encargado de realizar el inicio de sesión de AonSolutions. En este proyecto se tratara la opción de que el usuario pueda iniciar sesión desde Google.

→ Aon-jaas-auth-spi-db

OpenIDLoginModule.java: Clase la cuál hereda el objeto LoginModule. En este archivo se modifican dos funciones de LoginModule.

Funciones:

- **public void** initialize(Subject subject, CallbackHandler callbackHandler, Map sharedState, Map options)

- **protected** Principal createIdentity(String username)
- **protected** String getUsersPassword()

8.2. Iteración 2: Google Calendar

En el caso de Google calendar se implementará algo de código en el proyecto aon-ui-commercial, donde estarán los correspondientes listener a añadir evento, modificar evento y eliminar evento.

8.2.1. Aon-google-apis

En el caso de Google calendar, se utilizaran dos archivos Java del paquete correspondiente del proyecto que serán:

→ aon-google-apis

CalendarUtils.java: Donde se implementarán todas las funciones que tengan que ver con el funcionamiento de Google calendar.

Funciones:

- **public static void** serviceInitialize(String domain)
- **public static** CalendarList getCalendars()
- **public static** int searchCalendars(CalendarList calendars , String dato, int n)
- **public static** Calendar addCalendar(Calendar entry, String email)
- **public static** Calendar newCalendar(Domain company, String key)
- **public static** Calendar updateCalendar(Calendar calendar)
- **public static** void removeCalendar(String calendarId)
- **public static** Calendar modifyCalendar(Calendar entry,String email)
- **public static** Calendar modifyCommercialCalendar(CalendarListEntry calendar, Domain company, String key)
- **public static** Event newEvent(CommercialTracking commercialTracking, String key)
- **public static** Events getEvents(String calendarId)
- **public static** Event addEvent(String calendarId, Event event)
- **public static void** updateEvent(String calendarId, Event event)
- **public static void** removeEvent(String calendarId, String eventId)
- **public static** int searchEvents(Events events, int dato, int n)
- **public static void** modifyEvent(Event event, CommercialTracking eventBD, String id,String key)
- **public static void** synchronize()

DatabaseSync.java: Donde se desarrollarán todas las llamadas a la base de datos con sus correspondientes consultas.

Funciones:

- **public static** Map<String, String> getDomains()
- **public static** String getEnterpriseEmail(int id,String key)
- **public static** String getSellerEmail(CommercialTracking commercialTracking, String key)
- **public static** Vector<Domain> getDomain(String domain)
- **public static** Map<Integer,Vector<CommercialTracking>> getCommercialTrackingAll(String key)

- **public static** Project getProject(CommercialTracking commercialTracking,String key)
- **public static** CommercialActivity getActivity(CommercialTracking commercialTracking, String key)
- **public static** Registry getPotencialClient(CommercialTracking commercialTracking, String key)
- **public static** DomainGserviceaccount getServiceAccount(String key)

8.2.2. Aon-ui-commercial

En este proyecto se encuentran los archivos correspondientes a la agenda comercial de AonSolutions, la cuál se quiere sincronizar con Google Calendar. Para ello se cuando el cliente realice una operación sobre un evento, se utilizará un listener para ejecutar la acción pertinente en Google Calendar.

→Aon-ui-commercial-event

GoogleCalendarSynchronizer.java: Donde se encuentran los listener a las operaciones que se pueden realizar sobre un evento.

Funciones:

- **public void** afterBeanAdded(ControllerEvent event)
- **public void** afterBeanRemoved(ControllerEvent event)
- **public void** afterBeanUpdated(ControllerEvent event)

8.3. Iteración 3: Google Task

Para esta iteración, a la hora implementar se utilizarán los proyecto aon-web-aio y aon-ui-resources, a parte del proyecto especifico de este proyecto de fin de carrera, aon-google-apis.

8.3.1. Aon-google-apis

En el caso de Google Task los archivos utilizados del proyecto aon-google-apis son los siguientes:

→ aon-google-apis

TaskUtils.java: Donde se implementarán todas las funciones que tengan que ver con el funcionamiento de Google Task.

Funciones:

- **public static** Integer searchProject(Project project, TaskLists taskLists , int n)
- **public static** Integer searchTask(Task taskBD, Tasks tasks, int n)
- **public static** TaskList newTaskList(Project project)
- **public static** TaskList addTaskList(TaskList taskList, Tasks client)
- **public static** TaskList removeTaskList(TaskList taskList, Tasks client)
- **public static** TaskList updateTaskList(TaskList taskList, Tasks client)
- **public static** Task newTask(Task task)
- **public static** Task addTask(Task task, TaskList taskList, Tasks client)
- **public static** Task removeTask(Task task, TaskList taskList,Tasks client)
- **public static** Task updateTask(Task task, TaskList taskList, Tasks client)
- **public static void** synchronize(Tasks client)

DatabaseSync.java: Donde se desarrollan todas las llamadas a la base de datos con sus correspondientes consultas.

Funciones:

- **public static** Vector<Task> getTask()
- **public static** Project getProjectTask(Integer taskId)

→ aon-google-apis-servlets

GoogleAuthorizationCodeCallback.java: Callback del servlet para la conexión con Google.

GoogleAuthorizationCodeServlet.java: Código del servlet para la conexión con Google.

GoogleAuthorizationServletUtils.java: En este archivo se generan los datos correspondientes a la conexión con Google.

→aon-ui-google-apis-controller

*GoogleTaskController.java***:** Se implementan las acciones a realizar tras realizar un evento en la interfaz gráfica para Google Task.

Funciones:

- **public** Tasks getClientSession()
- **public boolean** isGoogle()
- **public void** sync()

8.3.2. Aon-web-aio

Aon-web-aio es el proyecto principal de la plataforma de gestión empresarial AonSolutions, en él se modificará una parte del portlet de las tareas, ubicado en la página de inicio de la web.

→ Webapp-facelet-homepage-portlet

*taskPortlet.xhtml***:** Corresponde al "widget" donde aparece información acerca de las tareas, se añadirá un nuevo atributo para sincronizar des la interfaz gráfica todas las tareas del usuario correspondiente.

8.3.3. Aon-ui-resources

En este proyecto se añadirán los iconos necesarios, para el portlet anteriormente descrito de la tareas.

→ resources-richCss

aon-richCss.css: Archivo css donde se especifican la ubicación de los iconos necesarios para esta parte.

8.4. Iteración 4: Google Drive

8.4.1. Aon-google-apis

→ aon-google-apis

DriveUtils.java: Donde se implementarán todas las funciones que tengan que ver con el funcionamiento de Google Drive.

