

UNIVERSIDAD DE ALMERIA

ESCUELA SUPERIOR DE INGENIERÍA

“SIGI: Sistema Integral para la Gestión de Importaciones”

**Mención: Ingeniería del software**

**Curso 2017/2018**

**Alumno/a: Salvador Briones Rosales**

**Director/es:  
Antonio Leopoldo Corral Liria**



# Contenido

1. INTRODUCCIÓN .....	7
1.1. Motivación .....	9
1.2. Objetivos .....	10
1.3. Planificación temporal .....	11
1.4. Organización de la memoria.....	12
2. COMPARATIVA DE SOLUCIONES .....	14
2.1. GTI.....	15
2.2. Diferencias con SIGI .....	17
3. TECNOLOGÍAS Y METODOLOGÍAS .....	20
3.1. Tecnologías .....	20
3.1.1. GIT .....	20
3.1.2. C#.....	21
3.1.3. XAML.....	22
3.1.4. JSON.....	23
3.1.5. Xamarin .....	25
3.1.6. Android .....	26
3.1.7. Windows Presentation Foundation .....	27
3.1.8. Windows Communication Foundation.....	29
3.1.9. UML v2.5 .....	30
3.1.10. SQL Server .....	31
3.2. Metodologías.....	32
3.2.1. Metodología Scrum.....	32
4. HERRAMIENTAS Y ADAPTACIONES EN EL PROYECTO .....	34
4.1. Balsamiq Mockups 3 .....	34
4.2. Visual Studio Enterprise 2015 .....	37
4.3. Microsoft Azure .....	41
4.4. Visual Studio Team Services.....	45
4.5. SQL Server 2014 Management Studio.....	51



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

4.6.	Postman.....	52
4.7.	Dispositivos de trabajo .....	54
4.8.	Esquema general de tecnologías y herramientas del proyecto software .....	56
5.	DESARROLLO DE SIGI.....	58
5.1.	Análisis y requisitos.....	58
5.1.1.	Objetivos .....	58
5.1.2.	Actores .....	62
5.1.3.	Requisitos funcionales .....	64
5.1.4.	Requisitos no funcionales .....	82
5.1.5.	Requisitos de información.....	83
5.2.	Modelado y diseño .....	85
5.2.1.	Diagrama de clases del sistema .....	86
5.2.2.	Diagrama lógico de la base de datos .....	87
5.2.3.	Prototipado de las interfaces gráficas .....	88
5.3.	Implementación y desarrollo aplicación de escritorio.....	91
5.3.1.	Creación y estructura del proyecto .....	91
5.3.2.	Configuración repositorio con GIT .....	92
5.3.3.	Aplicación de servicios web Restful .....	93
5.3.4.	Aplicación cliente con WPF .....	104
5.3.4.1.	Patrón de diseño. Modelo-vista-modelo de la vista .....	104
5.3.4.2.	Creación de documentos PDF .....	107
5.3.4.3.	Integración con Google Maps.....	111
5.3.4.4.	Consumo de datos del servicio web Restful .....	112
5.4.	Implementación y desarrollo aplicación móvil.....	115
5.4.1.	Creación y estructura del proyecto .....	115
5.4.2.	Uso de Xamarin en el proyecto.....	116
5.4.3.	Uso de código de barras.....	118
5.4.4.	Realización y creación de firmas grafológicas .....	119





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

5.4.5.	Consumo de datos del servicio web Rest .....	121
5.5.	Pruebas.....	123
5.6.	Despliegue y producción de SIGI .....	125
5.6.1.	Despliegue del sistema.....	125
6.	CONCLUSIONES Y TRABAJO FUTURO .....	127
6.1.	Conclusiones.....	127
6.2.	Trabajo futuro.....	128
	Bibliografía .....	129





## Índice figuras

Figura 1 - El comercio exterior en la economía española .....	7
Figura 2 - Evolución de importaciones y exportaciones por año en España .....	8
Figura 3 - Representación gráfica de SIGI.....	10
Figura 4 - Diagrama Gantt. Planificación temporal del trabajo.....	12
Figura 5 - Pantalla principal aplicación de escritorio GTI .....	15
Figura 6 - Consulta de una importación en GTI .....	16
Figura 7 - Ventana principal SIGI .....	17
Figura 8 - Detalle de una importación en SIGI.....	17
Figura 9 – Captura del listado de importaciones en reparto desde la aplicación móvil .....	18
Figura 10 - Captura de pantalla de la entrega de una importación desde la aplicación móvil .....	19
Figura 11 - Captura de pantalla confirmación de la entrega desde la aplicación móvil .....	19
Figura 12 - Logotipo tecnología GIT .....	20
Figura 13 - Logotipo lenguaje de programación C# de .Net.....	21
Figura 14 - Logotipo XAML de .Net .....	22
Figura 15 - Ejemplo formateo XAML en .Net .....	23
Figura 16 - Ejemplo uso de JSON .....	24
Figura 17 - Logo Xamarin.....	25
Figura 18 - Esquema general Xamarin .....	25
Figura 19 - Logotipo de Android .....	26
Figura 20 - Gráfica comparativa uso de S.O. en el mundo .....	27
Figura 21 - Logo Windows Presentation Foundation .....	27
Figura 22 - Modo de trabajo en WPF.....	28
Figura 23 - Logo Windows Communication Foundation .....	29
Figura 24 - Logotipo de UML.....	30
Figura 25 - Logotipo de Microsoft SQL Server.....	31
Figura 26 - Esquema general metodología de desarrollo software Scrum .....	32
Figura 27 - Logo Balsamiq Mockups .....	34
Figura 28 - Captura de pantalla de la herramienta Balsamiq Mockups.....	35
Figura 29 – Ejemplo 1 de transición entre interfaces dentro de la herramienta Balsamiq Mockups.....	36
Figura 30 - Ejemplo 2 de transición entre interfaces dentro de la herramienta Balsamiq Mockups.....	36
Figura 31 - Logo Microsoft Visual Studio .....	37





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

Figura 32 - Catalogo de las herramientas software en la plataforma Microsoft Imagine .....	37
Figura 33 - Captura de pantalla mostrando los diagramas de Casos de Uso y Clase en Visual Studio 2015.....	38
Figura 34 - Captura de pantalla mostrando el diagrama de la base de datos en Visual Studio 2015.....	39
Figura 35 - Captura de pantalla, ejemplo de codificación en Visual Studio.....	39
Figura 36 - Captura de pantalla, administrador de paquetes NuGet en Visual Studio.....	40
Figura 37 - Captura de pantalla del explorador de test unitarios en Visual Studio .....	40
Figura 38 - Captura de pantalla depurando código en Visual Studio .....	41
Figura 39 - Logo Microsoft Azure .....	41
Figura 40 - Captura de pantalla del grupo de recursos ProyectoTFGSIGI en nuestra cuenta de Azure.....	42
Figura 41 - Comparativa unidades de medida para un DTU en Azure.....	43
Figura 42 - Planes de servicio para bases de datos en Azure .....	43
Figura 43 - Comparativa Planes de Servicio para App Service en Azure .....	44
Figura 44 - Logotipo Microsoft Visual Studio Team Services .....	45
Figura 45 - Panel resumen o dashboard de SIGI sobre Visual Studio Team Services.....	45
Figura 46 - Ejemplo de ramificación por funciones en GIT .....	46
Figura 47 - Tareas de la actividad "Realizar consultas de envíos" desde Visual Studio Team Services .....	47
Figura 48 - Tareas de la actividad "Añadir usuarios" desde Visual Studio Team Services....	47
Figura 49 - Vista detalle de la actividad "Añadir usuarios" .....	48
Figura 50 - Representación gráfica de la velocidad de cada Sprint del proyecto .....	48
Figura 51 - Cumulative flow/Diagrama de flujo acumulativo de las actividades .....	49
Figura 52 - Panel de configuración para la construcción automatizada del proyecto.....	50
Figura 53 - Logotipo SQL Server 2014 Management Studio .....	51
Figura 54 - Ventana de acceso a la Base de datos en Microsoft Azure .....	51
Figura 55 - Añadir reglas de acceso para la base de datos en Azure.....	52
Figura 56 - Logotipo de la aplicación Postman .....	52
Figura 57 - Ejemplo de uso de Postman.....	53
Figura 58 - Fotografía del equipo empleado para elaborar el proyecto.....	54
Figura 59 - Fotografía del Smartphone empleado .....	55
Figura 60 - Esquema general de las tecnologías y herramientas usadas en el proyecto .....	56
Figura 61 - Diagrama de casos de uso de SIGI.....	64
Figura 62 - Diagrama de clases del sistema SIGI.....	86





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

Figura 63 - Diagrama lógico de la base de datos en el sistema SIGI.....	88
Figura 64 - Prototipo ventana principal de la aplicación de escritorio.....	89
Figura 65 - Vista resumida de todas las ventanas prototipadas de la aplicación de escritorio .....	90
Figura 66 - Prototipo ventana/menú principal de la aplicación móvil.....	90
Figura 67 - Vista resumen de todas las transiciones de la aplicación móvil.....	91
Figura 68 - Explorador de soluciones en Visual Studio.....	92
Figura 69 - Enlace/link repositorio GIT en Visual Studio Team Services .....	92
Figura 70 - Vista herramienta Team Explorer dentro de Visual Studio.....	93
Figura 71 - Configuración cadena de conexión entre Azure y Visual Studio .....	94
Figura 72 - Creación de modelo de clases en Visual Studio .....	94
Figura 73 - Explorador de soluciones con código autogenerado .....	95
Figura 74 - Consulta a la base de datos mediante Postman.....	96
Figura 75 - Añadimos modelo en la aplicación de escritorio .....	105
Figura 76 - Vista del patrón MVVM en la aplicación de escritorio .....	106
Figura 77 - Patrón de diseño MVVM en la aplicación de escritorio .....	107
Figura 78 - Captura de la ventana Listado de repartos en SIGI .....	108
Figura 79 - Documento PDF generado del listado de reparto en SIGI.....	109
Figura 80 - Captura de la ventana Ubicación repartidores en SIGI.....	112
Figura 81 - Distribución controladores en el proyecto WPF .....	112
Figura 82 - Captura de pantalla Gestión de usuarios en SIGI.....	114
Figura 83 - Estructura del proyecto de la aplicación móvil.....	116
Figura 84 - Creación de interfaces gráficas con Xamarin en Visual Studio .....	117
Figura 85 - Ejemplo de código de barras.....	118
Figura 86 - Lectura de código de barras desde la aplicación móvil.....	119
Figura 87 - Firma de conformidad de entrega en la aplicación móvil .....	121
Figura 88 - Estructura controladores en la aplicación móvil.....	121
Figura 89 - Listado de incidencias desde la aplicación móvil.....	122
Figura 90 - Testeo de datos con la herramienta Postman .....	123
Figura 91 - Explorador de pruebas en Visual Studio.....	124
Figura 92 - Ejemplo construcción del proyecto .....	125
Figura 93 - Historial de despliegues del sistema.....	126
Figura 94 -Vista detalle del despliegue del proyecto.....	126





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

*A todos los que me apoyan en mi día a día para crecer como persona y profesional.*

*A toda familia por apoyar mis sueños y continuo sacrificio y en especial a ti, que desde arriba siempre me apoyas.*







# 1. INTRODUCCIÓN

La *globalización* es uno de los procesos políticos, culturales, tecnológicos y económicos que a gran escala ha convertido al mundo en un lugar cada vez más interconectado. Los principales avances tecnológicos y las necesidades de expansión de los mercados han creado este concepto ya que a su vez las innovaciones en las áreas de comunicaciones e informática han transformado la manera de cómo nos comunicamos.

Muchos sectores de trabajo se han transformando adaptándose a otras maneras más evolutivas para poder seguir la línea del cambio. Uno de los sectores cambiantes es la logística. Como consecuencia de la globalización, la logística ha transformado tanto la calidad de sus procesos como la cantidad [1] de los servicios de una manera abismal, haciendo que sus procesos se actualicen para mantener este conjunto de medios y métodos fundamentales para el comercio. La logística es un servicio esencial que hace de enlace entre la producción y mercados. [2]

## COMERCIO EXTERIOR

Cifras en millones de euros

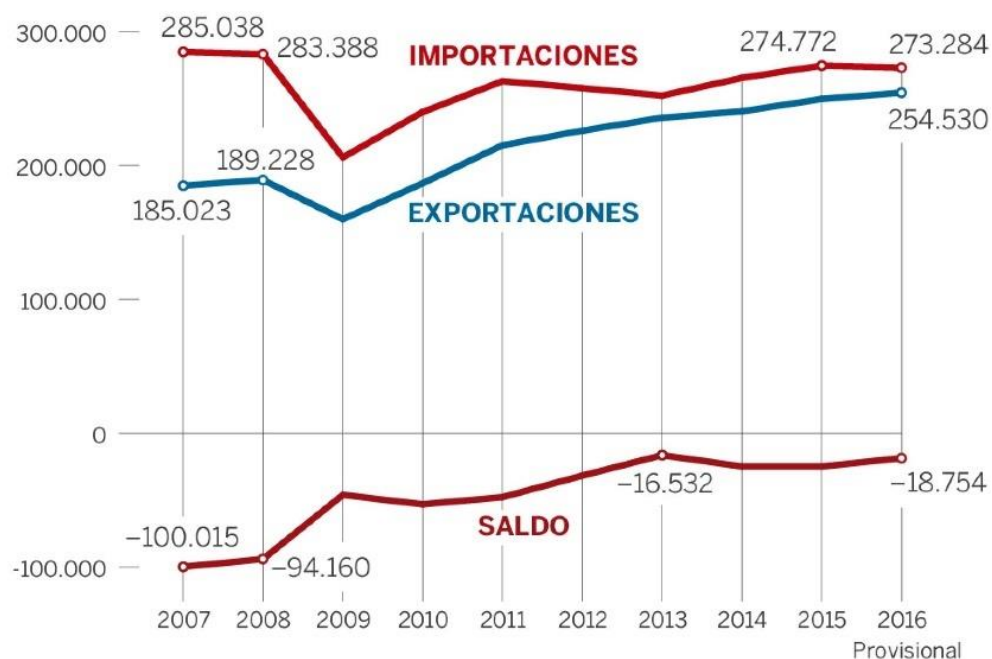


Figura 1 - El comercio exterior en la economía española



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

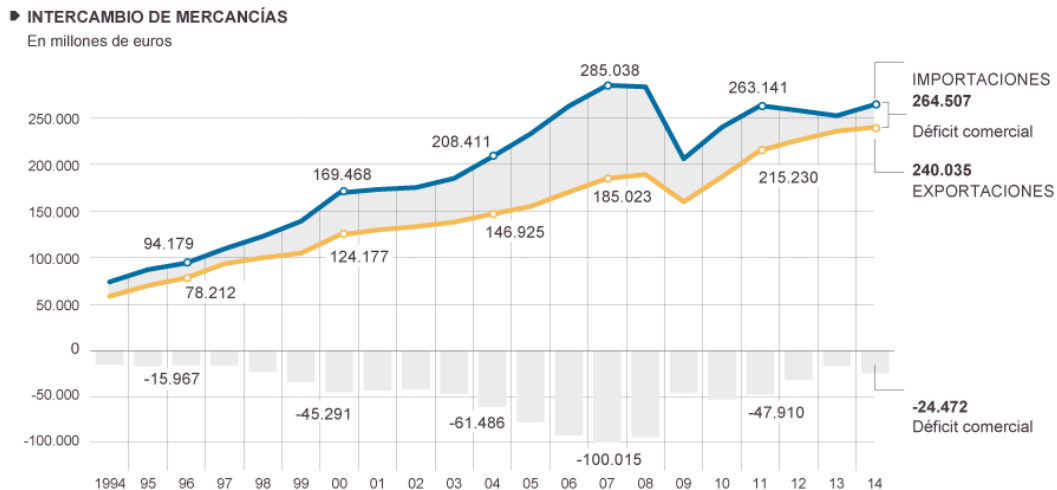


Figura 2 - Evolución de importaciones y exportaciones por año en España

Para dar solución a las necesidades de la globalización gracias al avance de las tecnologías y la expansión de los mercados se han creado sistemas de gestión para realizar la planificación de estos recursos y servicios.

La creación de un software de gestión en el sector de la logística permite administrar y gestionar todos los procesos de la organización logística de una empresa haciendo posible disminuir los costes de transporte, aumentando la calidad tanto en la puntualidad como en la fiabilidad en las entregas.

Crear una infraestructura logística supone unos costes enormes ya que lleva consigo la necesidad de la adquisición de instalaciones industriales, flotas de vehículos, costes de formación para los procesos logísticos y contratación de recursos humanos, sobre todo si se pretende llegar a todos los puntos del mundo manteniendo una calidad y un control de los recursos adecuados. Empresas de logística a nivel internacional subsanan esta problemática subcontratando a empresas del sector logístico para no tener que realizar un desembolso económico y así éstas, realicen la distribución de mercancía en aquellas zonas que no pueden llegar. [3]

Una empresa dedicada al sector servicios se diferencia de las demás empresas por la calidad en su servicio, y a su vez hace que sea imperativo seguir los métodos, pautas y procesos corporativos para mantener dicha calidad tan importante que hace de valor añadido a estas empresas dedicadas a la cadena de suministro. Por consiguiente, las empresas subcontratadas que realizan la distribución se tienen que adaptar a su sistema o como alternativa en crear un sistema diseñado para estos corresponsales logísticos puedan transmitir la información tal y como se trabaja en la empresa contratadora.





### 1.1. Motivación

Desde que terminé mis estudios obligatorios me vi con la necesidad de trabajar, he estado trabajando en varios sectores: agricultura, seguridad, alimentación, hostelería y fue que, hasta finalmente en el año 2014 a la par que estudiaba mis estudios universitarios me ofrecieron la oportunidad de trabajar en el complejo mundo de la logística.

Mi rol en esta empresa era meramente de peón, pero fui escalando hasta obtener una mayor responsabilidad en cuanto a las funciones a realizar. La logística es un sector que me gusta debido a que dentro de su compleja gestión existe un orden a la hora de realizar los procesos.

Una de mis funciones era el uso de una aplicación informática que poco a poco fui aprendiendo a usar. Esta aplicación tenía un sinfín de funciones que no diferían mucho entre si. Veía que necesitaba una reestructuración a la hora de presentar las funciones y su usabilidad. Con el paso del tiempo entendí mejor los procesos usados durante la gestión logística y comprendí que podía mejorarlo y añadir funcionalidades que pudieran complementarla.

Considero que era oportuno no solo crear una aplicación de gestión de escritorio sino también crear una aplicación móvil para el uso de los empleados que se encuentran realizando las labores de reparto.