Funciones:

- **public static** String getMD5Checksum(InputStream is)
- **public static** Drive serviceInitialize(DomainGserviceaccount d)
- **public static** DriveFile[] getFiles(Drive drive)
- **public static** File getFile(String fileId)
- **public static** FileList getRootFiles(Drive drive)
- **public static** File newFile(Rattach rattach)
- **public static** File insertFile(Drive drive,File file, DriveFile driveFile)
- **private static** File updateFile(Rattach rattach)
- **private static** File insertFile(Rattach rattach,Vector<String> emails,Vector<String> pemails)
- **public static void** setTypes(Vector<RegistryAttachmentType> types2)
- **public static void** setPermissions(String fileId,Vector<String> emails,Vector<String> pemails)
- **public static boolean** checkTypes(Rattach rattach)
- **public static boolean** checkType(Rattach rattach, Vector<RegistryAttachmentType> types)
- **public static void** sync(Rattach rattach,String domain)
- **public static void** synchronize(Integer id, String domain)
- **public static void** synchronize(String domain)
- **public static void** synchronize()
- **public static** InputStream downloadFile(Drive drive,File file)
- **public static void** deleteFile(Drive drive,String fileId)
- **public static void** delete(String fileId,String domain,int id)
- **public static void** main(String[] args)

DriveData.java: Es una clase creada para la aplicación ya especificada anteriormente.

DriveFile.java: Es una clase creada para la aplicación la cual ya se ha detallado anteriormente.

DatabaseSync.java: Donde se desarrollarán todas las llamadas a la base de datos con sus correspondientes consultas.

Funciones:

- **public static** Rattach getFile(int id,String key)
- **public static** InputStream getData(int id,String key)
- **public static** DriveData getDomainFiles(String key)
- **public static void** deleteBlob(int id,String key)
- **public static void** deleteDriveID(int id,String key)
- **public static** Vector<String> getEmails(int id,String key)
- public static Vector<String> getPersonEmails(int id, String key)
- **public static void** addDriveId(String driveId,int id, String domain)

→ aon-google-apis-servlets

GoogleAuthorizationCodeCallback.java: Callback del servlet para la conexión con Google.

*GoogleAuthorizationCodeServlet.java***:** Código del servlet para la conexión con Google.

*GoogleAuthorizationServletUtils.java***:**

→aon-ui-google-apis-controller

*GoogleDriveController.java***:** Se implementan las acciones a realizar tras realizar un evento en la interfaz gráfica para Google Drive.

Funciones:

- **public** Tasks getClientSession()
- **public boolean** isGoogle()
- **public** DriveFile [] getFiles()
- **public void** onDelete(ActionEvent event)
- **public void** fileUploaded(UploadEvent event)
- **public void** fileDownloaded(ActionEvent event)

8.4.2. Aon-web-aio

Aon-web-aio es el proyecto principal de la plataforma de gestión empresarial AonSolutions, en él se creará un nuevo portlet con los archivos que dispone el usuario en Google Drive, ubicado en la página de inicio de la web.

→ Webapp-facelet-homepage-portlet

*googleDrivePortlet.xhtml***:** Corresponde al "widget" donde aparecen los archivos que el usuario tiene en Google Drive con sus correspondientes opciones.

8.4.3. Aon-ui-resources

En este proyecto se añadirán los iconos necesarios, para el portlet anteriormente descrito de los documentos de Google Drive.

→ resources-richCss

*aon-richCss.css***:** Archivo css donde se especifica la ubicación de los iconos necesarios para esta parte.

9. PRUEBAS

Una de las fases del ciclo de vida de un producto, en este caso un modulo de una plataforma de gestión empresarial, es la fase de pruebas.

Como parte que es de un proceso industrial, la fase de pruebas añade valor y calidad al producto resultante que se maneja. Todos los programas tienen errores y en la fase de pruebas es donde se detectan la mayoría de ellos, ese es el valor que añade. El objetivo específico de la fase de pruebas es encontrar cuantos más errores, mejor.

La prueba ideal de un sistema sería analizar todo el sistema en todas las situaciones posibles, de esta manera se encontraría hasta el último fallo. Indirectamente, se garantiza su respuesta ante cualquier caso que se le presente en la ejecución real.

Tal y como se ha comentado hay que realizar el mayor número de pruebas que sean posible, para obtener un buen nivel de calidad en nuestro producto, en primer lugar hay que asegurarse de que las funciones utilizadas en la aplicación funcionan correctamente, pruebas unitarios.

Una vez realizadas las pruebas unitarias, hay que comprobar, que el sistema funciona correctamente con la arquitectura de la aplicación. Para se van a realizar las pruebas de sistema.

En tercer lugar se tiene que asegurar de que se cumplen tanto los requisitos minimos impuestos en el diseño de la aplicación como lo aspectos de calidad correspondiente al susodicho plan de calidad.Estos se asegurará realizando las pruebas de aceptación correspondientes.

Por último se llevará a cabo un proceso de pruebas por parte de la empresa, con dos fases, la primera las pruebas alfa, donde empleados específicos de la empresa AonSolutions, probarán las novedades de la plataforma, incluyendo este proyecto.Y una vez realizada esta primera fase de pruebas, se realizarán las pruebas beta, es decir, pruebas realizadas por un cliente de confianza de la empresa.

9.1. Pruebas Unitarias

Las pruebas unitarias son una forma de probar el correcto funcionamiento de un módulo de código. Sirve para asegurar que cada uno de los modulos funcione correctamente por separado.

Para ello se realizan los casos de prueba para cada una de las funciones no trivial en cada una de las iteraciones del proyecto, de forma que cada caso sea independiente del resto.

El método que se utiliza para llevar a cabo dichas pruebas unitarias, se trata del método de caja negra. Donde se utiliza la ayuda de las bibliotecas de JUnit para realizar cada uno de los casos de prueba.

Las pruebas de caja negra se centran en lo que se espera de un módulo, es decir, intentan encontrar casos en que el módulo no se atiene a su especificación. Por ello se denominan pruebas funcionales, y el que realiza las pruebas únicamente le suministra los datos de entrada y estudia la salida, sin tener en cuenta lo que suceda dentro de la función o módulo estudiado.

A continuación se detallaran todos los casos de prueba realizados en la aplicación, analizándolos por iteración:

9.1.1. Pruebas unitarias Iteración 1

Para la segunda iteración se realizarán las pruebas de caja negra a las siguientes funciones, consideradas indispensables para la aplicación.

Para la función crear identidad:

- Pasarle como dato un username con una cuenta de Google, no relacionado con ningún usuario de AonSolutions.
- Pasarle como dato un username con una cuenta de Google, relacionada con algún usuario de AonSolutions.

Para la función getUserPassword:

- No tiene ningún parámetro de entrada, comprobar que devuelve los resultados correctos.

Los problemas encontrados durante las pruebas unitarias de esta iteración son los siguientes:

- No distinguía si el usuario era de Google o de AonSolutions.
    - Solución: poner un String identificativo al principio del username, si inicia sesión mediante Google.

9.1.2. Pruebas unitarias Iteración 2

Para la segunda iteración se realizarán las pruebas de caja negra a las siguientes funciones, consideradas indispensables para la aplicación.

Para la función crear evento:

- Pasarle como dato un evento de la BD (Commercial Tracking) nulo.
- Pasarle un dominio no existente o que no se corresponde con el evento en cuestión.
- Pasarle los datos correctos.

Para la función Insertar evento:

- Pasarle como datos un evento Event nulo.
- Pasarle un identificador de calendario inexistente o no válido.
- Pasarle los datos correctos.

Para la función Modificar evento:

- Pasarle como datos un evento Event nulo.
- Pasarle un identificador de calendario inexistente o no válido.
- Pasarle los datos correctos.

Para la función Eliminar evento:

- Pasarle como identificador del calendario uno no existente o nulo.

- Pasarle como identificador del evento uno inexistente o nulo.
- Pasarle los datos correctos.

Para la función Crear calendario

- Pasarle como dato (Domain) una empresa nula.
- Pasarle un dominio no existente o que no se corresponde con el evento en cuestión.
- Pasarle los datos correctos.

Para la función Insertar calendario

- Pasarle como dato un calendario Calendar nulo.
- Pasarle los datos correctos.

Para la función Modificar calendario

- Pasarle como dato un calendario Calendar nulo.
- Pasarle los datos correctos.

Para la función Eliminar calendario

- Pasarle como identificador del calendario uno no existente.
- Pasarle los datos correctos.

Los problemas encontrados durante las pruebas unitarias de esta iteración son los siguientes:

- Problemas con consultas sql a la base de datos mal realizadas.
    - Solución: Volver a realizar las consultas correctamente, y volver a comprobar si dan los resultados esperados.
- A la hora de modificar e insertar un calendario, no lo realiza correctamente.
    - Solución: Fallaba la sincronización con Google, el parámetro de la conexión no era correcto.

9.1.3. Pruebas unitarias Iteración 3

Para la tercera iteración se realizan las pruebas de caja negra a las siguientes funciones, consideradas indispensables para la aplicación.

Para la función crear TaskList (Proyecto)

- Pasarle como dato un proyecto nulo.
- Pasarle los datos correctos.

Para la función Insertar TaskList (Proyecto)

- Pasarle como dato un taskList nulo.
- Pasarle los datos correctos.

Para la función Modificar TaskList (Proyecto)

- Pasarle como dato un taskList nulo.
- Pasarle los datos correctos.

Para la función Eliminar TaskList (Proyecto)

- Pasarle como dato un taskList nulo.
- Pasarle los datos correctos.

Para la función Crear Task:

- Pasarle como dato una tarea de AonSolutions nula.
- Pasarle los datos correctos.

Para la función Insertar Task:

- Pasarle como dato una tarea nula.
- Pasarle los datos correctos.

Para la función Modificar Task:

- Pasarle como dato una tarea nula.
- Pasarle los datos correctos.

Para la función Eliminar Task:

- Pasarle como dato una tarea nula.
- Pasarle los datos correctos.

Los problemas encontrados durante las pruebas unitarias de esta iteración son los siguientes:

- Problemas con consultas sql a la base de datos mal realizadas.
    - Solución: volver a realizar las consultas correctamente, y volver a comprobar si dan los resultados esperados.
- Problemas con la conexión con Google.
    - Solución: Google Task API no estaba activada en la consolo de Google API's.
- No se realizaba una comprobación entre fecha de inicio y fecha fin.

9.1.4. Pruebas unitarias Iteración 4

Para la última iteración se realizarán las pruebas de caja negra a las siguientes funciones, consideradas indispensables para la aplicación.

Para la función Crear archivo:

- Pasarle como dato un archivo de la base de datos nulo.
- Pasarle como dato un archivo con el InputStream dañado o inexistente.
- Pasarle los datos correctos

Para la función Insertar archivo:

- Pasarlo como dato un archivo nulo.
- Pasarlo los datos correctos.

Para la función Modificar archivo:

- Pasarlo como dato un archivo nulo.
- Pasarlo los datos correctos.

Para la función Eliminar archivo:

- Pasarlo como dato un identificador de archivo nulo.
- Pasarlo los datos correctos.

Para la función Descargar archivo:

- Pasarlo como dato un archivo nulo.
- Pasarlo los datos correctos.

Los problemas encontrados durante las pruebas unitarias de esta iteración son los siguientes:

- Problemas con consultas sql a la base de datos mal realizadas.
    - Solución: volver a realizar las consultas correctamente, y volver a comprobar si dan los resultados esperados.
- Problemas a la hora de descargar un archivo de Google Drive:
    - Solución: Modificar el código, hasta que funcionó.

9.2. Pruebas de Sistema

Las pruebas de sistema son similares a las pruebas de caja negro, con la diferencia de que a la hora de realizar las pruebas las realiza con el sistema o módulo completo.

Para realizar éstas pruebas, se hará por iteración, dividiendo el sistema en 4 módulos independientes el uno con el otro. Y cada módulo se le realizará las pruebas del pertinente dependiendo sus funciones.

9.2.1. Pruebas de Sistema Iteración 1

En esta primera iteración de la aplicación se realizan las pruebas pertinentes para el inicio de sesión mediante Google. Se han probado los errores posibles que pueda haber al iniciar sesión con Google, y que el comportamiento de logueo sea correcto y eficiente.

Los problemas encontrados durante las pruebas de sistema de esta iteración son los siguientes:

- No aparecían los permisos, para utilizar las Google API’s.
    - Solución: No se pasaban los parámetros correctamente.

9.2.2. Pruebas de Sistema Iteración 2

En esta primera iteración se realizarán las pruebas del modulo de Google Calendar, donde hay que comprobar que los eventos de la agenda comercial de AonSolutions, se sincronizan correctamente. Y comprobar en el sistema que al añadir, modificar o eliminar un evento de la agenda comercial, el listener funcione correctamente.

Primero se va a comprobar la sincronización de todos los eventos, para ello se van a utilizar 2 cuentas de Google propias, comprobando que al sincronizar tanto los calendarios como los eventos se han sincronizado de manera correcta. A parte de ha

creado una nueva clase view, para mostrar por pantalla todos los datos necesarios para poder asegurar que el sistema de sincronización para todos los eventos de AonSolutions funciona correctamente.

Para el segundo caso, añadir, modificar o eliminar un evento, se utilizará una cuenta personal de Google y también se utilizará la clase view creada para ver tanto el proceso como los resultados, y se ejecutará la interfaz visual de la plataforma, para el aseguramiento de que los listener funcionan bien.