El uso de nuevas tecnologías me fascina sobre todo si su uso facilita las labores de trabajadores en día a día. Por un lado, en el desarrollo de aplicaciones móviles, quería aplicar algo novedoso y me decante por el desarrollo multiplataforma nativa, es decir desarrollar todo el proyecto software desde un mismo lenguaje de programación para que sea procesado de manera nativa a los diferentes sistemas operativos móviles. No obstante, en nuestro caso desarrollaremos solo una versión para Android ya que es el S.O. más popular [4]. Por otro lado, quería realizar una aplicación de escritorio un tanto vistosa, usable y ligera. Dado que ya conocía el lenguaje de programación C# por asignaturas del Grado, y que el desarrollo de la aplicación móvil y de escritorio lo podía hacer con mismo lenguaje me decante por invertirme en magnifico mundo .NET de Microsoft, después vería que además de magnifico, es complejo. Para ponernos en contexto, mostraremos el sistema que queremos a desarrollar con una imagen:





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

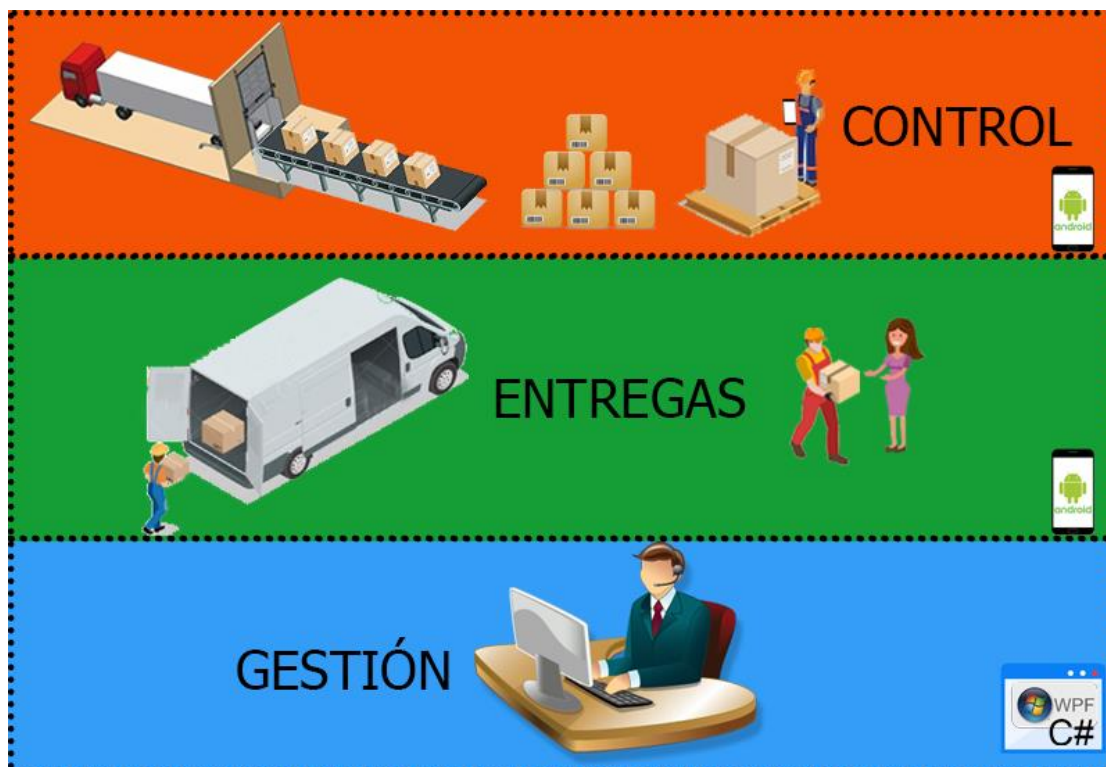


Figura 3 - Representación gráfica de SIGI

### 1.2. Objetivos

El objetivo principal del TFG es crear un sistema íntegro para el control, gestión y entregas de envíos que llegan a una provincia. Una aplicación móvil que tendrá el repartidor y una aplicación de gestión para escritorio que usará el auxiliar administrativo y encargado de la empresa. Para ello desarrollaremos dos aplicaciones:

#### ➤ Aplicación móvil:

- Acceder a la aplicación con las credenciales facilitadas por la empresa.
- Mediante un código de barras se podrá realizar una lectura de la mercancía que llega.
- Subida de datos de las lecturas de etiquetas de importaciones que han llegado.
- Asignar la mercancía que llevará a reparto. Una vez terminada dicha asignación se enviarán los datos al sistema para imprimir un listado con detalle de la mercancía que lleva.
- Actualizar el envío a entregado con la conformidad del destinatario gracias a su firma y el número identificativo (NIF, CIF, NIE o DNI). En caso contrario, el repartidor podrá actualizar la entrega con la incidencia pertinente.
- El terminal enviará en todo momento la ubicación exacta del chofer.

#### ➤ Aplicación de escritorio:



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

- 🚚 Descargar los datos de las importaciones que se han facturado y están pendientes de llegar.
- 🚚 Realizar consultas de envíos.
- 🚚 Gestión de incidencias. Podremos añadir datos relativos del estado de la mercancía.
- 🚚 Consultar listado de repartos de los chóferes.
- 🚚 Generar informes y reportes. Se podrá generar un informe de la mercancía que se ha repartido en el día. También se podrá generar reportes relativos a los diferentes estados de la mercancía: Pendientes de entrega, aplazados de entrega, faltas completas y parciales, mercancía rechazada, etc.
- 🚚 Geolocalización de repartidores. Podremos ver en tiempo real la información relativa del repartidor.
- 🚚 Gestión de usuarios. Añadir, modificar, eliminar y consultar los datos del repartidor.
- 🚚 Realizar modificaciones y eliminaciones de datos relativas al envío.

### 1.3. Planificación temporal

A continuación, detallamos la planificación temporal durante el cual se ha realizado el TFG:

En **febrero de 2017**, se contactó con varios profesores de Grado para ver la posibilidad de ser mi tutor en el trabajo, encuentro tutor y finalmente concretamos el tema del TFG.

En **mayo de 2017**, se entrega la propuesta/anteproyecto de TFG que una vez aprobado, comenzamos con el estudio o fase de investigación de las tecnologías, herramientas y procesos que contendrá el trabajo.

En **junio y julio de 2017**, se desarrolla el proyecto. Durante estos meses se hace un desarrollo intensivo del proyecto. El proyecto se dividió en dos subproyectos: el desarrollo de la aplicación móvil y el desarrollo de la aplicación de escritorio, cada proyecto con sus fases de desarrollo consecuentes: Análisis, modelado y diseño, implementación, prueba y producción.

Durante el mes de **agosto** y el mes de **noviembre del 2017** se realizó la documentación en dos fases; una primera dónde se redactó y se especificó la estructura del mismo y una segunda fase dónde, se desarrollaron los contenidos de este documento o trabajo de fin de Grado. Las correcciones y observaciones se ha ido corrigiendo conforme el tutor ha ido revisando el trabajo.





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

Nombre de tarea	Duración	Comienzo	Fin
Selección tutor y tema	1 ms	mié 01/02/17	mar 28/02/17
Entrega anteproyecto y fase de investigación	1 ms	mar 02/05/17	lun 29/05/17
Desarrollo del proyecto	2 mss	mar 06/06/17	lun 31/07/17
Documentación I	23 días	mar 01/08/17	jue 31/08/17
Documentación II	22 días	mié 01/11/17	jue 30/11/17

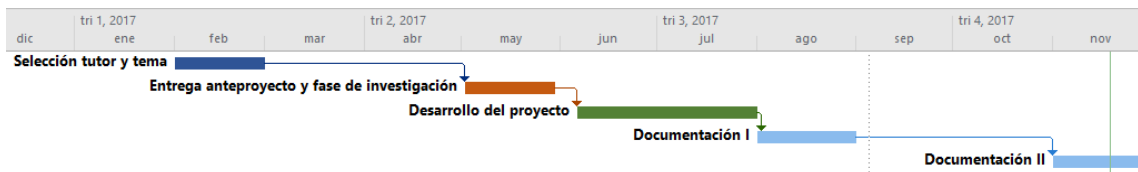


Figura 4 - Diagrama Gantt. Planificación temporal del trabajo.

### 1.4. Organización de la memoria

Conociendo los antecedentes del proyecto software, es decir, dando una introducción de la situación, motivos por los que nos decantamos por este modelo de negocio y las tecnologías a desarrollar, objetivos a realizar y planificación temporal de lo ocupado por parte de proyecto, comentaremos los capítulos de este trabajo y así facilitar la comprensión de la memoria:

#### 🚚 Capítulo 2: Comparativa de soluciones

Sabiendo nuestras necesidades que hemos remarcado en el capítulo de Objetivos, compararemos las diferentes soluciones/aplicaciones ya existentes, dado que la mayoría de aplicaciones son privativas, mostraremos la versión demo de una aplicación de gestión para importaciones.

#### 🚚 Capítulo 3: Tecnologías y metodología.

Describiremos las tecnologías usadas para la realización del proyecto software. Así mismo, mostraremos la metodología usada, detallaremos como hemos seguido sus pautas y explicaremos su adaptación en nuestro proyecto.

#### 🚚 Capítulo 4: Herramientas y adaptación para el proyecto software

Enumeraremos y detallaremos las herramientas que se han utilizado para la realización del proyecto así también mostraremos su configuración y función dentro del proyecto software.


#### 🚚 Capítulo 5: Desarrollo del proyecto





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

Mostraremos las diferentes fases del desarrollo software del sistema. Empezaremos planificando y analizando el sistema a desarrollar. Modelaremos y diseñaremos la lógica de negocio y sus funcionalidades. Seguidamente realizaremos el prototipo de la interfaz gráfica de cada subproyecto, móvil y de escritorio. Detallaremos fases relevantes de la implementación y mostraremos su fase de testing o prueba. Finalmente, explicaremos como se ha realizado la producción y despliegue de las aplicaciones para la puesta en marcha de la totalidad del sistema.

 Capítulo 6: Resultados, conclusiones y trabajo futuro.

Desarrollado el proyecto comentamos que resultados fueron los obtenidos y así también, si existiera alguna posible mejora u otras funcionalidades complementarias que fueran de trabajo futuro dentro de SIGI.





## 2. COMPARATIVA DE SOLUCIONES

Tal y como hemos comentado en el capítulo 1 de Objetivos, nuestras necesidades son las de realizar un sistema íntegro para el control, gestión y entregas de importaciones. Hoy en día existen numerosas aplicaciones para la gestión de envío, pero dado su carácter privado no nos es posible realizar una comparativa de estos sistemas.

Hemos hecho una búsqueda intensiva de aplicaciones o soluciones Open Source basadas en el transporte de mercancía. Hemos encontrado bastantes aplicaciones genéricas basadas en el control de mercancía y su distribución comercial pero no aplicaciones amoldadas o únicas para la gestión, control y entregas de mercancía.

No obstante, dada mi experiencia en el sector y el uso rutinario en mi puesto de trabajo como auxiliar administrativo en logística, conozco la aplicación de escritorio GTI o Gestión de Transporte Integrada, la empresa que desarrolla esta solución es Nivel IV Servicios Informáticos SL. Esta aplicación dispone de una versión demostrativa con todas sus funcionalidades principales y que nos será de ayuda para explicar sus características y compararlo con el sistema desarrollado, SIGI.







## 2.1. GTI

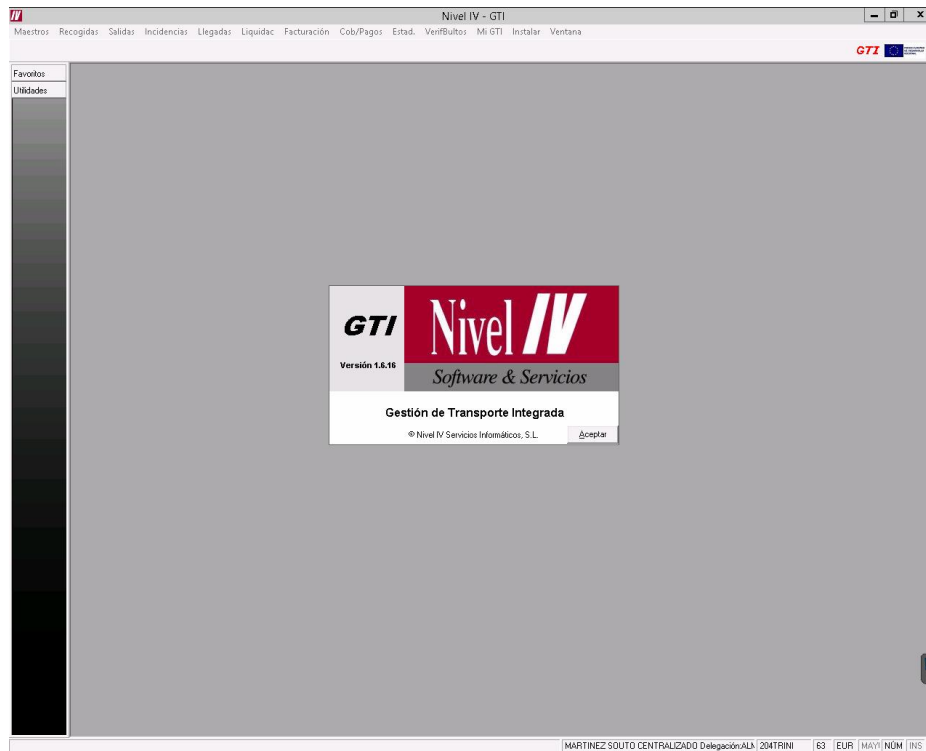


Figura 5 - Pantalla principal aplicación de escritorio GTI

GTI gestiona todas las actividades de una empresa de logística hablamos de: recogidas, exportaciones, importaciones, repartos, facturación, control de pagos, mantenimiento de clientes. GTI es una aplicación cliente que consume de información de una sola base de datos la información de todas las delegaciones.

Podemos ver que las todas sus tareas se organizan en un menú ordenado de manera funcional en procesos centrales. La manera como está organizada sus funcionalidades resulta compleja y difícil de usar si no conocemos los sistemas logísticos y si los conociéramos nos costaría un poco encontrar cada funcionalidad específica.

A continuación, mostraremos el detalle de la consulta de una importación dentro de GTI:



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

The screenshot shows a web application window titled 'Consulta Estado Expedición'. The main content area is a form with several sections: 'General' (Remitente: 116, Consignat: EMP PUBLICA HOSP DE PONIENTE, Salida: 05/09/2017, Llegada: 06/09/2017, Origen: 245 TOLEDO, Destino: 204 ALMERIA), 'Bultos' (1 bulto, 100 Kilos, 1,152 Volumen, 230 Peso Teórico), 'Palets' (0), 'Albarán' (55475), and 'Reparto' (F. Asignación, F. Reparto, Hora, Identikit). The interface is cluttered with many fields and buttons, and the data is not clearly distinguished.

Figura 6 - Consulta de una importación en GTI

Podemos ver que la interfaz es bastante mejorable, no se distinguen los datos más relevantes de la aplicación y contiene un menú desplegable con poca usabilidad, concepto muy importante si se utiliza diariamente.

GTI no tiene una aplicación móvil, el control de mercancía a la llegada de la delegación o empresa debe realizarse manualmente y los retrasos e incidencias por mercancía faltante deben de introducirse manualmente en la aplicación. Por otro lado, las entregas se integran en el sistema de manera manual, en donde el repartidor tiene que entregar en mano a sus compañeros de administración los conformes de entrega de su reparto en el día por lo que el estado de la entrega no es en tiempo real.



## 2.2. Diferencias con SIGI



Figura 7 - Ventana principal SIGI

Utilizando la misma consulta de importaciones usada en la figura 6 y comparando las interfaces graficas de GTI a SIGI existe un cambio a primera vista. La pantalla principal contiene las funciones que útiles y sólo necesitamos y que por consiguiente no tenemos que rebuscar en un menú desplegable. Contiene botones enumerados para mejorar la usabilidad. Siguiendo con el ejemplo de GTI, consultaremos una importación:



Figura 8 - Detalle de una importación en SIGI



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

Podemos comprobar que la información está mejor distribuida, marcando los datos más importantes y relevantes tal y como es el número de envío, la conformidad de entrega y el historial de incidencias si las hubiera.

Por otro lado, el desarrollo de una aplicación móvil hace de punto de inflexión dado que el estado de una importación es actualizado en tiempo real. Desde que llega las instalaciones industriales donde se clasifica por zonas geográficas, cuando sale el repartidor sale a reparto con la mercancía hasta que la mercancía está entregada.

El uso de la aplicación móvil a su vez hace que se reduzca el uso de papel haciendo que a su vez se reduzcan los costes de papelería.

A continuación, mostraremos la interfaz gráfica de la entrega de una importación desde la aplicación móvil:

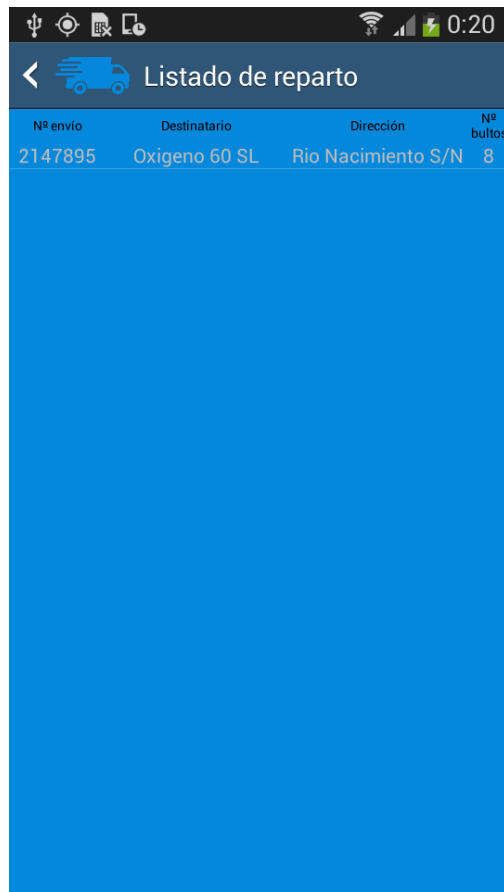


Figura 9 – Captura del listado de importaciones en reparto desde la aplicación móvil



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES



Figura 10 - Captura de pantalla de la entrega de una importación desde la aplicación móvil

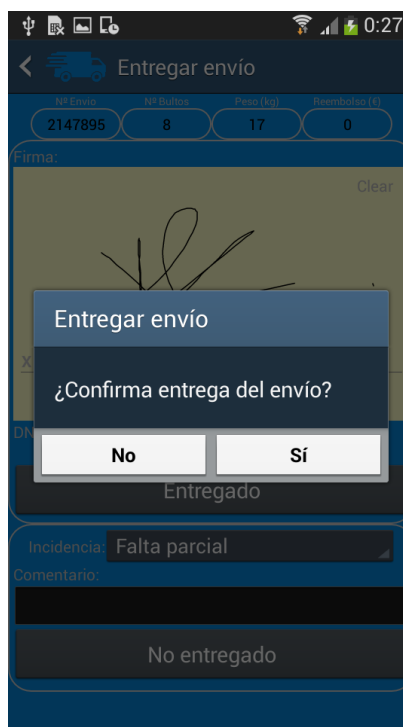


Figura 11 - Captura de pantalla confirmación de la entrega desde la aplicación móvil

Cómo podemos apreciar en las anteriores capturas de pantalla realizadas desde la aplicación móvil existe una correlación de pasos para que la mercancía sea entregada correctamente y que en tiempo real se actualice su estado.



# 3. TECNOLOGÍAS Y METODOLOGÍAS

En este capítulo hablaremos de las tecnologías y la metodología que hemos usado para la realización de nuestro proyecto software.

## 3.1. Tecnologías

Las tecnologías usadas es un punto muy importante a la hora de elaborar un proyecto, ya que si no se escoge la opción correcta puede acarrear atrasos y pérdida de recursos.

### 3.1.1. GIT



Figura 12 - Logotipo tecnología GIT

GIT es un sistema de control de versiones distribuido para del código fuente de nuestra implementación. Un sistema de control de versiones permite la creación de una trazabilidad de los archivos de nuestro proyecto o repositorio permitiendo que incluya un reversionado de estados. Es distribuido ya que cada usuario puede tener su propio repositorio que se almacenara de forma local y que podrán compartir, intercambiar y mezclar las versiones de código fuente entre ellos.

Características:

- 🚚 GIT soporta el concepto de “Ramificaciones” o “Branch” es decir, puedes tener varias versiones diferentes del código fuente, creando de manera paralela una línea nueva al código original. Este sistema es muy flexible ya que se puede crear distintas ramas



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

para distintos usos creando ramas paralelas para su posterior mezcla y así unir todos los cambios del código fuente.

- 🚚 Gestión eficiente y gran rendimiento en repositorios grandes, búsquedas más eficientes lo que supone una gran fluidez para detectar las diferencias de archivos.