Los problemas encontrados durante las pruebas de sistema de esta iteración son los siguientes:

- A la hora de comprobar los eventos en la sincronización, no lo realizaba correctamente.
    - Solución: Cambiar el método de comprobación.
- El tiempo de sincronización no era muy rápido.
    - Solucón: Hacer el algoritmo de sincronización más eficiente.

9.2.3. Pruebas de Sistema Iteración 3

En esta iteración se van a realizar las pruebas del módulo de Google Task, donde hay que comprobar que todas las tareas de un determinado usuario en AonSolutions, se sincronizen correctamente con Google Task.

Para realizar dicha prueba de sincronización de Google Task, se necesitará ejecutar la interfaz visual de la aplicación debido a que el usuario deberá de identificarse con Google para poder ejecutar la acción. Una vez identificado el usuario, se utilizará la clase view creada anteriormente para observar todos los resultados por pantalla y poder determinar que la sincronización con Google Task es correcta.

Los problemas encontrados durante las pruebas de sistema de esta iteración son los siguientes:

- A la hora de comprobar si una tarea o un proyecto, estaban en Google Task, no lo realizaba de forma correcta.
    - Solución Cambiar el método de comprobación.

9.2.4. Pruebas de Sistema Iteración 4

Para esta tercera iteración, correspondiente al módulo de Google Drive, se van a probar dos casos, el primer caso es la sincronización de todos los archivos de una empresa a Google Drive, y la segunda las opciones de Añadir, borrar y descargar un archivo para el widget con los archivos personales de un usuario.

Para el primer caso, se utilizarán una cuenta de Google para ir viendo que el proceso de compartir un archivo al sincronizarlo funcioan correctamente, también se utilizarrá la clase view ya nombrada en las anteriores iteraciones, para poder ver los resultado por pantalla y poder llegar a la conclusión de que funcionan correctamente.

Para el segundo caso se ejecutará la interfaz visual de la plataforma, primero se iniciará sesión de forma normal, para comprobar que el widget no aparece si no se inicia sesión

con Google. Una vez iniciada la sesión con Google, Subir, borrar y descargar archivos, para comprobar que el controlador del portlet funciona de forma correcta.

Los problemas encontrados durante las pruebas de sistema de esta iteración son los siguientes:

- En la sincronización de los archivos, no subía, todos los archivos que tenía que subir a Google Drive.
    - Solución: La comprobación estaba mal realizada.

9.3. Pruebas de Implantación

A parte de las pruebas realizadas por el desarrollador de la aplicación, también se realizan pruebas por parte de la empresa. Estas pruebas dispondrán de dos fases, la fase alfa y la fase beta.

9.3.1. Fase Alfa

En esta primera fase el personal específico de la empresa realizará las pruebas de aceptación del producto en producción, incluyendo los módulos de éste proyecto.Éstas pruebas se realizan una vez se hayan realizado las pruebas detalladas anteriormente.

Una vez que se acepte el módulo por parte de la empresa en la fase alfa, se dispondrá a preparar la versión correspondiente para pasar a la siguiente fase, la fase beta.

9.3.2. Fase Beta

En esta segunda base de pruebas por parte de la empresa, un cliente de confianza de la empresa, realizará las pruebas pertinentes de aceptación, comprobando si consigue los requisitos indicados, y si añadirían algún elemento más a la aplicación para futuras mejoras de la aplicación.

10. IMPLANTACIÓN

La parte de la plataforma de gestión empresarial AonSolutions realizada en este proyecto se implantará una vez pase la fase alfa de pruebas de aceptación por parte de la empresa.

Los 4 módulos de la aplicación realizados a lo largo de este proyecto se encuentran en estos momentos en la fase beta de pruebas, siendo probada por un cliente de confianza de la empresa. Por lo tanto se puede decir, que ya está siendo usado en el entorno laboral de dicho cliente.

Para la implantación de la plataforma de gestión empresarial AonSolutions, la empresa tiene contratados varios servidores en amazon, con sistema operativo Linux.

Para subir toda la aplicación a dichos servidores se utiliza el servicio de Jenkins, el cuál compila toda la plataforma en busca de errores y lo pone a disposición del cliente correspondiente.

Antes de implantar el sistema final, se realizan las pruebas de implantación, ya descritas en el apartado de pruebas. Primer se llevará a cabo la implantación de la versión **alfa** y posteriormente la implantación de la versión **beta**.

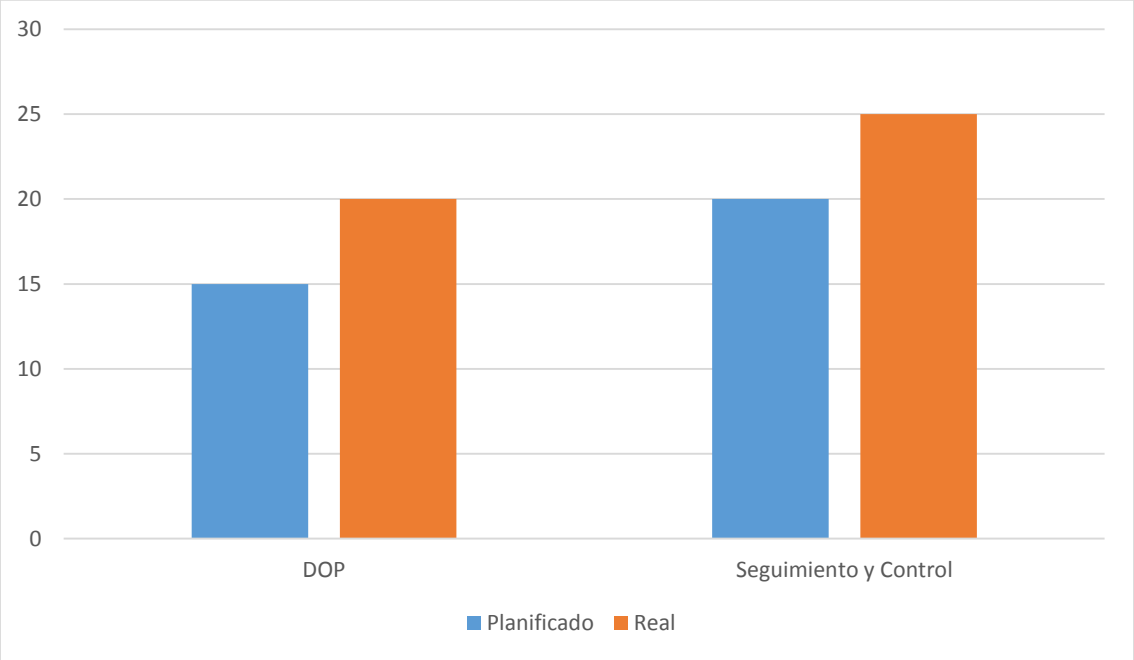
Cuando todo es en condiciones para su uso, se implantarán los módulos realizados en este proyecto en una versión final.