### 3.1.2. C#



*Figura 13 - Logotipo lenguaje de programación C# de .Net*

C# es un lenguaje de programación que se ha diseñado para compilar diversas aplicaciones que se ejecutan en .NET Framework. C# es simple, eficaz, con seguridad de tipos y orientado a objetos. Las numerosas innovaciones de C# permiten desarrollar aplicaciones rápidamente y mantener la expresividad y elegancia de los lenguajes de estilo de C.

C# se puede usar para crear aplicaciones cliente de Windows, servicios web XML, componentes distribuidos, aplicaciones cliente-servidor, aplicaciones de base de datos y muchas, muchas más cosas.

- 🚚 Sencillez. C# elimina muchos elementos añadidos por otros lenguajes y que facilitan su uso y comprensión.
- 🚚 Modernidad. Al ser C# un lenguaje de última generación incorpora elementos que se ha demostrado a lo largo del tiempo que son muy útiles para el desarrollador, como tipos decimales o booleanos, un tipo básico "string", así como una instrucción que



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

permita recorrer colecciones con facilidad (instrucción foreach). Estos elementos hay que simularlos en otros lenguajes como C++ o Java.

- ☛ Orientado a objetos. C# como lenguaje de última generación, y de propósito general, es orientado a objetos. C# no permite la inclusión de funciones ni variables globales que no estén incluidos en una definición de tipos, por lo que la orientación a objetos es más pura y clara que en otros lenguajes como C++. Además, C# soporta todas las características del paradigma de la programación orientada a objetos, como son la encapsulación, la herencia y el polimorfismo.
- ☛ Orientado a componentes. La propia sintaxis de C# incluye elementos propios del diseño de componentes que otros lenguajes tienen que simular. La sintaxis de C# incluye por ejemplo formas de definir propiedades, eventos o atributos.
- ☛ Recolección de basura. Todo lenguaje incluido en la plataforma .NET tiene a su disposición el recolector de basura del CLR. Esto implica que no es necesario incluir instrucciones de destrucción de objetos en el lenguaje.
- ☛ Eficiente. En C#, todo el código incluye numerosas restricciones para garantizar su seguridad, no permitiendo el uso de punteros. Sin embargo, y a diferencia de Java, existen modificadores para saltarse esta restricción, pudiendo manipular objetos a través de punteros. Para ello basta identificar regiones de código con el identificador “unsafe”, y podrán usarse en ellas punteros de forma similar a como se hace en C++. Esta característica puede resultar de utilidad en situaciones en las que se necesite gran velocidad de procesamiento.

### 3.1.3. XAML



Figura 14 - Logotipo XAML de .Net

El lenguaje de marcado de aplicaciones extensible, o XAML, es un lenguaje de marcado basado en XML desarrollado por Microsoft. XAML es el lenguaje que subyace a la presentación visual de una aplicación desarrollada en Microsoft Expression Blend, al igual que HTML es el lenguaje que subyace la presentación visual de una página web. Sigue el mismo formato que XML y se declaran de igual manera. A continuación, mostraremos un ejemplo:







## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

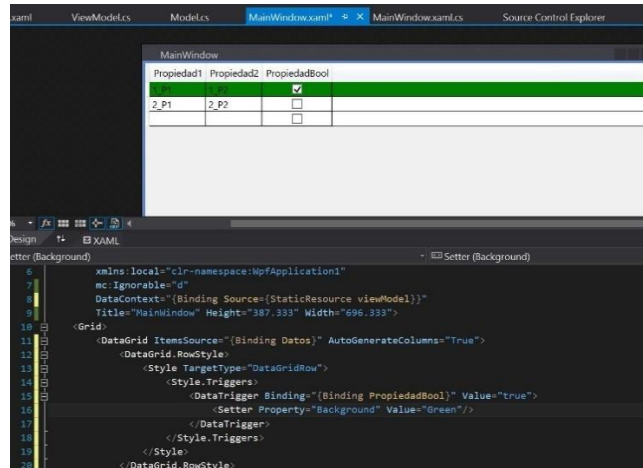


Figura 15 - Ejemplo formateo XAML en .Net

### 3.1.4. JSON



JSON es un formato de texto ligero para el intercambio de datos. JSON es un subconjunto de la notación literal de objetos de JavaScript, aunque hoy, debido a su amplia adopción como alternativa a XML, se considera un formato de lenguaje independiente.

Utiliza una sintaxis que nos permite crear objetos de manera rápida y simple, su uso posibilita la emisión y recepción de información desde aplicaciones y servicios webs.

A continuación, mostraremos un ejemplo de su uso:



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

```
1 {  
2   "nombre": "pedro",  
3   "apellidos": "gomez",  
4   "edad": 20,  
5   "compras": [  
6     {  
7       "producto": "galletas"  
8     },  
9     {  
10      "producto": "limones"  
11    },  
12    {  
13      "producto": "pescado"  
14    }  
15  ]  
16 }
```

Figura 16 - Ejemplo uso de JSON

Podemos ver que muestra la información de un objeto cuyas propiedades o atributos son “nombre”, “apellidos”, “edad” y una lista llamada “compras” en la cual existen varios elementos llamados “producto”.



### 3.1.5. Xamarin



Figura 17 - Logo Xamarin

Xamarin es un sistema y plataforma para el desarrollo de aplicaciones móviles. Su núcleo o API se encarga en el procesamiento y compilación del lenguaje de programación C# para su posterior complicación a entornos iOS, Android y Windows Phone. Se trata de un sistema de desarrollo multiplataforma nativo.



Figura 18 - Esquema general Xamarin

Esto hace posible conseguir que entre un 75% y 100% del código se pueda reutilizar. En nuestro caso podremos reutilizar la lógica de negocio de nuestras aplicaciones de escritorio y poder utilizarla en la aplicación móvil.

Xamarin ya viene instalada por defecto a partir de la actualización 3 de Visual Studio 2015.

La mayor ventaja que tiene Xamarin es la reutilización de código fuente, haciendo posible generar un código compartido para las diferentes plataformas móviles.

En nuestro caso, utilizaremos las ventajas de Xamarin para usar la misma lógica de negocio, pero solo desarrollaremos una aplicación en la plataforma Android.



### 3.1.6. Android



*Figura 19 - Logotipo de Android*

Android es un sistema operativo inicialmente pensado para teléfonos móviles, al igual que iOS, Symbian y Blackberry OS. Lo que lo hace diferente es que está basado en Linux, un núcleo de sistema operativo libre, gratuito y multiplataforma.

El sistema permite programar aplicaciones en una variación de Java llamada Dalvik. El sistema operativo proporciona todas las interfaces necesarias para desarrollar aplicaciones que accedan a las funciones del teléfono (como el GPS, las llamadas, la agenda, etc.).

A continuación, enumeraremos las características más relevantes:

- ☛ Código abierto.
- ☛ Núcleo basado en el Kernel de Linux.
- ☛ Adaptable a muchas pantallas y resoluciones.
- ☛ Utiliza SQLite para el almacenamiento de datos.
- ☛ Ofrece diferentes formas de mensajería.
- ☛ Navegador web basado en WebKit incluido.
- ☛ Soporte de Java y muchos formatos multimedia.
- ☛ Soporte de HTML, HTML5, Adobe Flash Player, etc.
- ☛ Incluye un emulador de dispositivos, herramientas para depuración de memoria y análisis del rendimiento del software.
- ☛ Catálogo de aplicaciones gratuitas o pagas en el que pueden ser descargadas e instaladas (Google Play).
- ☛ Bluetooth.
- ☛ Google Talk desde su versión HoneyComb, para realizar videollamadas.
- ☛ Multitarea real de aplicaciones.



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

Android es el sistema operativo móvil más utilizado. Esto es a sus grandes características y a la gratuidad de su uso. Es por ello por lo que nos decantamos por este sistema operativo que va en alza y es interesante saber más sobre este entorno.

A continuación, mostraremos una gráfica donde podemos ver el volumen de mercado que tiene Android [4]:

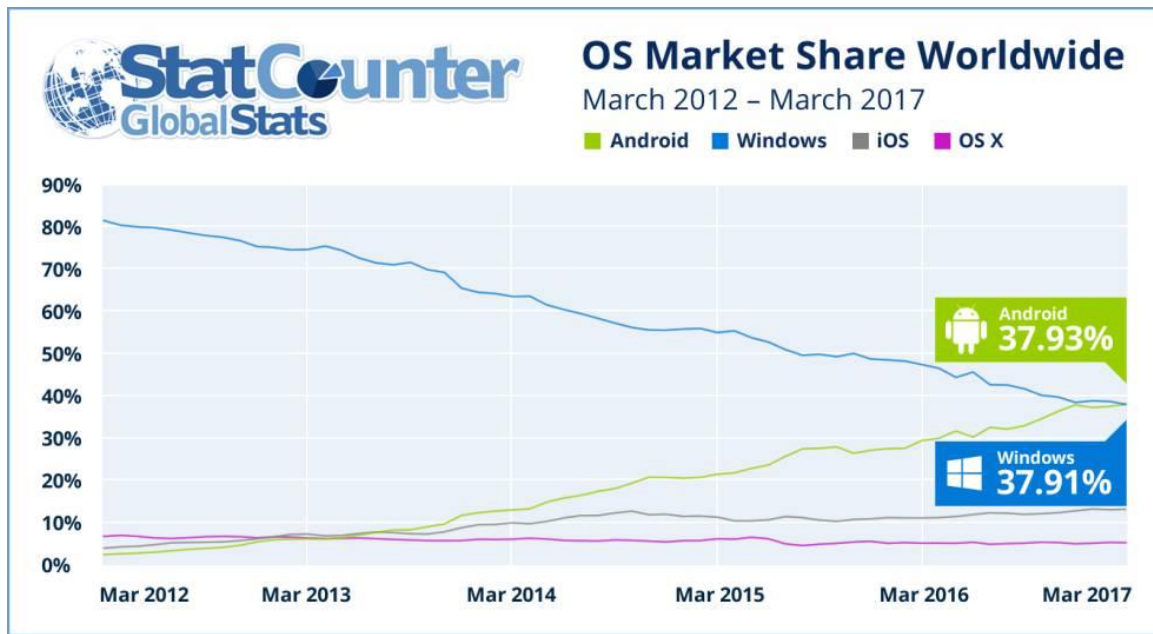


Figura 20 - Gráfica comparativa uso de S.O. en el mundo

Android es el SO más utilizado, podemos ver como Windows ya no es el sistema operativo más utilizado y por consiguiente la tendencia a usar un dispositivo móvil es mayor.

### 3.1.7. Windows Presentation Foundation



Figura 21 - Logo Windows Presentation Foundation



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

Windows Presentation Foundation o WPF es un sistema de presentación enriquecido para crear aplicaciones de escritorio de Windows. WPF nos permite crear experiencias de usuario visualmente atractivas que incorporan que ya incorporan redes sociales y modelos empresariales complejos. Con WPF se pueden desarrollar rápidamente aplicaciones de línea de negocio de clase empresarial con un conjunto de características como:

- 🚚 Interfaz gráfica declarativa. WPF permite crear interfaces de usuario utilizando un lenguaje de marcado llamado XAML. Una de las características más importantes que aporta este lenguaje de programación es que, XAML proporciona un medio para que los diseñadores puedan colaborar estrechamente en la creación de aplicaciones de este tipo.

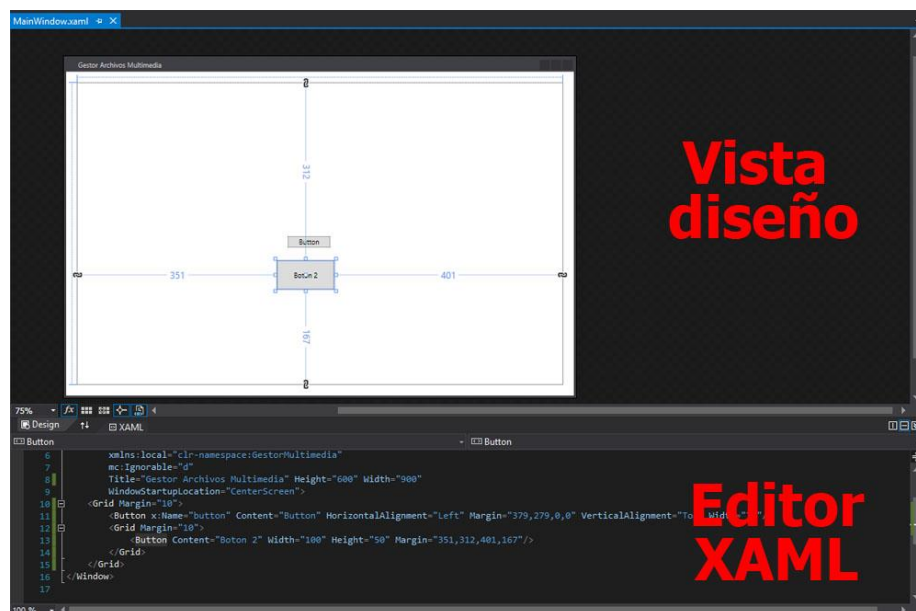


Figura 22 - Modo de trabajo en WPF

Como podemos ver en la figura 19, podemos generar el código XAML en el editor para generar la interfaz gráfica o ya bien podemos crear los elementos en la vista diseño para que se autogenera el código.

- 🚚 Binding. Es la funcionalidad más importante que contiene WPF, gracias a esta funcionalidad podemos utilizar de manera más sencilla el patrón ModeloVistaControlador. Más concreto se utiliza una variante del patrón llamada ModeloVista-VistaModelo.
- 🚚 Estilos. WPF está enfocado agilizar el trabajo de frontend en aplicaciones de escritorio o web es por ello por lo que nos permite reutilizar los estilos con diferentes controles parecido a lo que ocurre con CSS y HTML.





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

Nosotros utilizaremos esta tecnología para el diseño y desarrollo de la aplicación de escritorio.

### 3.1.8. Windows Communication Foundation



*Figura 23 - Logo Windows Communication Foundation*

Windows Communication Foundation o WCF es un marco de trabajo para la creación de aplicaciones orientadas a servicios. Con WCF, es posible enviar datos como mensajes asíncronos de un extremo del servicio a otro. Los mensajes pueden ser tan simples como un carácter o cadena de caracteres, en formato XML, formato JSON (nosotros vamos a utilizar este formato) o simplemente una secuencia de datos binarios.

Las características más relevantes de WCF y en las que nos vamos a basar a utilizar son:

- ☛ **Orientación a servicios.** WCF permite crear aplicaciones orientadas a servicios. Los servicios tienen la ventaja general de estar débilmente acoplados entre una aplicación y otra haciendo más fácil su mantenimiento o mejora.
- ☛ **Metadatos de servicios.** WCF admite la publicación de datos de servicios utilizando los formatos especificados en los estándares de la industria como WSDL, Esquemas XML y WS-Policy. Los metadatos se pueden publicar sobre HTTP y HTTPS.
- ☛ **Compatibilidad con REST.** Los servicios Restful es una tecnología Web 2.0 en evolución. WCF se puede configurar para procesar formatos de mensajes de datos como XML o como JSON.
- ☛ **Extensibilidad.** La arquitectura WCF tiene varios puntos de extensibilidad. Si se necesita una función adicional, existen una serie de puntos de entrada que le permite personalizar el comportamiento de un servicio.

WCF es una tecnología integrada en .Net de Microsoft, vamos hacer uso de ella y de sus características para crear un servicio web que haga de enlace entre nuestra base de datos SQL y nuestras aplicaciones cliente, es decir, la aplicación de escritorio y la aplicación móvil.





### 3.1.9. UML v2.5



*Figura 24 - Logotipo de UML*

UML siglas de Unified Modeling Language es un lenguaje de modelado de sistemas software más conocido y utilizado en la actualidad. Está respaldado por la organización OMG (Object Management Group). Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos, funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y compuestos reciclados.

UML tiene diversos diagramas, cada uno enfocado con un objetivo distinto, los tipos de diagramas vamos a utilizar y están contenido en la versión 2.5 de UML son:

- ☛ Diagrama de clases. Los diagramas de clase son, sin duda, el tipo de diagrama UML más utilizado. Es el bloque de construcción principal de cualquier solución orientada a objetos. Muestra las clases en un sistema, atributos y operaciones de cada clase y la relación entre cada clase.
- ☛ Diagrama de casos de uso. Los diagramas de casos de uso ofrecen una visión general de los actores involucrados en un sistema, las diferentes funciones que necesitan esos actores y cómo interactúan estas diferentes funciones

UML ha sido utilizado en el Grado, sobre todo en asignaturas de la rama Ingeniería de Software. Hemos hecho uso de él para modelar y diseñar nuestro sistema.





### **3.1.10. SQL Server**



*Figura 25 - Logotipo de Microsoft SQL Server*

SQL de las siglas Structured Query Language o lenguaje de consulta estructurada es un lenguaje específico dentro del mundo de las bases de datos. Es un lenguaje estándar e interactivo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones en ellas, gracias a la utilización del álgebra y de cálculos relacionales, el SQL brinda la posibilidad de realizar consultas con el objetivo de recuperar información de las bases de datos de manera sencilla. Las consultas toman la forma de un lenguaje de comandos que permite seleccionar, insertar, actualizar, averiguar la ubicación de los datos, y más.

En nuestro trabajo hemos utilizado este lenguaje y un servidor SQL Server en Microsoft Azure, en los siguientes capítulos hablaremos de su implementación.





### 3.2. Metodologías

Las metodologías de la mano de las tecnologías y herramientas hacen posible la elaboración y desarrollo de un proyecto software. Las metodologías en ingeniería de software es un marco de trabajo usado para estructurar, planificar y controlar los procesos en desarrollo de sistemas o proyectos. A continuación, describiremos la metodología usada.

#### 3.2.1. Metodología Scrum

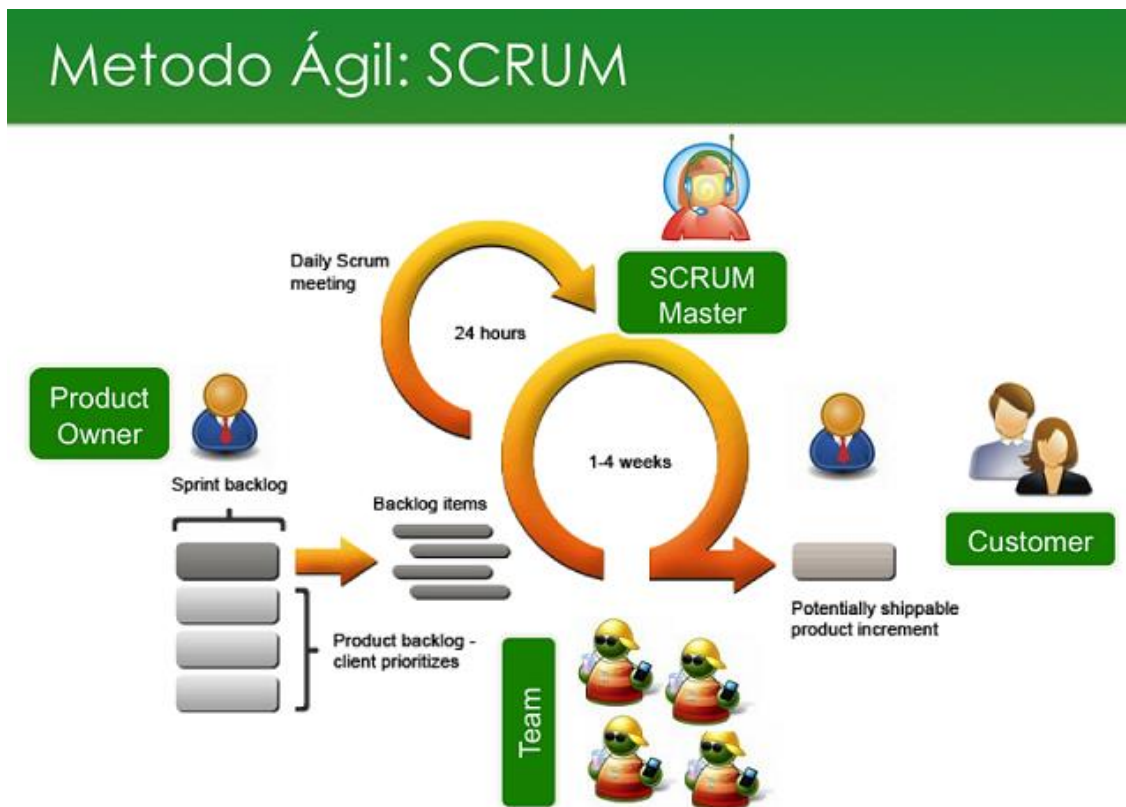


Figura 26 - Esquema general metodología de desarrollo software Scrum

La metodología Scrum es una metodología ágil empleada en el Desarrollo de Software y la Gestión de Proyectos TIC, especialmente indicada para proyectos en entornos complejos, donde se necesita obtener resultados pronto, los requisitos cambian constantemente o están poco definidos y donde la innovación, competitividad, flexibilidad y productividad son fundamentales para conseguir cumplir los objetivos del cliente. De hecho, Scrum es un término del Rugby que quiere decir "melé", cuyo significado es trabajar todos agrupados en equipo. Los actores que intervienen (scrum roles) los podéis ver en la Figura 23.

Existen muchos principios y ventajas de utilizar Scrum, descritos en el Manifiesto Ágil [5] pero los 4 principios fundamentales son estos:

- 🚚 Importancia de los individuos más que los procesos y herramientas.





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

- 🚚 Entregar software que funciona más que documentación exhaustiva.
- 🚚 Colaborar con el cliente más que negociación de contratos.
- 🚚 Respuesta a los cambios más que seguimiento de un plan.

El motivo por el que hemos realizado el proyecto con esta metodología es el desconocimiento de los requisitos necesarios y el tiempo limitado para desarrollar el proyecto. Scrum es complejo pero sencillo de implementar. En mi caso, al solo trabajar una persona en el proyecto hemos versionado esta metodología de manera que el único actor, jefe y desarrollador es el actor del trabajo.

En Scrum se utilizan las iteraciones o Sprints. Estas iteraciones pueden ser bloques temporales cortas y fijas (iteraciones de un mes natural y hasta de dos semanas). Cada iteración tiene que proporcionar un resultado completo, un incremento de producto que sea potencialmente entregable, de manera que cuando el cliente lo solicite sólo sea necesario un esfuerzo mínimo para que el producto esté disponible para ser utilizado.

En nuestro caso, no tenemos ningún cliente al que entregar el proyecto, pero si tenemos que cumplir los plazos para entregarlo. Hemos creado varias iteraciones en el capítulo 5.1 Planificación y análisis, comentaremos con mayor detalle nuestra metodología.





# 4. HERRAMIENTAS Y ADAPTACIONES EN EL PROYECTO

En este capítulo hablaremos de las diferentes herramientas empleadas durante la realización del proyecto software. A su vez, mostraremos las configuraciones utilizadas para que la unión de las tecnologías, herramientas y del mismo proyecto se haya podido dar a cabo.

Una vez enumeradas todas las herramientas mostraremos los dispositivos usados para la realización del proyecto.

Finalmente mostraremos el esquema general con todas las herramientas y tecnologías usadas durante el proyecto.

## 4.1. Balsamiq Mockups 3



*Figura 27 - Logo Balsamiq Mockups*

Balsamiq Mockups es una aplicación para crear maquetas para interfaces gráficas para usuario. Le permite al diseñador diagramar widgets pre construidos utilizando un editor WYSIWYG (what you see is what you get) de drag and drop.

Se trata de un programa profesional de licencia privada. Usaremos la versión más completa, la versión 3. Cuando instalamos la aplicación nos da la opción de usarla gratuitamente durante 90 días, espacio temporal suficiente para poder usarla en nuestro proyecto.

Con esta herramienta diseñaremos las interfaces de usuario tanto de la aplicación de escritorio como la de móvil.





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

Esta aplicación la hemos usado para el prototipado de la aplicación de escritorio y móvil, contiene menús y controles específicos para cada plataforma. Contiene botones, cuadros de texto y todo lo necesario para crear el aspecto de una aplicación, a su vez, existe la posibilidad de añadir botones personalizados de imágenes externas.

A continuación, mostraremos una captura de pantalla del prototipado de la aplicación móvil dentro de Balsamiq Mockups:

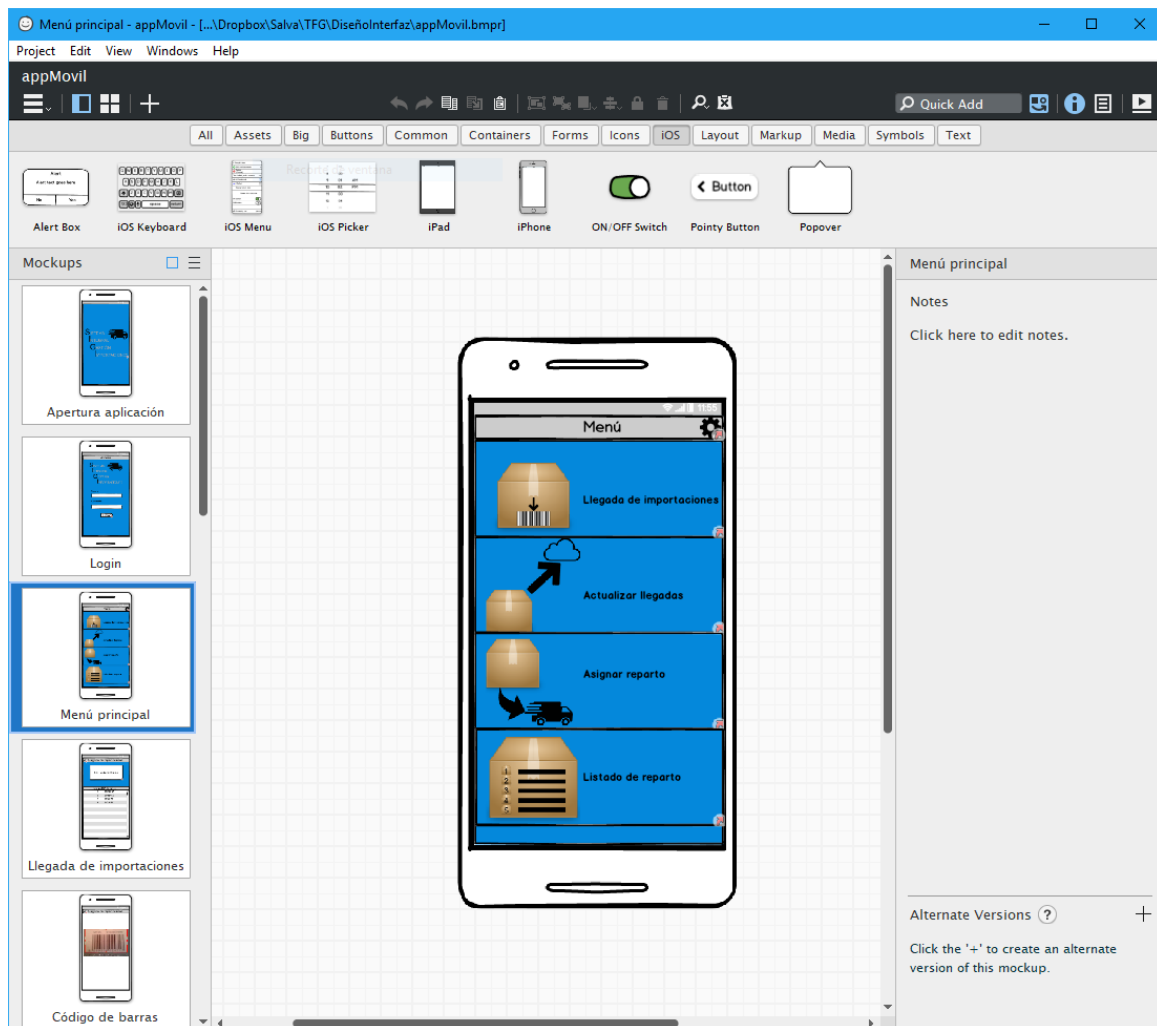


Figura 28 - Captura de pantalla de la herramienta Balsamiq Mockups

Podemos ver en la parte superior de la aplicación que existen elementos categorizados para su uso, desde controles, botones, elementos para formularios, iconos, elementos para app móviles, contenedores y de más elementos que posibilitan la edición de un prototipo.

En la parte izquierda podemos ver las diferentes pantallas e interfaces o transiciones de la app. Cada interfaz simula un estado dentro de la aplicación. Balsamiq tiene la posibilidad de



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

enlazar las interfaces de tal manera que podemos generar una transición entre interfaces y simular una interacción entre estas.

A continuación, mostraremos un ejemplo de transición dentro de Balsamiq:

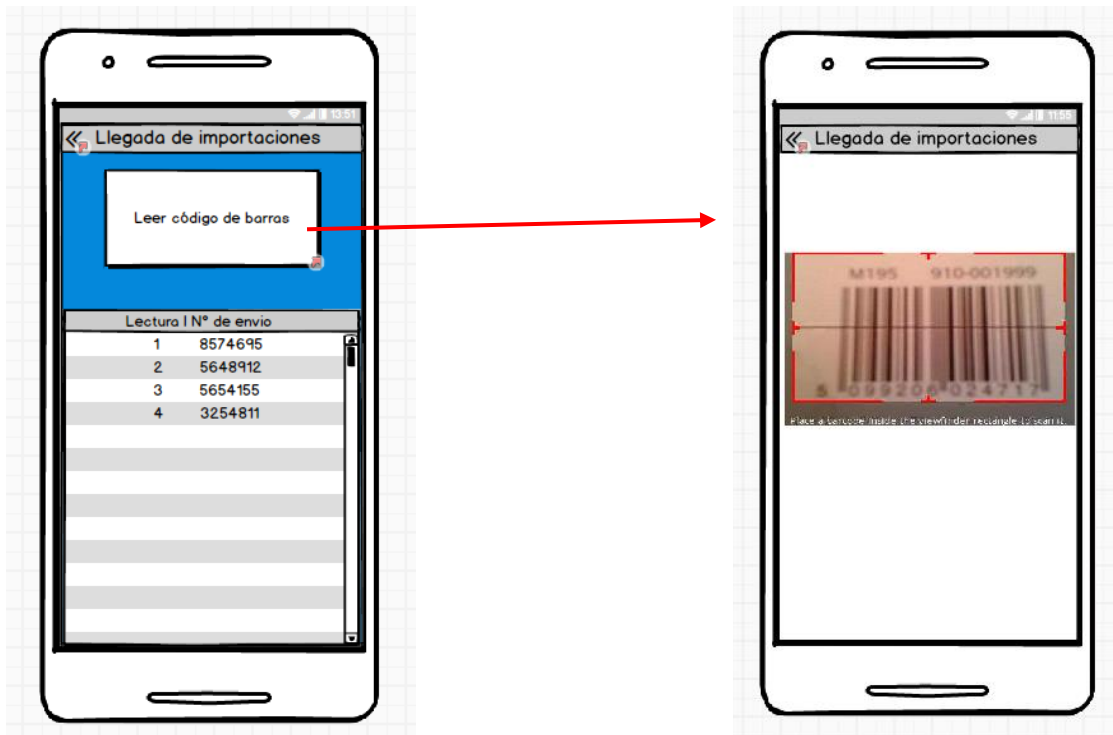


Figura 29 – Ejemplo 1 de transición entre interfaces dentro de la herramienta Balsamiq Mockups

Basándonos en el ejemplo anterior, podemos ver también una transición realizada entre dos interfaces gráficas de la aplicación de escritorio. Hay que recordar que solo interfaz gráfica y no existe ninguna funcionalidad o codificación de la aplicación:

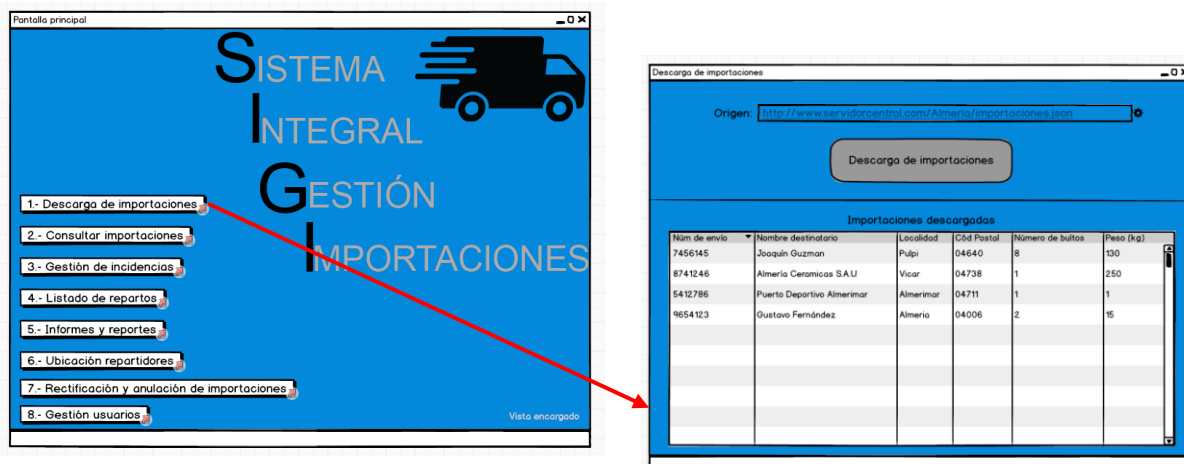


Figura 30 - Ejemplo 2 de transición entre interfaces dentro de la herramienta Balsamiq Mockups



## 4.2. Visual Studio Enterprise 2015



Figura 31 - Logo Microsoft Visual Studio

La herramienta más usada durante nuestra implementación ha sido esta herramienta. Es un IDE de desarrollo integrado creado por Microsoft. Nos permite desarrollar en múltiples lenguajes. No obstante, nosotros nos centraremos en desarrollar en C#. Visual Studio permite a los desarrolladores crear sitios y aplicaciones web, así como servicios web en cualquier entorno que soporte la plataforma .NET (a partir de la versión .NET 2002). Así, se pueden crear aplicaciones que se comuniquen entre estaciones de trabajo, páginas web, dispositivos móviles, dispositivos embebidos y consolas, entre otros.

Hemos usado la versión Visual Studio 2015 Enterprise, exactamente la versión Update 3 que viene con Xamarin integrado.

Hay que decir que esta herramienta ha sido obtenida gracias al acuerdo que existe entre Microsoft y la Universidad de Almería, Visual Studio y números herramientas más de Microsoft, las tenemos totalmente gratuitas todos los alumnos de la Universidad [6].

La plataforma desde la cual hemos obtenido se llama Microsoft Imagine y está gestionado por el departamento de Tecnología de la Información de la Universidad de Almería.

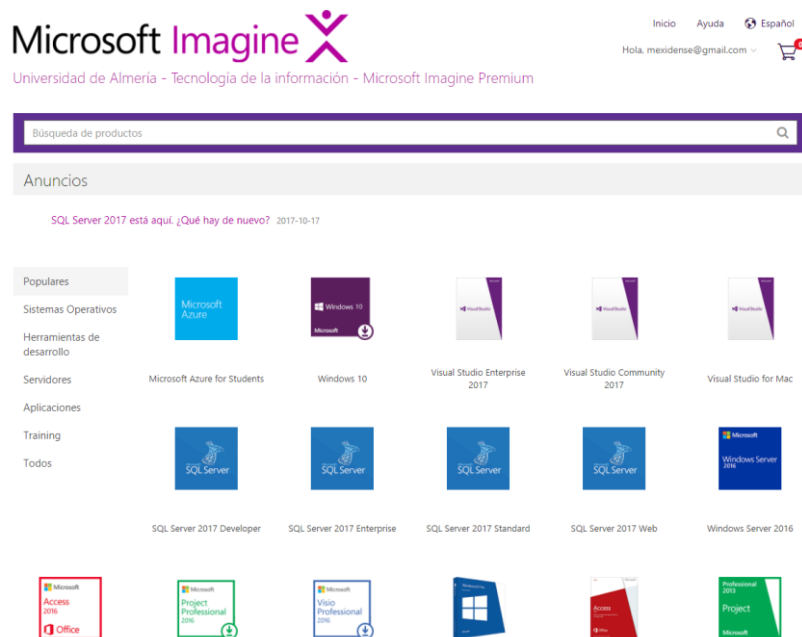


Figura 32 - Catalogo de las herramientas software en la plataforma Microsoft Imagine





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

Visual Studio es una herramienta potente y funcional a la hora de integrar las diferentes etapas del ciclo de vida de nuestro proyecto software. A continuación, enumeraremos las diferentes etapas del ciclo de vida de un proyecto software con el uso dado a la herramienta:

**🚚 Análisis y diseño.** En esta etapa de análisis y diseño del sistema hemos hecho uso de Visual Studio para crear un diagrama de casos de uso para analizar y reunir las funcionalidades del sistema. También se ha creado un diagrama de clases para crear la lógica de negocio.