11. GESTIÓN

Durante el desarrollo del proyecto, se ha encontrado diversos problemas y cambios en las cuatro iteraciones del proyecto, por lo que se puede decir que planificación inicial se ha ido modificando y adecuando a las necesidades de cada momento. Todos estos retrasos han surgido debido a varios factores que se detallarán en cada uno de los siguientes apartados, como por ejemplo, la inexperiencia en la realización de proyectos en este entorno, o funcionalidades añadidas a mitad del proceso.

11.1. Incidencias

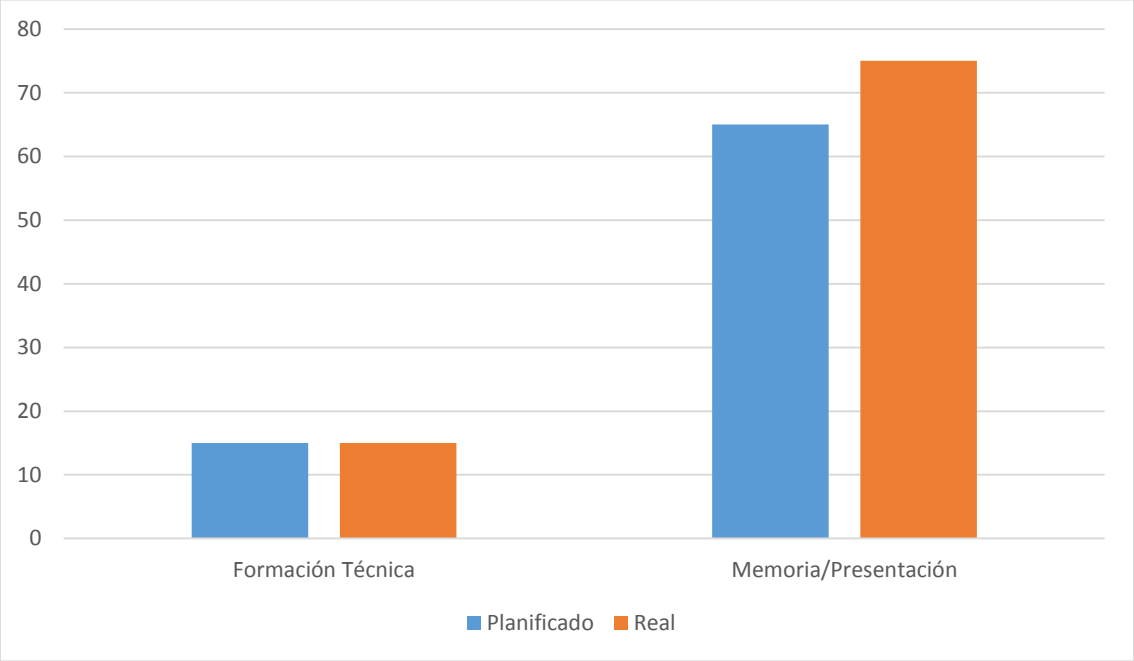
11.1.1. Procesos Tácticos



En general los procesos tácticos, tanto el dop como el seguimiento y control, han tenido un ligero retraso en horas, debido a cambios realizados a lo largo de todo el proyecto. Aunque no haya sido una gran diferencia, con lo que se planificó en un principio, hay que tenerlo en cuenta.

La resolución ante está pérdida de tiempo estaba ya previsto, por lo que se puso un tiempo de colchón para prevenir este suceso.

11.1.2. Procesos Formativos



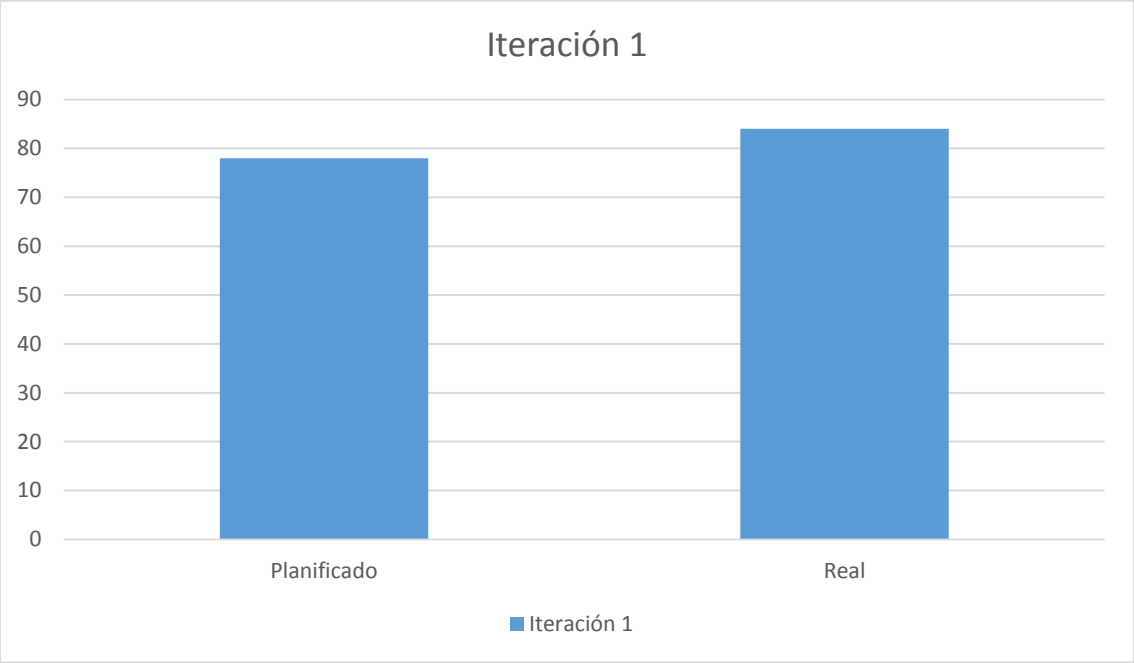
En lo referente a la formación técnica del proyecto, ha ido más o menos según lo planeado, por la tanto, la diferencia entre el tiempo planificado y el real es mínimo.

En lo que se refiere tanto a la memoria como a la presentación, se ha necesitado un poco más de tiempo del estimado sobre todo en el caso de la memoria.

11.1.3. Procesos Operativos

En este apartado se especificaran los problemas que han surgido en cada una de las iteraciones del proyecto. Si han sido incidencias que ya estaban previstas, si se han seguido correctamentes las pautas detalladas en la planificación, o si los problemas surgidos han sido imprevistos y que impacto a generado dichos problemas en proceso del proyecto.