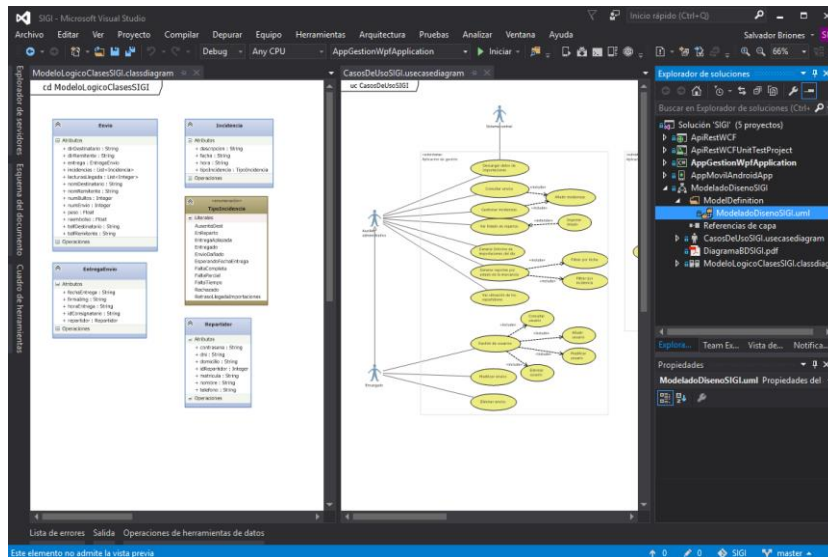


Figura 33 - Captura de pantalla mostrando los diagramas de Casos de Uso y Clase en Visual Studio 2015

A su vez, hemos creado el diagrama del modelo de la base de datos para poder después migrarlo al servidor de base de datos. Cabe decir que en esta parte Visual nos ayuda a crear un diagrama de base de datos exportable a un servidor local basado en SQL Server o alguno creado en la nube de Microsoft Azure:





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

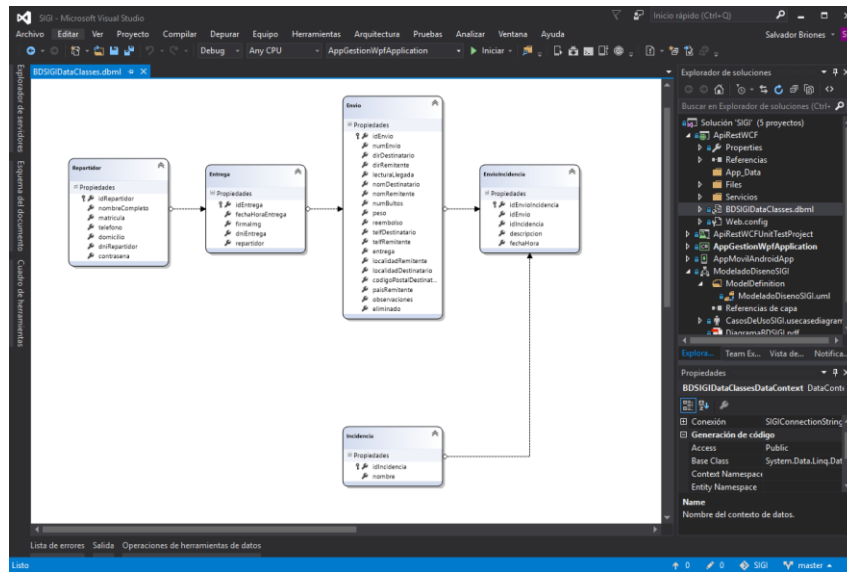


Figura 34 - Captura de pantalla mostrando el diagrama de la base de datos en Visual Studio 2015

**Codificación.** Este entorno de desarrollo integrado es ideal ya que permite codificar de manera precisa y eficiente sin perder el contexto actual. Facilita el uso a llamada de estructuras de datos, funciones relacionadas de dentro y fuera del proyecto mediante librerías. C# es un lenguaje de programación orientado a objetos que en unión a Visual Studio hace que la etapa de implementación sea más fácil y así poder centrarnos más en la ingeniería del proyecto.

```
RepartidorRest.cs | IncidenciaRest.cs | EnvioRest.cs | EntregarRest.cs | client
AppMovilAndroidApp | AppMovilAndroidApp.Controller.RepartidorRest | client
2 referencias | Salvador Briones, Hace 118 días | 1 autor, 2 cambios
58 public List<Repartidor> getRepartidores<T>()
59 {
60     try
61     {
62         var uri = new Uri(Uri + "getRepartidores");
63
64         var response = client.GetAsync(uri).Result;
65         if (response.IsSuccessStatusCode)
66         {
67             string json = response.Content.ReadAsStringAsync().Result;
68             JObject jo = JObject.Parse(json);
69             List<Repartidor> listado = jo.SelectToken("getRepartidoresResult", false).ToObject<List<Repartidor>>();
70
71             return listado;
72         }
73     }
74 }
```

Figura 35 - Captura de pantalla, ejemplo de codificación en Visual Studio

Visual Studio contiene un sistema llamado NuGet, es un potente administrador de paquetes y librerías Open-Source para la plataforma de desarrollo .Net. Para nuestro proyecto ha sido de gran ayuda ya que nos hemos encontrado con una infinidad de recursos para implementar o retocar para nuestro proyecto. Un ejemplo de uso de este administrador de paquetes ha sido al añadir una librería para crear y personalizar





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

documentos PDF en nuestra aplicación de escritorio. De esta manera no volvemos a inventar la rueda ya inventada, sino que la retocamos a nuestro uso.

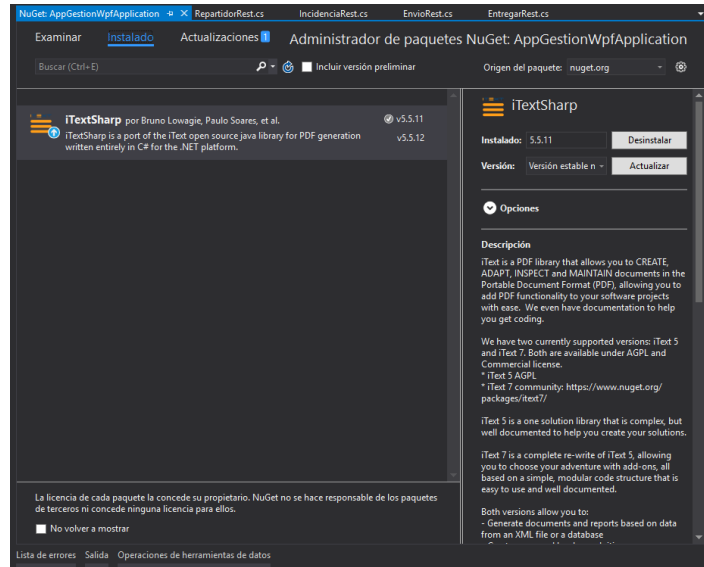


Figura 36 - Captura de pantalla, administrador de paquetes NuGet en Visual Studio

**Pruebas.** Al igual que nos ha facilitado la tarea de codificación, la etapa de pruebas ha sido bastante sencilla. Hemos generado test unitarios para que el desarrollo siga sin ningún tipo de bug o problemas por cambios en el desarrollo.

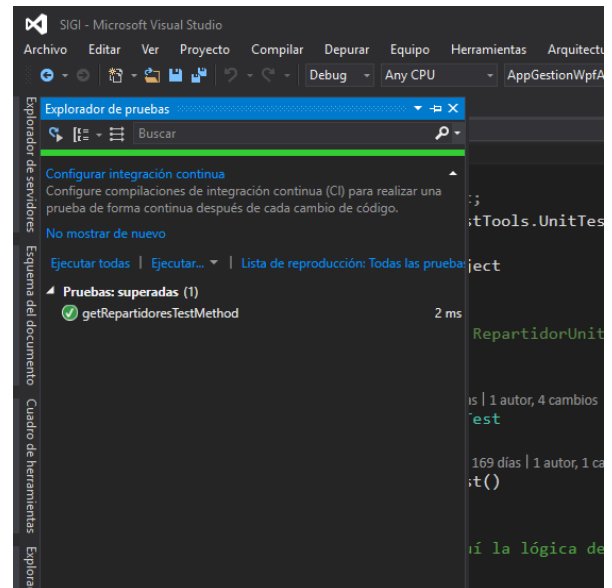


Figura 37 - Captura de pantalla del explorador de test unitarios en Visual Studio

**Mantenimiento.** Al igual que en las demás etapas, Visual Studio contiene un potente depurador de código para tratar aquellos posibles errores que surgen en posteriormente a la implementación. Contiene tareas y elementos fáciles de utilizar para la reutilización de código y refactorización de elementos de implementación.



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

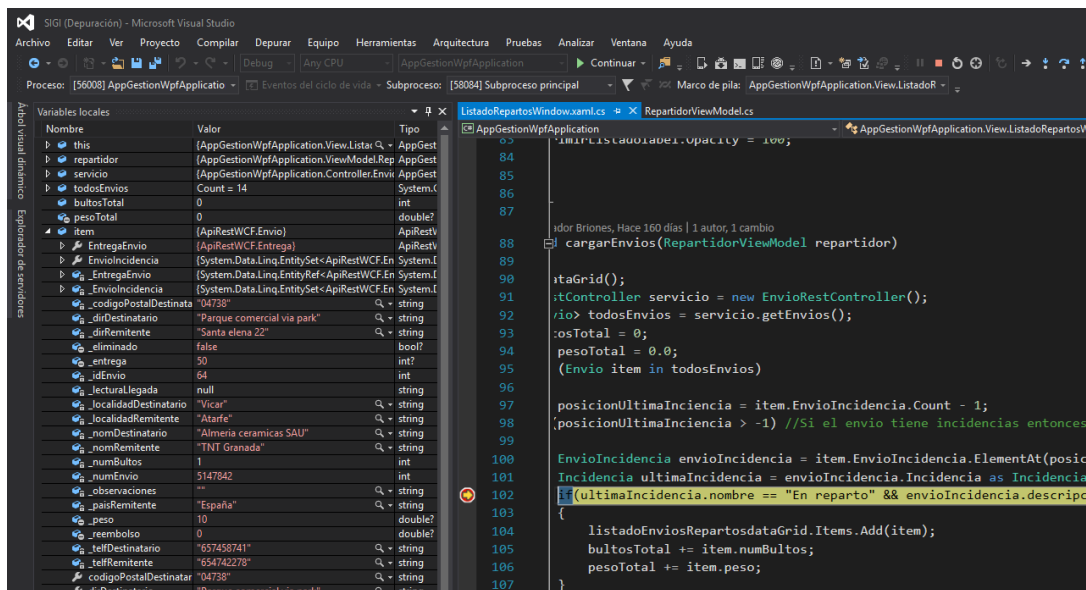


Figura 38 - Captura de pantalla depurando código en Visual Studio

Visual Studio es una herramienta de desarrollo que cumple perfectamente todas las etapas del ciclo de vida de un proyecto software. En los siguientes puntos veremos su integración con Microsoft Azure y Visual Studio Team Services.

### 4.3. Microsoft Azure



Figura 39 - Logo Microsoft Azure

Microsoft Azure es una creciente colección de servicios en la nube integrados que los desarrolladores y los profesionales de TI utilizan para crear, implementar y administrar aplicaciones a través de nuestra red global de centros de datos. Con Azure, tienes la libertad de crear e implementar donde quiera, utilizando las herramientas, las aplicaciones y los marcos que se prefieran.





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

También gracias al acuerdo entre Microsoft y la Universidad de Almería [7] podemos disfrutar de una licencia con servicios mínimos y poder disfrutar de todas las opciones que nos ofrece Azure.

En nuestro caso, hemos creado una base de datos SQL Server 14 que esta comunicada con un servicio web Restful que también hemos desplegado desde una aplicación web.

Desde Azure podemos crear agrupaciones de recursos, es decir, grupos de recursos para interconectarlos en un mismo proyecto. Nosotros hemos creado un grupo de recursos llamado “ProyectoTFGSIGI” en el cual tenemos contratado o creado 4 elementos en la nube:

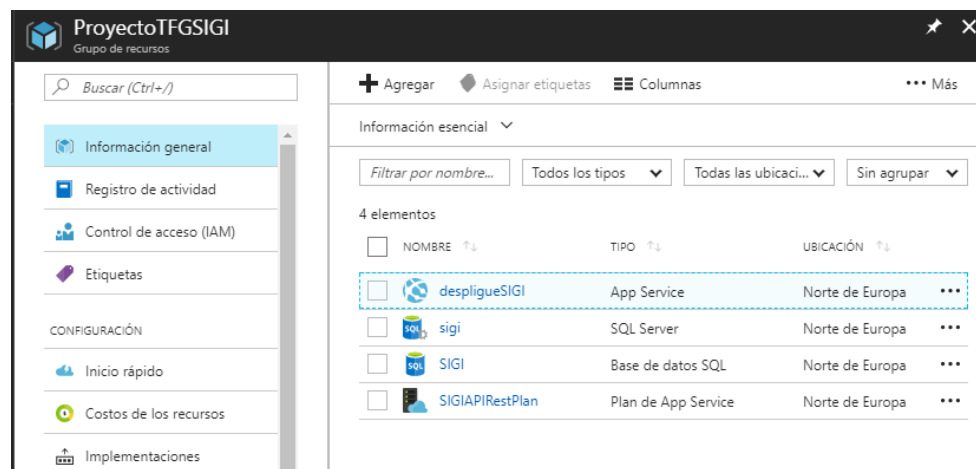


Figura 40 - Captura de pantalla del grupo de recursos ProyectoTFGSIGI en nuestra cuenta de Azure

**🚚 Servidor SQL Server y base de datos SQL.** Hemos creado una base de datos SQL Server 14 en la cual integramos una base de datos sencilla para alojar los datos e información de la aplicación. Nuestra suscripción en Azure es limitada debido a que es un tipo de suscripción gratuita para grupos estudiantiles. En nuestro caso, nos dan hasta 5 DTU para gastar en base de datos. Los DTU es una medida que combina la CPU, memoria y entrada/salida de datos y de registros de transacciones. Esta medida es útil para comprender la cantidad relativa de recursos entre las bases de datos de Azure en diferentes niveles de rendimiento y de servicio. A continuación, mostraremos en Figura 41 la relación de las unidades de medida calculadas en un DTU y en la Figura 42 los diferentes planes que tenemos en Azure.





## Database Transaction Unit – DTU

### Bounding box

Monitoring database workload utilization within bounding box

Represents the relative power (resources) assigned to the database  
Blended measure of CPU, memory, and read-write rates

Compare the power across performance levels

Simplifies talking about performance, think IOPS vs. %

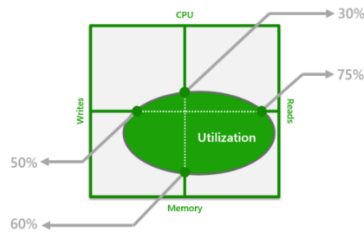


Figura 41 - Comparativa unidades de medida para un DTU en Azure

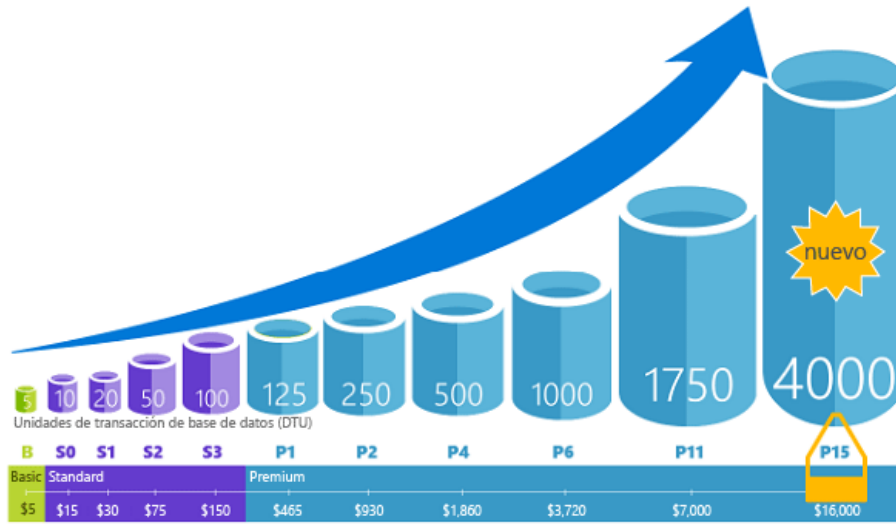


Figura 42 - Planes de servicio para bases de datos en Azure

**Plan de App Service y una App Service.** Primeramente, hemos creado un plan dada nuestra suscripción solo hemos podido crear el Plan más básico en la cual tenemos hasta 1GB de Disco Duro y un límite de 10 aplicaciones por plan, además de estas limitaciones existen otras características a la hora de elegir un plan u otro, pero en nuestro caso es suficiente para crear la aplicación en la nube [8]. En la Figura 43 mostramos la comparativa de los diferentes planes de servicio para App Service.



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

	GRATIS	COMPARTIDA	BASIC	ESTÁNDAR	PREMIUM	AISLADO *	APP SERVICE LINUX	PLAN DE CONSUMO (FUNCTIONS)
<b>- Límites **</b>								
Aplicaciones	10	100	Ilimitado	Ilimitado	Ilimitado	Ilimitado		500
Espacio en disco	1 GB	1 GB	10 GB	50 GB	250 GB	1 TB		
Instancias máximas			Hasta 3	Hasta 10	Hasta 20	Hasta 100		
Contrato de nivel de servicio			99,95 %	99,95 %	99,95 %	99,95 %		
Funcions en Planes de App Service *			✓	✓	✓	✓		
<b>- Implementación de aplicaciones</b>								
Implementación continua †	✓	✓	✓	✓	✓	✓ <sup>4</sup>	✓	✓
Ranuras de implementación				✓	✓	✓	✓	
Docker (Containers)							✓ <sup>1</sup>	

Figura 43 - Comparativa Planes de Servicio para App Service en Azure

Hay que tener en cuenta que nuestro plan contiene la posibilidad de usar Implementación continua, este proceso lo detallaremos en el capítulo 5 de Desarrollo del proyecto.

Por otro lado, Azure nos da la posibilidad de crear varios tipos de App Service, podemos crear desde Web Apps para hospedar sitios y/o aplicaciones web, API Apps para hospedar API de Restful o también podemos crear Azure Functions para hospedar cargas de trabajo sin servidor basas en eventos, en nuestro caso hemos creado una App Service para alojar nuestro servicio web Restful desde el cual consumiremos en nuestras aplicaciones.



## 4.4. Visual Studio Team Services



Figura 44 - Logotipo Microsoft Visual Studio Team Services

Esta potente herramienta de Microsoft ofrece al desarrollador una herramienta para cada fase del proyecto desde el seguimiento del trabajo con planificaciones ágiles, control de versiones, publicación y búsqueda de código en la comunidad .Net, contiene también una herramienta de integración continua bastante potente para cualquier lenguaje, administración de construcciones/builds, un sistema para realizar pruebas manuales, automatizadas y de rendimiento así como, informes y logs para comprobar el funcionamiento de nuestros proyecto.

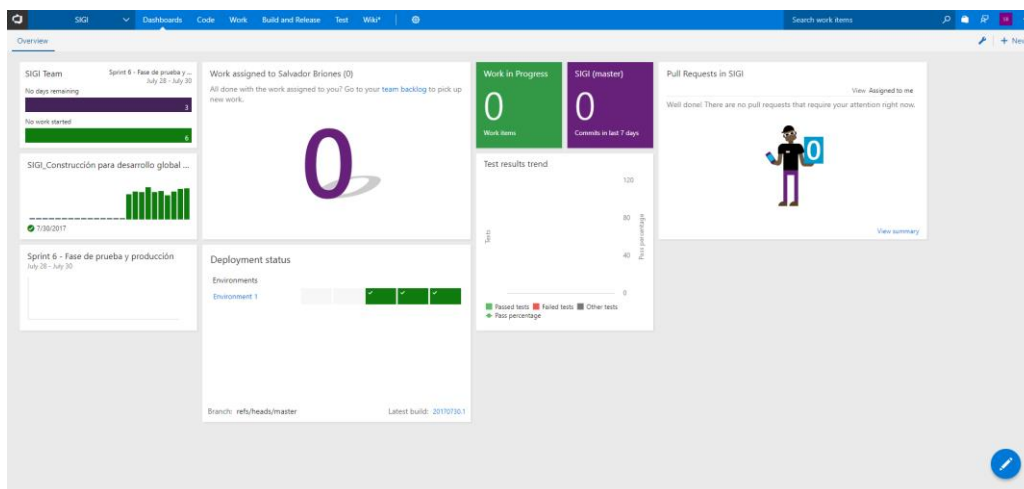


Figura 45 - Panel resumen o dashboard de SIGI sobre Visual Studio Team Services

Desde esta gran herramienta crearemos nuestro repositorio en base a GIT, crearemos las tareas e iteraciones de la planificación basada en nuestra adaptación a metodología Ágil, crearemos nuestras construcciones, probaremos las aplicaciones y las desplegaremos en Azure.



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

**Repositorio.** Desde Team Services podemos crear proyectos, añadir usuarios y crear grupos de roles para estos usuarios. En nuestro caso, sólo habrá un usuario que será el mío como desarrollador y propietario del proyecto software.

Hemos creado un repositorio para el control de versiones de nuestro proyecto, como hemos visto en el capítulo 3 Tecnologías, GIT es la tecnología de versionado a usar y su característica más importante es la de ramificaciones. Hemos creado varias ramificaciones por funcionalidad de tal manera cuando estas han estado totalmente realizadas se han integrado a la rama inicial o rama máster. A continuación, mostraremos de manera visual un ejemplo de ramificación y unión de una rama por funcionalidad del proyecto:

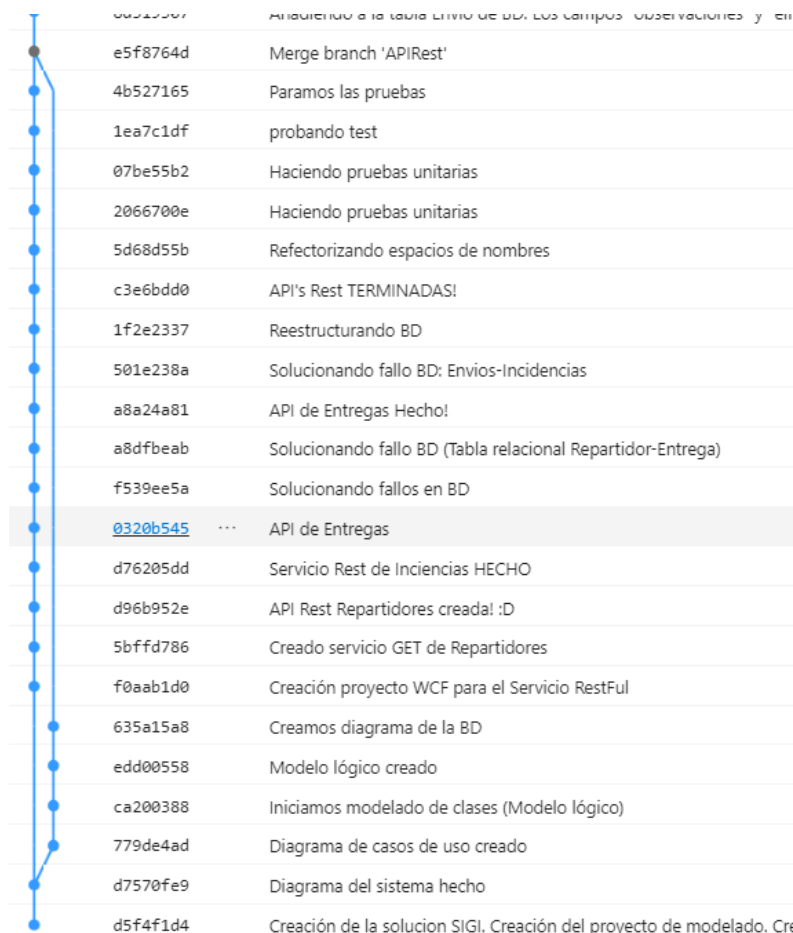


Figura 46 - Ejemplo de ramificación por funciones en GIT

**Scrum.** En la pestaña “Work” de Team Services podemos crear tareas e iteraciones para realizar la planificación de nuestro proyecto. Lo primero que hemos hecho es realizar las iteraciones de nuestro proyecto, en nuestro caso hemos realizado 6 iteraciones o Sprint:

- Sprint 1 – Fase de investigación







## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

- Sprint 2 – Planificación, análisis e instalación de las herramientas
- Sprint 3 – Modelo y diseño del sistema
- Sprint 4 – Desarrollo de la aplicación de gestión
- Sprint 5 – Desarrollo de la aplicación móvil
- Sprint 6 – Fase de prueba y producción

Cada uno de los Sprint contiene actividades y estas a su vez tareas. Las tareas tienen una dificultad o recursos temporales distintos según su complejidad. Así bien, nos es lo mismo crear la actividad para crear usuarios que una actividad para una de las funcionalidades principales de la aplicación como realizar las consultas de los envíos, véase la Figura 47/48:

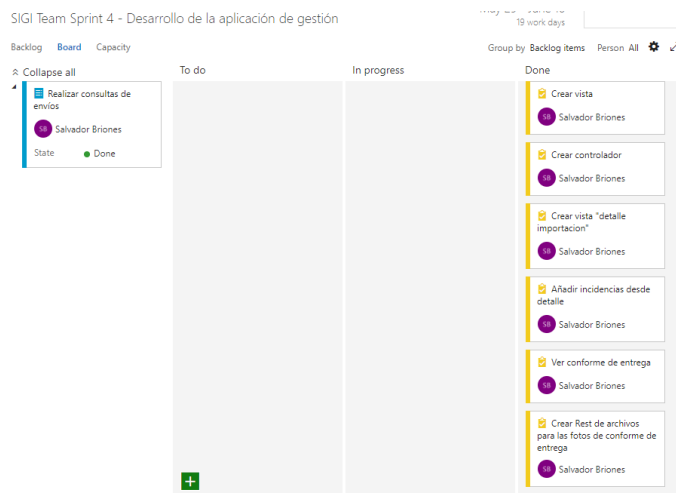


Figura 47 - Tareas de la actividad "Realizar consultas de envíos" desde Visual Studio Team Services

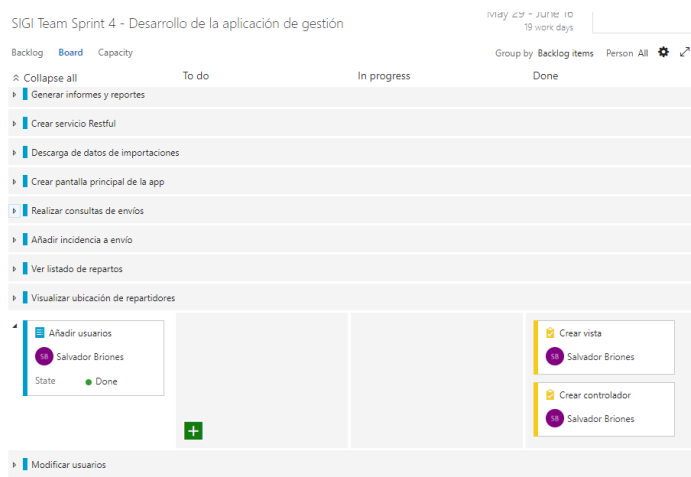


Figura 48 - Tareas de la actividad "Añadir usuarios" desde Visual Studio Team Services

Podemos ver que la complejidad de las actividades se rige por el número de tareas que las actividades contienen o simplemente por su dificultad. Cada actividad tiene un histórico de actividad, véase la Figura 49:



# SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

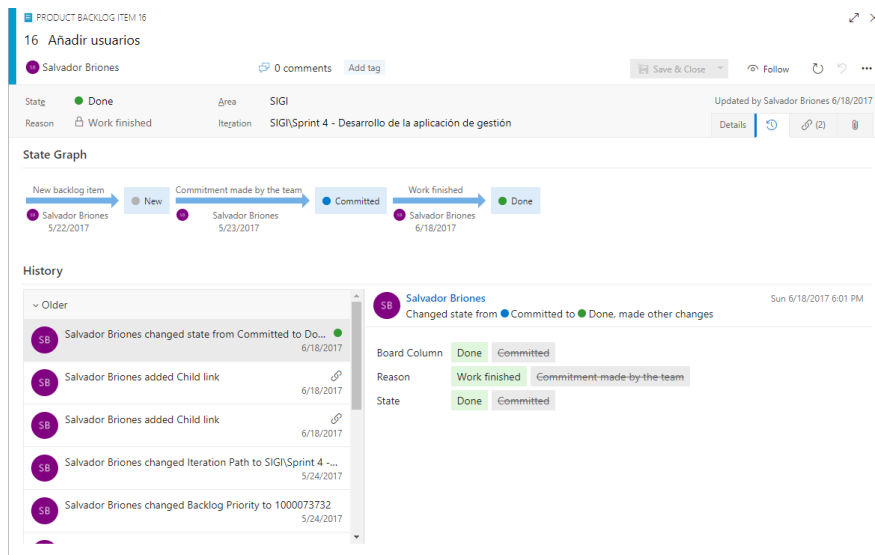


Figura 49 - Vista detalle de la actividad "Añadir usuarios"

Vemos quien ha intervenido en la actividad, los diferentes estados de la misma y cuando se produjeron.

Por último, de manera representativa podemos ver estadísticas cuantitativas de la planificación de nuestro proyecto, del estado de las actividades y de la velocidad en las que se han terminado tanto de cada iteración/Sprint o de todo el proyecto a nivel general. A continuación, en la Figura 50/51, podemos ver la velocidad en la cual se han realizado cada Sprint y el estado temporal de cada uno durante la realización del proyecto:

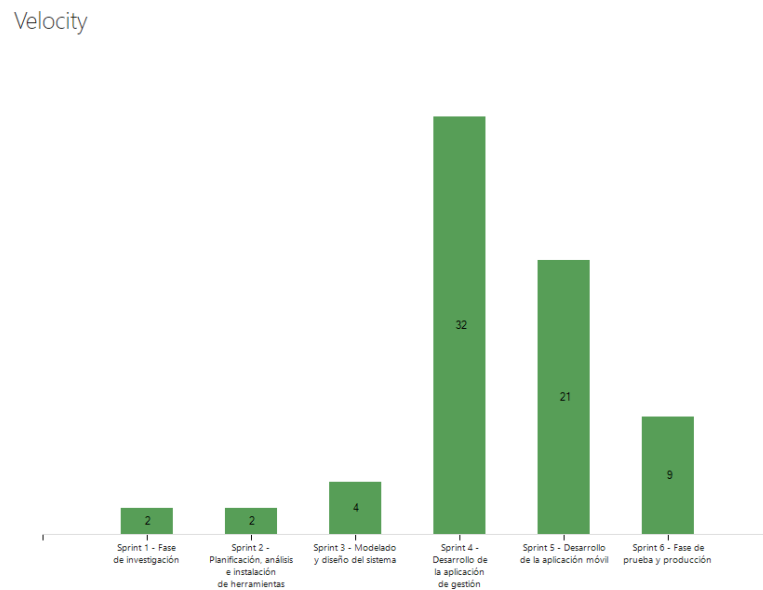


Figura 50 - Representación gráfica de la velocidad de cada Sprint del proyecto



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

Cumulative flow

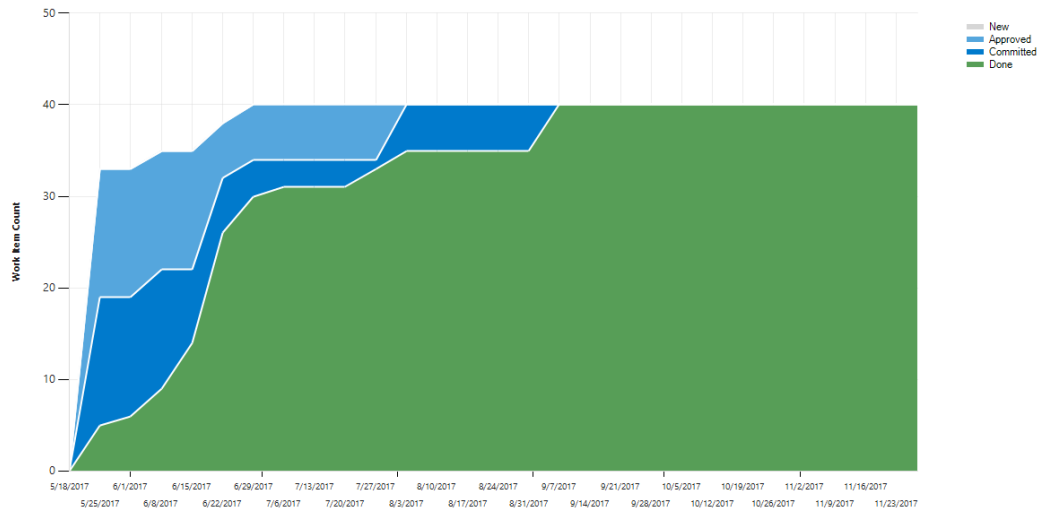


Figura 51 - Cumulative flow/Diagrama de flujo acumulativo de las actividades

La Figura 51 muestra un diagrama de flujo acumulativo, este diagrama es una representación fundamental que permite visualizar el esfuerzo realizado y el progreso del proyecto. Un diagrama La forma ideal que se vea un diagrama de flujo acumulativo es que ascienda uniformemente.

**🚚 Construcción y despliegue.** Desde la pestaña “Build and Release” podremos crear las distintas configuraciones y opciones para realizar las construcciones del proyecto en general y subproyectos. A su vez, en caso de que la construcción sea correcta y la necesidad los requiera podremos realizar la integración continua de nuestro proyecto en Azure. Cabe decir que nuestro proyecto está asociado y enlazado al repositorio y es donde obtiene los recursos de nuestro proyecto.

La construcción del proyecto se realiza de manera automática siempre que se haga un commit en la rama master, existe la posibilidad de configurar el proyecto mediante un Hashtag o etiqueta para cuando se llame a ese Hashtag desde el commit se realice también de forma automática la construcción siempre que se quiera. Las tareas de la construcción son las siguientes:

- **Descargar y restaurar los paquetes NuGet.**
- **Realizar la construcción de la solución de Visual Studio**
- **Realizar la construcción del proyecto Xamarin.Android**
- **Realizar los Tests**
- **Crear los artefactos de la construcción.**



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

En caso de que todo el proceso de construcción se haya realizado con éxito se procederá a realizar el despliegue en Azure de manera que la construcción e integración es continua de esta manera siempre tenemos el proyecto activo y con disponibilidad a cambios con tan solo realizar un cambio en el repositorio. En la Figura 52 mostraremos el panel de configuración de la construcción y despliegue del proyecto, podremos ver que se trata de una interfaz bastante amigable y fácil de usar:

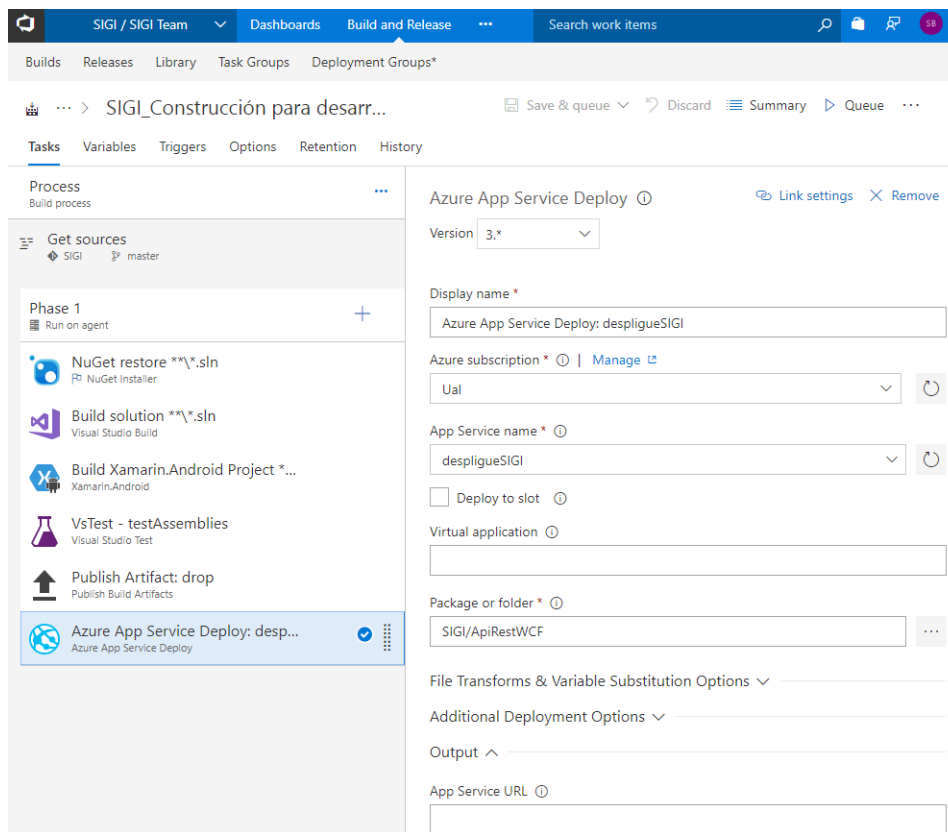


Figura 52 - Panel de configuración para la construcción automatizada del proyecto

Cabe resaltar que para que funcione la integración continua tuvo que realizar numerosas pruebas y configuraciones que, una vez hechas, supuso la automatización completa del sistema. Un paso importante es el de enlazar Team Services con Azure para ello nos iremos a la pestaña Configuración→Servicios y desde allí podremos añadir nuestra cuenta de Azure y el Plan que necesitaremos usar para nuestro despliegue automático.



#### 4.5. SQL Server 2014 Management Studio



Figura 53 - Logotipo SQL Server 2014 Management Studio

SQL Server 2014 Management Studio o SSMS es un entorno integrado para administrar cualquier infraestructura de SQL, desde SQL Server a SQL Database. SSMS proporciona herramientas para configurar, supervisar y administrar instancias de SQL. Use SSMS para implementar, supervisar y actualizar los componentes de capa de datos usados por las aplicaciones, además de compilar consultas y scripts.

Hemos usado SSMS para consultar, diseñar y administrar la base de datos de SIGI que se encontraba en Microsoft Azure.



Figura 54 - Ventana de acceso a la Base de datos en Microsoft Azure

Usando nuestras credenciales y el enlace de configuración que nos ha proporcionado Azure hemos configurado y conectado nuestro administrador de base de datos.



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

El acceso a nuestra base de datos es restringido y además de acceder con nuestras credenciales debemos abrir el Firewall de la base de datos con nuestra IP pública desde Azure. A continuación, mostraremos como lo hemos realizado desde Azure, Figura 55:

NOMBRE DE REGLA	IP INICIAL	IP FINAL
IP_Casa	90.68.108.158	90.68.108.158
IP_Casa_Madrid	79.159.211.132	79.159.211.132
IP_Nave	80.58.150.180	80.58.150.180
IP_Uni	150.214.223.70	150.214.223.70
IP_Uni2	80.27.27.166	80.27.27.166

Figura 55 - Añadir reglas de acceso para la base de datos en Azure

Una vez que hemos accedido ya podemos navegar por nuestra base de datos para ejecutar consultas, editar la estructura y datos de la base de datos.

### 4.6. Postman



Figura 56 - Logotipo de la aplicación Postman

Postman es una extensión gratuita para el navegador Google Chrome que permite probar servicios web fácilmente, basta con indicar la url, el método HTTP (POST, GET, etc.) y los parámetros de la petición.

Hemos usado esta herramienta para probar las peticiones creadas en nuestro servidor web de servicios Rest y probado las peticiones por HTTP.





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

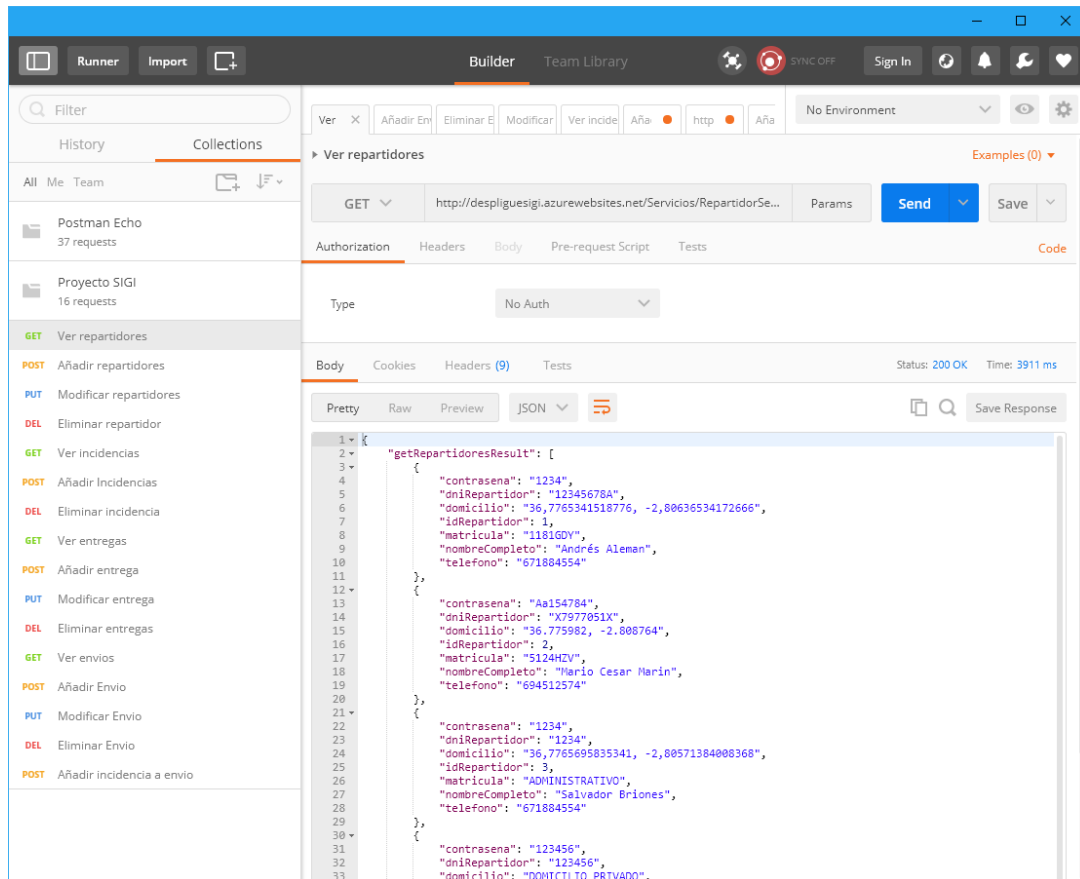


Figura 57 - Ejemplo de uso de Postman

Tal y como se aprecia en la Figura 57, hemos realizado una petición web por HTTP a nuestro servidor en Azure y nos ha respondido con la petición en formato JSON, estos datos serán formateados por nuestras aplicaciones cliente.