Iteración 1

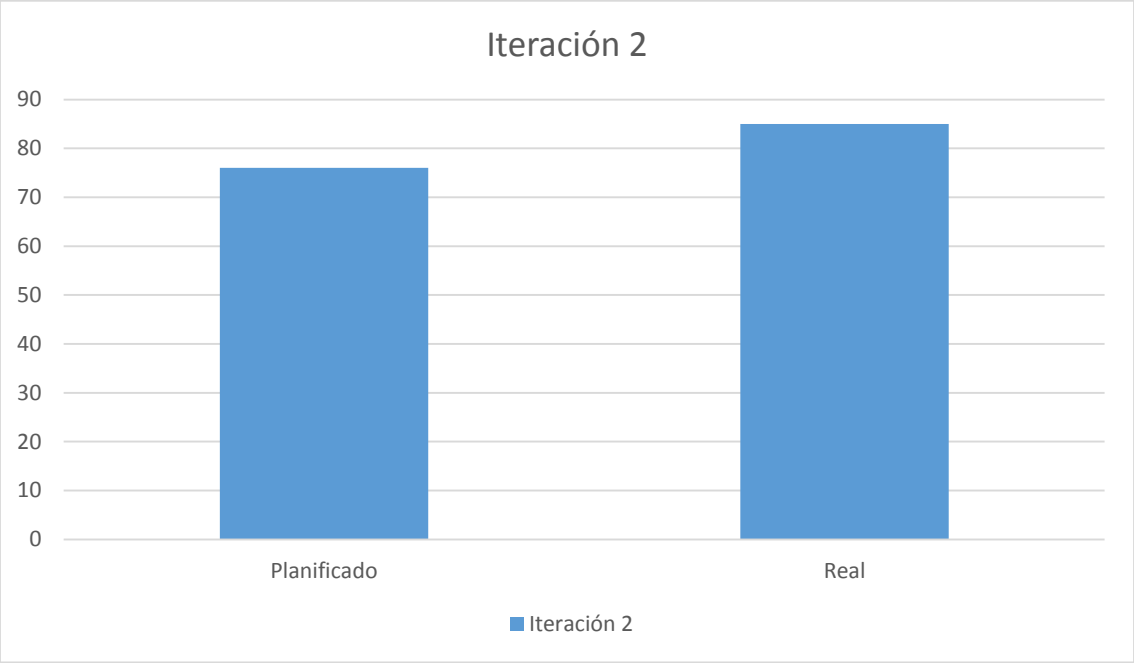


En esta cuarta iteración hubo un cambio importante, una vez realizado casi todo el proceso de la fase. Se cambió por completo la tecnología utiliza, ya que, aunque funcionaba correctamente, se trataba de unos componentes antiguos, por lo tanto se decidió, volver a realizar la mayoría del proceso utilizando en ésta ocasión las librerías que Google pone a disposición OAuth2 API.

Por éste cambio hubo un retraso de horas y fechas de finalización importante, pero al reaprovechar partes de lo ya realizado, no llevo un retraso excesivo en el tiempo.

Con respecto a la calidad planificada, se ha conseguido llegar hasta el nivel mínimo de calidad detallado, por lo tanto se considera satisfactorio el proceso, tanto de verificación como de pruebas utilizado a lo largo de la tercera iteración.

Iteración 2



En un principio se había pensado en realizar una sincronización bidireccional en Google Calendar, debido a la posibibilidad de que el cliente pueda borrar un evento comercial desde Google sin darse cuenta, borrándolo a su vez automáticamente de la base de datos de AonSolutions, por ello se decidio realizar la sincronización de forma unidireccional.

Pero aun así seguía habiendo el mismo problema, en caso de que lo borrase de Google, el evento se tendría que estar sincronizando de nuevo con AonSolutions, por ello se llego a la conclusión que utilizar una cuenta de servicio de la aplicación era una forma mas sensata de realizar dicha sincronización con la agenda comercial de AonSolutions.

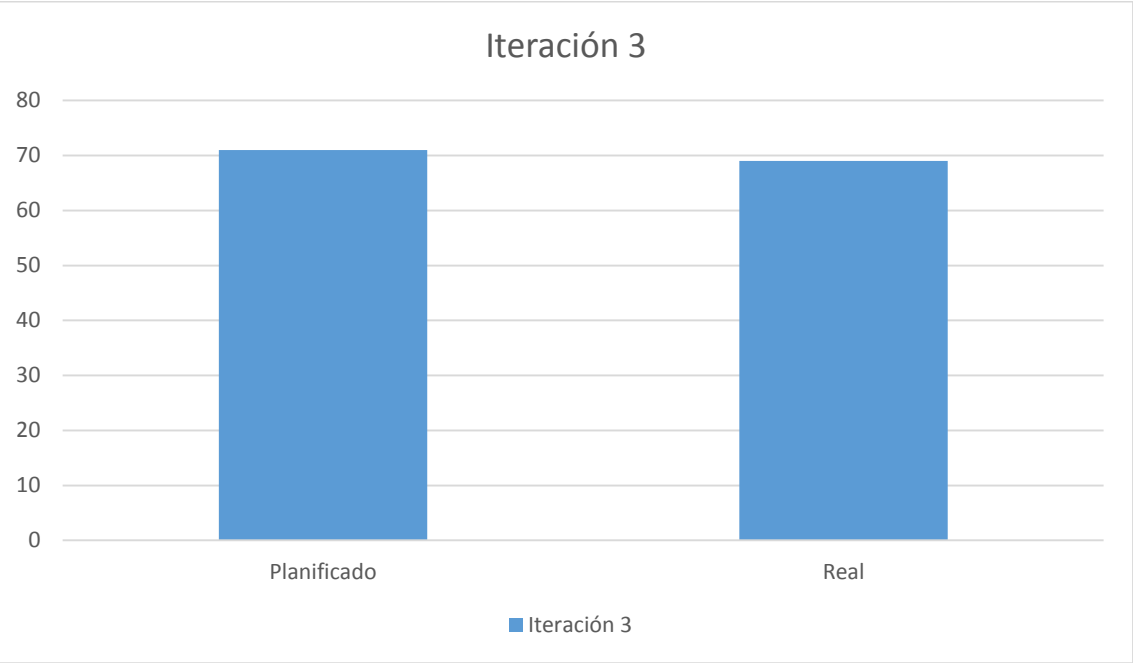
Estos son los grandes cambios que hubo a lo que a funcionalidad se refiere, lo cuál hizo que el tiempo planificado para la realización de esta fase aumentára. Con todo esto e incluyendo la inexperiencia en el ámbito del proyecto, hizo que el trabajo a realizar fuera un poco mas costoso de lo realmente planificado.