#### **4.7. Dispositivos de trabajo**

Para la realización de este trabajo, proyecto y memoria ha sido imprescindible el uso de mi ordenador personal cuyas características me han podido realizar con eficiencia y de forma eficaz las tareas a realizar.

Se trata de un ordenador portátil MSI GE70 2PE destinado comercialmente para jugar videojuegos pero que adquirí para realizar las labores académicas dada su potencia.

Las características del equipo son las siguientes:

- 🖨️ Procesador Intel® Core™ i7-4720HQ Processor (6M Cache, up to 3.60 GHz)
- 🖨️ Memoria RAM 8GB DDR3 (1 x 8GB)
- 🖨️ Disco duro 1TB (7200rpm) + sistema RAID0 512GB
- 🖨️ Almacenamiento óptico DVD SuperMulti
- 🖨️ Pantalla 17.3" Full HD (1920X1080), Anti-reflejo
- 🖨️ Controlador gráfico nVidia Geforce GTX860M, 2GB GDDR5
- 🖨️ Sistema operativo Windows 10 Education



*Figura 58 - Fotografía del equipo empleado para elaborar el proyecto*

Con este equipo se realizaron las pruebas de la aplicación de escritorio de SIGI.

Por otro lado, aunque contábamos con un emulador de Android en Xamarin, hemos utilizado un smartphone para realizar las pruebas físicas y comprobar que todas las funcionalidades era las deseadas.





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

Hemos usado un smartphone Samsung Core II, tiene las siguientes características Sistema operativo: Android 4.4.

### Pantalla:

- Tecnología: TFT.
- Tamaño: 4.5" (114.2mm).
- Resolución: 480 x 800 (WVGA).
- Número de colores: 262K.

### Procesador:

- Velocidad CPU: 1,2 Ghz.
- Tipo: Quad-Core.

### Memoria:

- RAM: 0.75GB.
- Interna: 4GB.
- Externa: MicroSD (hasta 64GB)..:

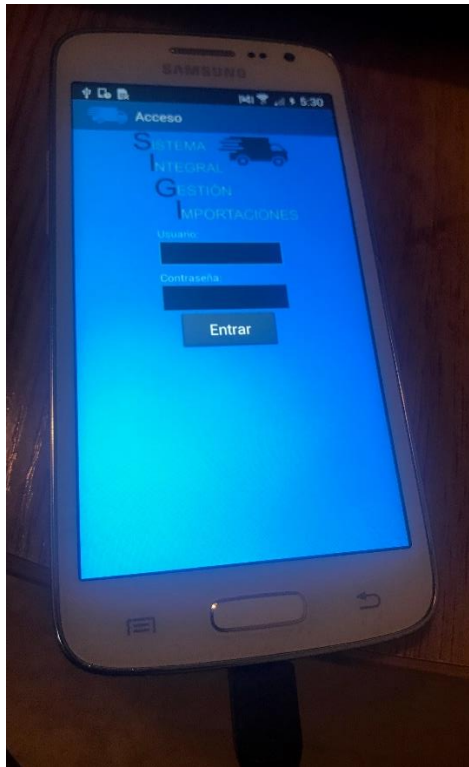


Figura 59 - Fotografía del Smartphone empleado



#### 4.8. Esquema general de tecnologías y herramientas del proyecto software

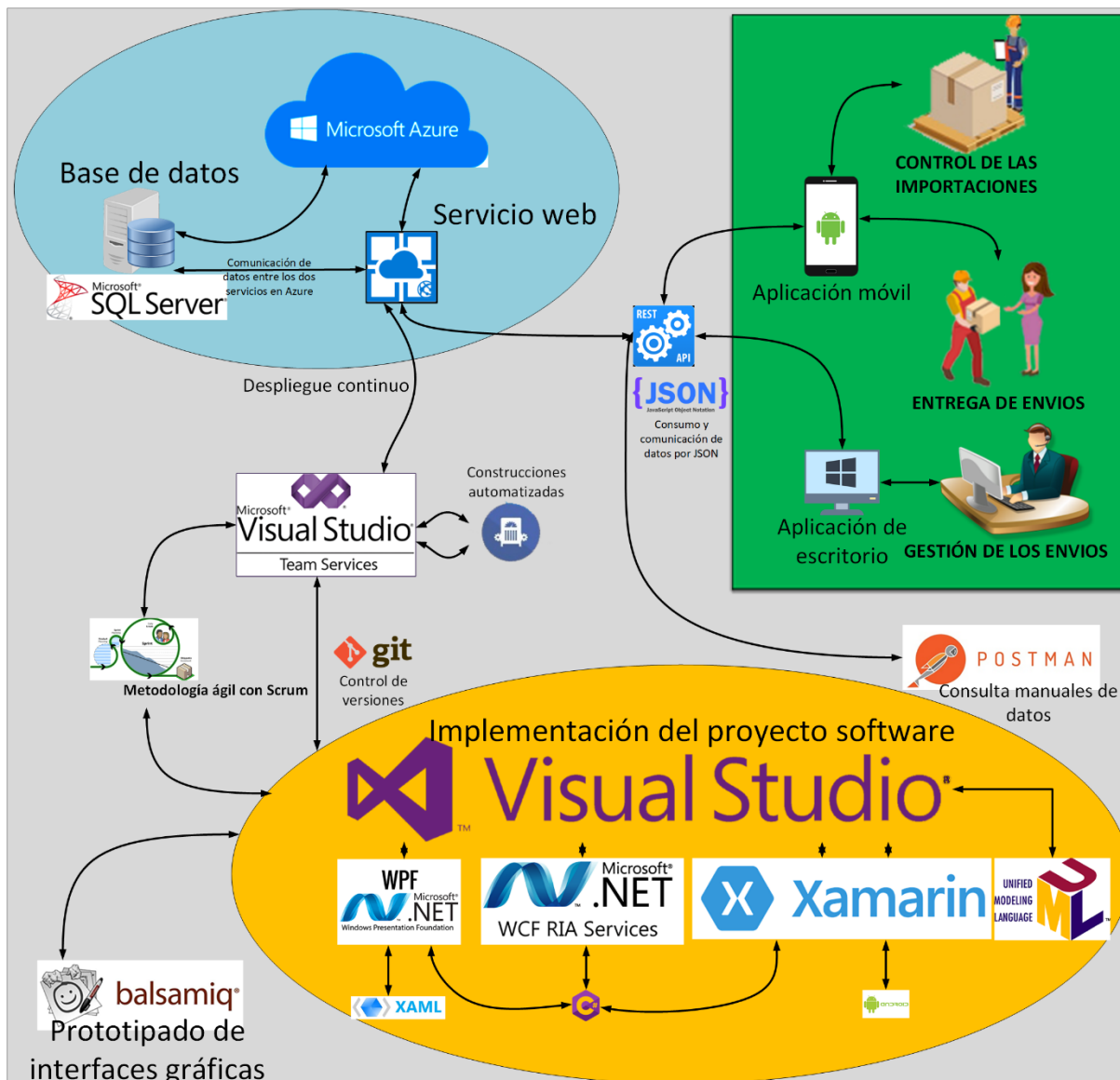


Figura 60 - Esquema general de las tecnologías y herramientas usadas en el proyecto

En la Figura 60 podemos ver como se agrupan las tecnologías y herramientas según la etapa del proyecto, en la esquina inferior izquierda tenemos el prototipado de las interfaces gráficas que va ligado a la implementación y desarrollo del proyecto software.

Vemos que las tecnologías de implementación y desarrollo como WPF, WCF, Xamarin y UML está integradas en Visual Studio y su lenguaje principal es C#.

Por otro lado, el servicio externo de Visual Studio Team Service sirve de intermediario con el servicio de Microsoft Azure en la nube y sus App Services.





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

Podemos ver también que la comunicación realizada entre la implementación y la construcción automatizada se realiza mediante el control de versiones GIT. JSON es utilizado como medio para la consulta de datos de nuestro servicio web Restful por parte de las aplicaciones cliente de escritorio y móvil.

Vemos en la parte superior la creación de los dos servicios de Azure, por un lado, un servidor de base de datos SQL Server y por otro la creación de un Plan de App Service para un servidor web Rest. El servidor web API resultante es un servicio Restful del que consumen las aplicaciones clientes, la aplicación de escritorio y la aplicación móvil.

Finalmente, las aplicaciones cliente creadas son dos: una aplicación de escritorio diseñada para la gestión de las importaciones y una aplicación móvil para el control y entrega de las importaciones.





# 5. DESARROLLO DE SIGI

En anteriores capítulos hemos recopilado todos los recursos y conocimientos que necesitamos para desarrollar el sistema. En este capítulo detallaremos todas las fases de desarrollo software.

## 5.1. Análisis y requisitos

En el capítulo 1.2 Objetivos, se enumeraron todos los objetivos a realizar. A continuación, detallaremos estos objetivos:

### 5.1.1. Objetivos

<b>Objetivo-01</b>	<b>Acceso con credenciales. Aplicación móvil</b>
<b>Versión</b>	1.0
<b>Autor</b>	Salvador Briones Rosales
<b>Descripción</b>	El usuario podrá acceder a la aplicación con sus datos de usuario.
<b>Comentarios</b>	En caso de no poder acceder, la aplicación mostrará el motivo del error.
<b>Importancia</b>	Muy alta

<b>Objetivo-02</b>	<b>Llegada de importaciones. Aplicación móvil</b>
<b>Versión</b>	1.0
<b>Autor</b>	Salvador Briones Rosales
<b>Descripción</b>	Mediante un código de barras se podrá realizar la lectura de los envíos que llegan al almacén.
<b>Comentarios</b>	El dispositivo móvil deberá tener una cámara para realizar la captura de los códigos de barra. La aplicación deberá analizar el código y verificar si la lectura no se ha realizado anteriormente.
<b>Importancia</b>	Alta





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

<b>Objetivo-03</b>	<b>Actualizar llegadas. Aplicación móvil</b>
<b>Versión</b>	1.0
<b>Autor</b>	Salvador Briones Rosales
<b>Descripción</b>	El sistema deberá permitir subir las actualizaciones de las llegadas de importaciones (objetivo 02).
<b>Comentarios</b>	La aplicación deberá alertar al usuario si el proceso se ha completado correctamente o en caso contrario de porque no se ha logrado.
<b>Importancia</b>	Alta

<b>Objetivo-04</b>	<b>Asignar a reparto. Aplicación móvil</b>
<b>Versión</b>	1.0
<b>Autor</b>	Salvador Briones Rosales
<b>Descripción</b>	El sistema deberá permitir al usuario asignar mercancía que llevará en ruta. Una vez terminada dicha asignación se enviarán los datos al sistema para actualizar el sistema e imprimir el detalle de la mercancía que lleva.
<b>Comentarios</b>	Durante la asignación podrá ver lo detalles de la mercancía o eliminarla del listado de asignación.
<b>Importancia</b>	Alta

<b>Objetivo-05</b>	<b>Entregar envío. Aplicación móvil</b>
<b>Versión</b>	1.0
<b>Autor</b>	Salvador Briones Rosales
<b>Descripción</b>	El sistema deberá permitir al usuario actualizar el envío que llevaba asignado.
<b>Comentarios</b>	El envío se actualizará con la conformidad del destinatario con su firma y el número de identificación. En caso contrario, podrá actualizar el envío alguna incidencia predefinida en el sistema.





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

<b>Importancia</b>	Alta
--------------------	------

<b>Objetivo-06</b>	<b>Geolocalización. Aplicación móvil</b>
<b>Versión</b>	1.0
<b>Autor</b>	Salvador Briones Rosales
<b>Descripción</b>	El dispositivo móvil enviará en todo momento la ubicación exacta del chofer.
<b>Comentarios</b>	Cuando el repartidor sea geolocalizado, se mandará una alerta en la pantalla para que tenga constancia.
<b>Importancia</b>	Media

<b>Objetivo-07</b>	<b>Descarga de importaciones. Aplicación de escritorio</b>
<b>Versión</b>	1.0
<b>Autor</b>	Salvador Briones Rosales
<b>Descripción</b>	El sistema deberá permitir descargar los datos de las importaciones que se han facturado y están pendientes de llegar.
<b>Comentarios</b>	Cuando la tarea se haya hecho efectiva mostrará una alerta con el número de envíos integrados y un resumen de ellos.
<b>Importancia</b>	Alta

<b>Objetivo-08</b>	<b>Consultar importaciones. Aplicación de escritorio</b>
<b>Versión</b>	1.0
<b>Autor</b>	Salvador Briones Rosales
<b>Descripción</b>	El sistema deberá permitir realizar consulta de importaciones.
<b>Comentarios</b>	Se podrá realizar la consulta por número de envío o por filtrar el contenido del nombre de destinatario.
<b>Importancia</b>	Alta

<b>Objetivo-09</b>	<b>Gestión de incidencias. Aplicación de escritorio</b>
--------------------	---





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

<b>Versión</b>	1.0
<b>Autor</b>	Salvador Briones Rosales
<b>Descripción</b>	El sistema deberá realizar un seguimiento de las incidencias y envíos pendientes.
<b>Comentarios</b>	El sistema permitirá filtrar las incidencias entre dos fechas y podremos también filtrar el tipo de incidencia.
<b>Importancia</b>	Alta

<b>Objetivo-10</b>	<b>Listado de repartos. Aplicación de escritorio</b>
<b>Versión</b>	1.0
<b>Autor</b>	Salvador Briones Rosales
<b>Descripción</b>	El sistema deberá permitir consultar listado de reparto de cada repartidor.
<b>Comentarios</b>	Se podrá ver también el número de bultos totales y peso total que lleva en ruta.
<b>Importancia</b>	Alta

<b>Objetivo-11</b>	<b>Generar informes y reportes. Aplicación de escritorio</b>
<b>Versión</b>	1.0
<b>Autor</b>	Salvador Briones Rosales
<b>Descripción</b>	El sistema deberá permitir generar un informe de la mercancía que se ha repartido en el día. También se podrá generar reportes relativos a los diferentes estados de la mercancía.
<b>Comentarios</b>	Estos estados son relativos las incidencias pendientes. Se generará un archivo PDF que se podrá imprimir.
<b>Importancia</b>	Alta

<b>Objetivo-12</b>	<b>Geolocalización de repartidores. Aplicación de escritorio</b>
<b>Versión</b>	1.0
<b>Autor</b>	Salvador Briones Rosales





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

<b>Descripción</b>	El sistema deberá permitir ver en tiempo real la ubicación en tiempo real del repartidor.
<b>Comentarios</b>	Se mostrará una imagen en el mapa
<b>Importancia</b>	Alta

<b>Objetivo-13</b>	<b>Gestión de usuarios. Aplicación de escritorio</b>
<b>Versión</b>	1.0
<b>Autor</b>	Salvador Briones Rosales
<b>Descripción</b>	El sistema deberá permitir añadir, modificar, eliminar y consultar los datos de un usuario.
<b>Comentarios</b>	Esta función sólo estará habilitada por el encargado. Un usuario podrá ser repartidor, administrativo o encargado.
<b>Importancia</b>	Alta

<b>Objetivo-14</b>	<b>Rectificación y anulación de envíos. Aplicación de escritorio</b>
<b>Versión</b>	1.0
<b>Autor</b>	Salvador Briones Rosales
<b>Descripción</b>	El sistema deberá permitir modificar y eliminar datos relativos a un envío.
<b>Comentarios</b>	Esta función sólo estará habilitada por el encargado. Cada vez que se realice una modificación o eliminación se deberá especificar el motivo.
<b>Importancia</b>	Alta

### 5.1.2. Actores

En el sistema intervendrá actores o usuarios con un rol específico. A continuación, los detallaremos:

<b>Actor-01</b>	<b>Repartidor</b>
-----------------	-------------------







## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

<b>Descripción</b>	Este usuario tendrá acceso sólo a la aplicación móvil. Es la persona responsable del control en almacén y entrega de la mercancía en reparto.
<b>Requisitos asociados</b>	

<b>Actor-02</b>	<b>Auxiliar administrativo</b>
<b>Descripción</b>	Este usuario sólo tiene acceso a la aplicación de escritorio. Se encargará de realizar la gestión de SIGI.
<b>Requisitos asociados</b>	

<b>Actor-03</b>	<b>Encargado</b>
<b>Descripción</b>	Se trata de la autoridad máxima dentro del sistema informático. Tiene acceso a todo el sistema y puede hacer uso de todas sus funciones.
<b>Requisitos asociados</b>	





### 5.1.3. Requisitos funcionales

A continuación, mostraremos el diagrama de casos de uso representando toda la funcionalidad del sistema:

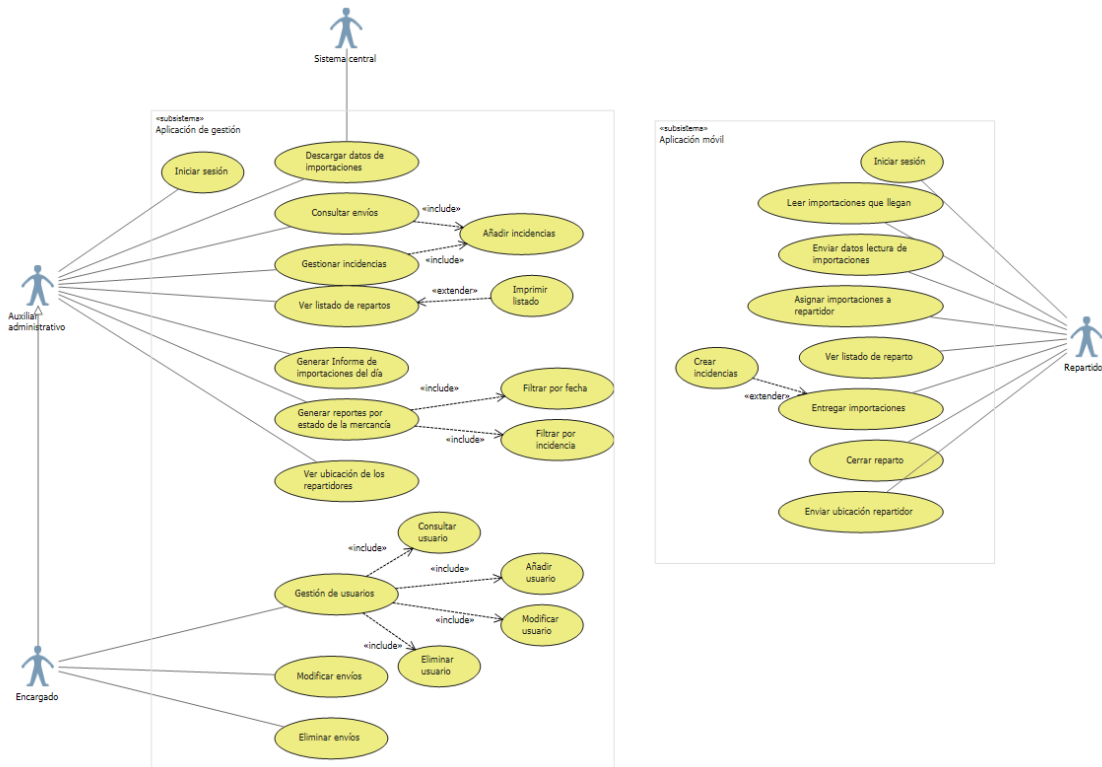


Figura 61 - Diagrama de casos de uso de SIGI

En el diagrama podemos ver dos subsistemas, es decir, el proyecto con la aplicación móvil y el proyecto con la aplicación de escritorio. Aparecen los tres actores del sistema y sus funcionalidades derivadas.