Donde realmente se nota la diferencia, en los retrasos con lo planificado, es en el análisis, debido a los cambios realizado y en el diseño, debido a que se tuvo que volver a re planificar el este módulo. En la parte de la implementación, los retrasos fueron debidos a problemas técnicos, tanto en la configuración de la plataforma como en los aspectos técnicos en los que abundaba la inexperiencia.
En las demás fases de la iteración el retraso o adelanto conseguido, son despreciables, por lo tanto se puede decir que se cumplió con lo previsto en esos aspectos.

En definitiva, debido a que sí que se había previsto que pudiera haber algún que otro retardo, por la causa que fuera, se reservó un tiempo aproximado de colchón para no tener excesivos problemas con la gestión del tiempo del proyecto. Por ello no ha sido un aumentado temporal grave en el proceso del proyecto.
En cuenta a las reuniones tanto con la empresa como con el director del proyecto, se retrasaron unos días, hasta disponer de la iteración como terminada.

Con respecto a la calidad planificada, se ha conseguido llegar hasta el nivel mínimo de calidad detallado, por lo tanto se considera satisfactorio el proceso tanto de verificación como de pruebas utilizado durante esta primera iteración.

Iteración 3



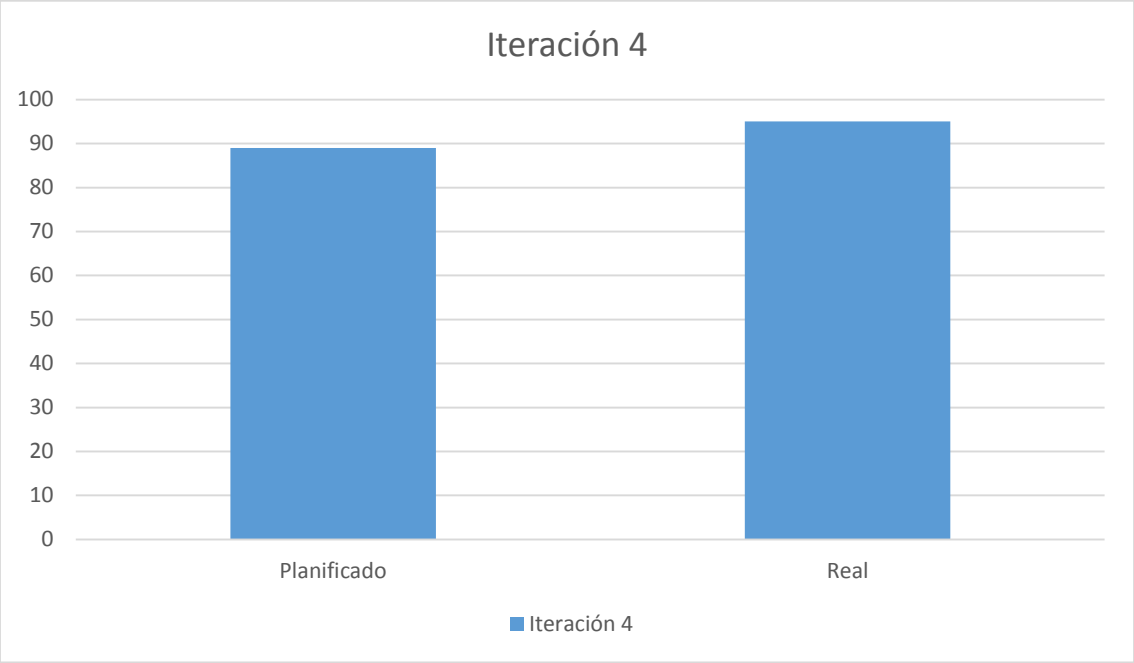
En esta iteración, correspondiente con la sincronización de Google Task, se optó desde un principio, por utilizar la cuenta propia del usuario, para sincronizar las tareas, en vez de utilizar una cuenta de servicio (Service Account). Por lo tanto no hubo ningún cambio importante a lo largo del proceso de esta segunda iteración.

Con respecto a los tiempos necesitados para la realización de esta fase, se ajustan a lo planificado, con un ligero adelante, debido a la experiencia obtenida en la anterior iteración , tanto a la hora de analizar y diseñar la iteración como a la hora de implementarla.

Por lo tanto, se puede decir que para esta fase del proyecto no ha ocurrido ningún problema, y el proceso se ha realizado con normalidad según lo planificado.

Con respecto a la calidad planificada, se ha conseguido llegar hasta el nivel mínimo de calidad detallado, por lo tanto se considera satisfactorio el proceso, tanto de verificación como de pruebas utilizado a lo largo de la segunda iteración.

Iteración 4



Imágen 12.4: Gráfica horas planificadas vs reales, iteración 4.

En un principio se había pensado en realizar únicamente la sincronización de los archivos de AonSolutions, con Google Drive, pero según pasando el tiempo en está tercera iteración, surgió la idea de realizar un portlet donde el usuario pudiera disponer de sus propios archivos de Google Drive.

Por este cambio donde se aumenta la funcionalidad en éste módulo de la aplicación hubo un pequeño retraso tanto en el diseño como en la implementación.
Por otra parte, durante el proceso de esta iteración surgieron varios problemas importantes lo que hizo que se retraso entre una y dos semanas esta parte del proyecto.

Para empezar debido a un problema personal del propio desarrollador del proyecto, no se pudo trabajar y retraso en unos cuantos días la realización de la iteración. Esto ya se tenía previsto y se resoluciona de una manera positiva.
Los otros dos problema importantes surgido, pasaron prácticamente durante la misma época del proceso, primero se estropearon los servidores de la empresa, donde se iban realizando las copias de seguridad, del código implementado. El personal indicado de la empresa consiguió solucionar el problema y recuperar los datos.

El segundo problema fue, que la distribución de Linux instalada falló en el ordenador utilizado para la realización del proyecto, incidencia también prevista que no produjo mas allá de la pérdida de tiempo en volver en reinstalar y configurar el entorno de trabajo para la realización del proyecto. No hubo ninguna pérdida de datos,y aunque se disponía de una copia de seguridad, no hizo falta recurrir a ella debido a que se puedieron recuperar todos los datos de una forma sencilla.

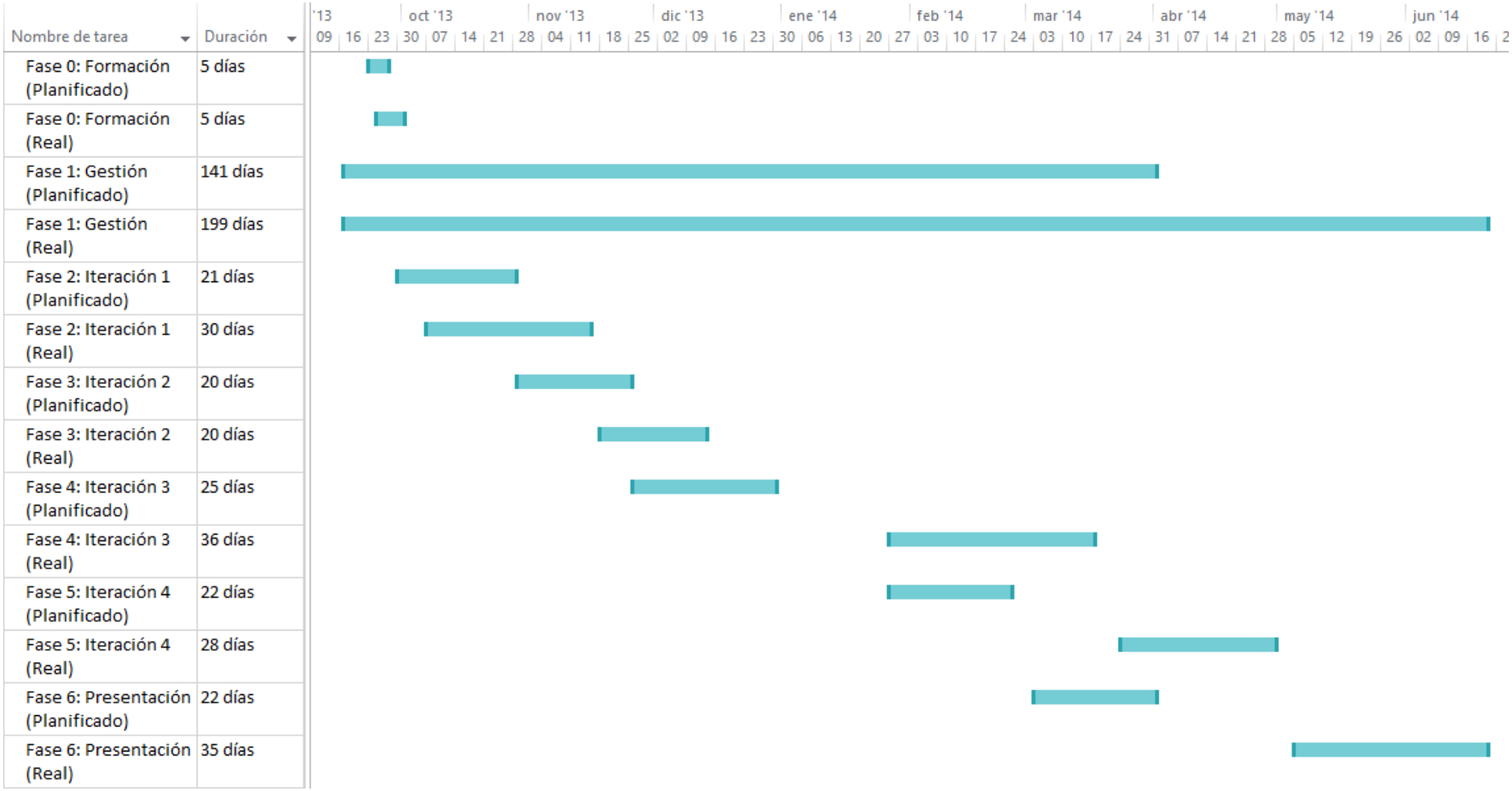
El tiempo tanto en las fechas de de finalización y los horas utilizadas para está iteración ha aumentado, es decir, ha habido un retraso importante en su finalización, aunque se preveeía la posibilidad de un retraso, para esta iteración ha sido excesivo, aunque se hayan cumplido los demás objetivos correctamente.

Con respecto a la calidad planificada, se ha conseguido llegar hasta el nivel mínimo de calidad detallado, y superado, por lo tanto se considera satisfactorio el proceso, tanto de verificación como de pruebas utilizado a lo largo de la tercera iteración.

11.2. Gestión del tiempo

Tareas	***Horas Reales***	***Horas Planificadas***
Procesos Tácticos	45 H.	35 H.
1ª Iteración	85 H.	76 H.
2ª Iteración	69 H.	71 H.
3ª Iteración	95 H.	89 H.
4ª Iteración	84 H.	78 H.
Procesos Formativos	100 H.	90 H.
Total	478 H.	439 H.

11.3. Diagrama Gantt – Planificado vs. Real



Imagén 12.5: Diagrama Gantt – planificado vs. Real

12. CONCLUSIONES

En este apartado se detallan las conclusiones después de la realización del proyecto, tanto en el ámbito del proyecto como en el ámbito personal.

Analizando los resultados, estimo que este proyecto se ha llevado a cabo de una forma razonable, consiguiendo crear una aplicación estable y que cumple con todos los objetivos establecidos al inicio. Incluso superándolos a medida que se le añadieron funcionalidades a la aplicación. Además, de cara a futuro, la aplicación es fácilmente modificable, ya sea para añadirle nuevos casos de uso como para modificar los que ya posee, algo que se tuvo en cuenta desde el inicio del proyecto.

12.1 Trabajo Futuro

Con respecto a las futuras actualizaciones de este proyecto, se puede destacar que sería sencillo tanto añadir nuevas funcionalidades a los módulos realizados en el proyecto como añadir nuevos módulos como por ejemplo Google Analytics Api u otras.

Otro posible avance podría ser el inicio de sesión mediante otras redes sociales, como puede ser, facebook, twitter o linkedIn, aprobechando la parte de OAuth 2.0 con el inicio de sesión de Google.

12.2. Experiencia Personal

Mi experiencia obtenida a lo largo del proyecto en general ha sido muy gratificante, por una parte el hecho de realizar el proyecto en empresa, es una buena forma de ir entrando poco a poco en el mundo laboral, algo que personalmente me ha venido bastante bien debido a que no tenía ninguna experiencia laboral previa.

Por otra parte los aspectos tratados en el proyecto, me han involucrado tanto en la empresa como a la hora de realización del proyecto, ya que es un tema el cuál me interesa bastante.

Por último, todo lo aprendido, ya sean aspectos técnicos sobre el proyecto o aspecto sobre la gestión, va a ser una experiencia importante para mi futuro laboral.

13. BIBLIOGRAFÍA

Google APIs: https://developers.google.com/apis-explorer/

OAuth 2.0: http://oauth.net/2/

Wikipedia: http://es.wikipedia.org/

Consultas de Código: http://stackoverflow.com

Diseño de la aplicación: UML y Patrones – Craig Larman

Documentación : Guía de los fundamentos para la dirección de proyectos (Guía del PMBOK) cuarta edición

Apuntes de las asginaturas:

- Ingeniería del Software.
- Planificación y Gestion de Proyectos.