Podemos ver también que existe una herencia de funciones. El actor Encargado podrá hacer todo lo que hace el actor Auxiliar Administrativo.

Detallaremos el diagrama de casos de uso en tablas especificando de manera formal su función. Primeramente, comentaremos los requisitos funcionales del subsistema de gestión o aplicación de escritorio.





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

<b>RF-01</b>	<b>Descargar datos de importaciones</b>
<b>Actor asociado</b>	Actor-02, Actor-03
<b>Descripción</b>	El usuario podrá conectarse al servidor central y descargar los datos de importaciones/envíos que están por llegar.
<b>Precondición</b>	El usuario deberá estar autenticado y en la ventana “Ventana principal”.
<b>Secuencia normal</b>	<ul style="list-style-type: none"><li>▪ Paso 1: El usuario accede al menú “1.-Descargar importaciones”.</li><li>▪ Paso 2: En la nueva ventana “Descarga de importaciones” pulsamos en el botón “Descargar importaciones”.</li><li>▪ Paso 3: Pulsamos en el botón habilitado “Integrar importaciones”.</li><li>▪ Paso 4: Confirmamos la ventana emergente con el número de importaciones a integrar.</li></ul>
<b>Postcondición</b>	El usuario verá una ventana informativa confirmando que las importaciones se han integrado y se nos redirigirá al menú principal.
<b>Excepción</b>	Si no existieran importaciones para integrar se le informaría al usuario con una ventana emergente.
<b>Comentarios</b>	Requisitos relacionados: RF-05

<b>RF-02</b>	<b>Consultar envíos</b>
<b>Actor asociado</b>	Actor-02, Actor-03
<b>Descripción</b>	El usuario podrá consultar los envíos por número de envío o por palabra clave contenida en el nombre del destinatario
<b>Precondición</b>	El usuario deberá estar autenticado y en la ventana “Ventana principal”.
<b>Secuencia normal</b>	<ul style="list-style-type: none"><li>▪ Paso 1: El usuario pulsa desde el menú principal al botón “2.- Consultar importaciones”.</li><li>▪ Paso 2: En la nueva ventana “Consulta de importaciones” introduciremos el número de envío o las palabras o letras contenidas en el nombre del destinatario.</li></ul>





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

	<ul style="list-style-type: none"><li>▪ Paso 3: Pulsaremos sobre la lupa (buscar) para realizar la consulta.</li><li>▪ Paso 4: El usuario verá un listado del resultado de la consulta realizada, estos resultados podrán ser con el envío relacionado en base al número de envío o un listado con los envíos relacionados en base al criterio de nombre de destinatario.</li><li>▪ Paso 5: En la nueva ventana llamada “Detalle importación” que ha surgido de la selección el envío dentro del listado del resultado de la consulta podrá observar los datos relativos al envío consultado.</li></ul>
<b>Postcondición</b>	El usuario podrá añadir incidencias al envío.
<b>Excepción</b>	Si no se introdujera ningún criterio en los campos de búsqueda (número de envío o nombre de destinatario) el sistema alertará la excepción al usuario mediante una ventana emergente. Si una vez introducidos los criterios de búsqueda no se encontrará ningún envío relacionado el sistema alertará la excepción al usuario mediante una ventana emergente.
<b>Comentarios</b>	Requisitos relacionados: RF-03, RF-04, RF-05

<b>RF-03</b>	<b>Añadir incidencias</b>
<b>Actor asociado</b>	Actor-02, Actor-03
<b>Descripción</b>	El usuario podrá añadir incidencias a los envíos.
<b>Precondición</b>	El usuario deberá estar en la ventana “Detalle importación” desde la que se accede desde la ventana “2.-Consulta de importaciones” o la ventana “3.-Gestión de incidencias”.
<b>Secuencia normal</b>	<ul style="list-style-type: none"><li>▪ Paso 1: El usuario desde la ventana “Detalle importación” podrá añadir incidencias al envío pulsando el botón “Añadir incidencia”.</li><li>▪ Paso 2: Se mostrará una ventana emergente en la cual el usuario podrá añadir un tipo de incidencia y una descripción de la incidencia si fuera conveniente, el usuario pulsará el botón “Añadir” para añadir la incidencia al envío consultado.</li></ul>





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

<b>Postcondición</b>	El usuario podrá ver la incidencia nueva dentro del listado de incidencias del envío consultado.
<b>Excepción</b>	Si el envío/importación está entregado al pulsar el botón “Añadir incidencia” el sistema alertará la excepción al usuario mediante una ventana emergente. Si el envío se encuentra en reparto, al pulsar el botón “Añadir incidencia” el sistema alertará la excepción al usuario mediante una ventana emergente.
<b>Comentarios</b>	Requisitos relacionados: RF-02, RF-04

<b>RF-04</b>	<b>Gestionar incidencias</b>
<b>Actor asociado</b>	Actor-02, Actor-03
<b>Descripción</b>	El usuario podrá consultar todas las incidencias existentes en el sistema. Podrá filtrar la consulta entre dos fechas y por tipo de incidencia.
<b>Precondición</b>	El usuario deberá estar autenticado y en la ventana “Ventana principal”.
<b>Secuencia normal</b>	<ul style="list-style-type: none"><li>▪ Paso 1: El usuario pulsa desde el menú principal en el botón “3.-Gestión de incidencias”.</li><li>▪ Paso 2: El usuario inserta los criterios de consulta si lo prefiere: Desde y hasta una fecha y/o selecciona por filtro el tipo de incidencias a buscar en los envíos. En caso de no seleccionar ningún criterio se realizará una consulta de todos los envíos que contienen incidencias.</li><li>▪ Paso 3: Se listarán todos los envíos con los criterios de búsqueda seleccionados.</li></ul>
<b>Postcondición</b>	El usuario podrá ver el detalle de cualquier envío encontrado en el resultado de la búsqueda.
<b>Excepción</b>	Si no existieran envíos con incidencias en la búsqueda el sistema alertará la excepción al usuario mediante una ventana emergente informando la excepción de consulta.
<b>Comentarios</b>	Requisitos relacionados: RF-02, RF-03, RF-05





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

<b>RF-05</b>	<b>Iniciar sesión</b>
<b>Actor asociado</b>	Actor-02, Actor-03
<b>Descripción</b>	El usuario podrá acceder al sistema con sus credenciales.
<b>Precondición</b>	El usuario deberá tener credenciales de acceso al sistema. Las credenciales son expedidas por el Actor-03.
<b>Secuencia normal</b>	<ul style="list-style-type: none"><li>▪ Paso 1: El usuario ingresa sus credenciales de acceso.</li><li>▪ Paso 2: El sistema mostrará una ventana emergente dando la bienvenida al usuario.</li><li>▪ Paso 3: El usuario será redirigido a su ventana "Ventana principal".</li></ul>
<b>Postcondición</b>	El usuario verá la vista a la que tiene acceso. El Actor-02 tiene una vista diferente del Actor-03.
<b>Excepción</b>	<p>Si las credenciales son incorrectas el sistema alertará la excepción al usuario mediante una ventana emergente informando el error de acceso.</p> <p>Si el usuario Actor-01 intenta acceder, el sistema alertará la excepción al usuario mediante una ventana emergente informando la restricción de acceso.</p>
<b>Comentarios</b>	La ventana "Ventana Principal" será una vista u otra dependiendo del actor que sea. Existen una vista para el Actor-02 y otra para el Actor-03.

<b>RF-06</b>	<b>Ver listado de repartos</b>
<b>Actor asociado</b>	Actor-02, Actor-03
<b>Descripción</b>	El usuario podrá visualizar el listado de repartos de cada repartidor.
<b>Precondición</b>	El usuario deberá estar autenticado y en la ventana "Ventana Principal".
<b>Secuencia normal</b>	<ul style="list-style-type: none"><li>▪ Paso 1: El usuario pulsa desde el menú principal en el botón "4.-Listado de repartos".</li></ul>





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

	<ul style="list-style-type: none"><li>▪ Paso 2: En la nueva ventana “Ver listado de repartos” el usuario seleccionará el repartidor de la lista “Seleccione repartidor” y a continuación pulsará en Actualizar.</li><li>▪ Paso 3: El usuario podrá visualizar un listado con los envíos que lleva en reparto y que aún no se han entregado.</li></ul>
<b>Postcondición</b>	El usuario tendrá la posibilidad de imprimir el listado de reparto del repartidor seleccionado.
<b>Excepción</b>	Si el repartidor seleccionado no tiene envíos asignados el sistema alertará la excepción al usuario mediante una ventana emergente informando la excepción.
<b>Comentarios</b>	Requisitos relacionados: RF-05, RF-07

<b>RF-07</b>	<b>Imprimir listado</b>
<b>Actor asociado</b>	Actor-02, Actor-03
<b>Descripción</b>	El usuario podrá imprimir un listado de los repartos de un repartidor seleccionado.
<b>Precondición</b>	El usuario deberá estar en la ventana “4.-Listado de repartos” y el repartidor seleccionado deberá tener envíos en reparto.
<b>Secuencia normal</b>	<ul style="list-style-type: none"><li>▪ Paso 1: El usuario pulsará en el botón “Imprimir listado”</li><li>▪ Paso 2: Desde el visor predeterminado de PDF el usuario podrá imprimir el documento generado desde la aplicación.</li></ul>
<b>Postcondición</b>	El usuario deberá tener instalado algún visor de documentos PDF en el sistema.
<b>Excepción</b>	Si el repartidor seleccionado no tiene envíos asignados el sistema alertará la excepción al usuario mediante una ventana emergente informando la excepción.
<b>Comentarios</b>	Requisitos relacionados: RF-05, RF-06

<b>RF-08</b>	<b>Generar informe de importaciones del día</b>
<b>Actor asociado</b>	Actor-02, Actor-03
<b>Descripción</b>	El usuario podrá generar un documento con todas las actualizaciones de envíos realizados en un día específico.





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

<b>Precondición</b>	El usuario deberá estar autenticado y en la ventana “Ventana Principal”.
<b>Secuencia normal</b>	<ul style="list-style-type: none"><li>▪ Paso 1: El usuario pulsa desde el menú principal en el botón “5.-Informes y reportes”.</li><li>▪ Paso 2: En la nueva ventana “Generación de informes y reportes” el usuario podrá ver la sección “Generación de informes”.</li><li>▪ Paso 3: El usuario seleccionará la fecha en la cual quiere que se genere el informe de las actualizaciones de importaciones y pulsará en el botón “Generar informe”.</li><li>▪ Paso 4: El sistema creará una nueva ventana con el informe en forma de documento PDF.</li></ul>
<b>Postcondición</b>	El usuario deberá tener instalado algún visor de documentos PDF en el sistema.
<b>Excepción</b>	Si el día seleccionado no contiene ninguna actualización de importaciones realizada se generará un documento PDF en blanco.
<b>Comentarios</b>	Requisitos relacionados: RF-05, RF-07

<b>RF-09</b>	<b>Generar reportes por estado de la mercancía</b>
<b>Actor asociado</b>	Actor-02, Actor-03
<b>Descripción</b>	El usuario podrá generar un documento que liste las importaciones según su estado final de incidencia entre dos fechas.
<b>Precondición</b>	El usuario deberá estar autenticado y en la ventana “Ventana Principal”.
<b>Secuencia normal</b>	<ul style="list-style-type: none"><li>▪ Paso 1: El usuario pulsa desde el menú principal en el botón “5.-Informes y reportes”.</li><li>▪ Paso 2: En la nueva ventana “Generación de informes y reportes” el usuario podrá ver la sección “Generación de reportes”.</li><li>▪ Paso 3: El usuario deberá seleccionar las fechas en la cual quiere que se genere el reporte y seleccionar el filtro de</li></ul>







## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

	<p>incidencias. Finalmente, pulsará en el botón “Generar reporte”.</p> <ul style="list-style-type: none"><li>▪ Paso 4: El sistema creará una nueva ventana con el informe en forma de documento PDF.</li></ul>
<b>Postcondición</b>	El usuario deberá tener instalado algún visor de documentos PDF en el sistema.
<b>Excepción</b>	<p>Si no se ha introducido ninguna fecha en la consulta el sistema alertará la excepción al usuario mediante una ventana emergente informando la excepción.</p> <p>Si las fechas introducidas no son coherentes, la fecha “desde” es posterior a la fecha “hasta” el sistema alertará la excepción al usuario mediante una ventana emergente informando la excepción.</p> <p>Si entre las fechas seleccionadas y el filtro seleccionado no contiene ninguna actualización de importaciones se generará un documento PDF en blanco.</p>
<b>Comentarios</b>	Requisitos relacionados: RF-05, RF-06

<b>RF-10</b>	<b>Ver ubicación de los repartidores</b>
<b>Actor asociado</b>	Actor-02, Actor-03
<b>Descripción</b>	El usuario podrá ver en tiempo real la ubicación de los repartidores.
<b>Precondición</b>	El usuario deberá estar autenticado y estar ubicado en la ventana “Ventana Principal”. El repartidor debe tener iniciada la aplicación móvil.
<b>Secuencia normal</b>	<ul style="list-style-type: none"><li>▪ Paso 1: El usuario pulsa desde el menú principal en el botón “6.-Ubicación de repartidores”.</li><li>▪ Paso 2: En la nueva ventana “Ubicación repartidores” el usuario podrá seleccionar un repartidor dentro del listado “Seleccione repartidor” y posteriormente en el botón “Actualizar”.</li><li>▪ Paso 3: El usuario verá una imagen con la última actualización de geolocalización del repartidor.</li></ul>





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

<b>Postcondición</b>	El repartidor deberá haber iniciado la aplicación móvil.
<b>Excepción</b>	
<b>Comentarios</b>	Requisitos relacionados: RF-05 Siempre se mostrará la última posición del repartidor.

<b>RF-11</b>	<b>Modificar envíos</b>
<b>Actor asociado</b>	Actor-03
<b>Descripción</b>	El usuario podrá realizar una rectificación de los datos de una importación
<b>Precondición</b>	El usuario deberá estar autenticado y estar ubicado en la ventana "Ventana Principal". El usuario deberá ser un Actor-03.
<b>Secuencia normal</b>	<ul style="list-style-type: none"><li>▪ Paso 1: El usuario pulsa desde el menú principal en el botón "7.-Rectificación y anulación".</li><li>▪ Paso 2: En la nueva ventana "Rectificación y anulación de importaciones" el usuario deberá introducir el número de envío y pulsar el botón "buscar" o la imagen "lupa".</li><li>▪ Paso 3: El usuario modificará todos los datos del envío que vea pertinentes.</li><li>▪ Paso 4: El usuario deberá insertar en el cuadro "Motivo" los comentarios por los que ha tenido que realizar el cambio.</li><li>▪ Paso 5: El usuario pulsará en el botón "Modificar".</li><li>▪ Paso 6: El sistema mostrará una ventana emergente para confirma la acción.</li><li>▪ Paso 7: El sistema confirmará la modificación.</li></ul>
<b>Postcondición</b>	Se modifican los datos de una importación.
<b>Excepción</b>	Si el usuario introduce el número de envío erróneo o inexistente el sistema alertará la excepción al usuario mediante una ventana emergente informando la excepción. Si el usuario no introduce un comentario en el recuadro "Motivo" el sistema alertará la excepción al usuario mediante una ventana emergente informando la excepción.





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

	Si el envío se encuentra en reparto el sistema alertará la excepción al usuario mediante una ventana emergente informando la excepción.
<b>Comentarios</b>	Requisitos relacionados: RF-05 El usuario deberá ser de tipo Actor-03

<b>RF-12</b>	<b>Eliminar envíos</b>
<b>Actor asociado</b>	Actor-03
<b>Descripción</b>	El usuario podrá eliminar una importación/envío.
<b>Precondición</b>	El usuario deberá estar autenticado y estar ubicado en la ventana "Ventana Principal". El usuario deberá ser un Actor-03.
<b>Secuencia normal</b>	<ul style="list-style-type: none"><li>▪ Paso 1: El usuario pulsa desde el menú principal en el botón "7.-Rectificación y anulación".</li><li>▪ Paso 2: En la nueva ventana "Rectificación y anulación de importaciones" el usuario deberá introducir el número de envío y pulsar el botón "buscar" o la imagen "lupa".</li><li>▪ Paso 3: El usuario deberá insertar en el cuadro "Motivo" los comentarios por los que ha tenido que realizar el cambio.</li><li>▪ Paso 4: El usuario pulsará en el botón "Eliminar".</li><li>▪ Paso 6: El sistema mostrará una ventana emergente para confirma la acción.</li><li>▪ Paso 7: El sistema confirmará la eliminación del envío.</li></ul>
<b>Postcondición</b>	Se eliminará una importación/envío.
<b>Excepción</b>	<p>Si el usuario introduce el número de envío erróneo o inexistente el sistema alertará la excepción al usuario mediante una ventana emergente informando la excepción.</p> <p>Si el usuario no introduce un comentario en el recuadro "Motivo" el sistema alertará la excepción al usuario mediante una ventana emergente informando la excepción.</p> <p>Si el envío se encuentra en reparto el sistema alertará la excepción al usuario mediante una ventana emergente informando la excepción.</p>





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

<b>Comentarios</b>	Requisitos relacionados: RF-05. El usuario deberá ser de tipo Actor-03
--------------------	--

<b>RF-13</b>	<b>Gestión de usuarios. Consultar usuario</b>
<b>Actor asociado</b>	Actor-03
<b>Descripción</b>	El usuario podrá ver el listado de todos los usuarios
<b>Precondición</b>	El usuario deberá estar autenticado y estar ubicado en la ventana "Ventana Principal". El usuario deberá ser un Actor-03.
<b>Secuencia normal</b>	<ul style="list-style-type: none"><li>▪ Paso 1: El usuario pulsa desde el menú principal en el botón "8.-Gestión de usuarios".</li><li>▪ Paso 2: En la nueva ventana "Gestión de usuarios" el usuario verá el listado de los usuarios registrados en el sistema.</li><li>▪ Paso 3: El usuario deberá seleccionar el registro del usuario que quiera detallar</li><li>▪ Paso 4: El sistema mostrará al usuario otra ventana emergente con los datos del usuario.</li></ul>
<b>Postcondición</b>	-
<b>Excepción</b>	-
<b>Comentarios</b>	Requisitos relacionados: RF-05. El usuario deberá ser de tipo Actor-03

<b>RF-14</b>	<b>Gestión de usuarios. Añadir usuario</b>
<b>Actor asociado</b>	Actor-03
<b>Descripción</b>	El usuario podrá añadir un usuario nuevo al sistema.
<b>Precondición</b>	El usuario deberá estar autenticado y estar ubicado en la ventana "Ventana Principal". El usuario deberá ser un Actor-03.
<b>Secuencia normal</b>	<ul style="list-style-type: none"><li>▪ Paso 1: El usuario pulsa desde el menú principal en el botón "8.-Gestión de usuarios".</li><li>▪ Paso 2: En la nueva ventana "Gestión de usuarios" el usuario pulsará sobre el botón "Añadir usuario".</li><li>▪ Paso 3: El sistema mostrará al usuario una ventana emergente donde podrá insertar los datos del usuario nuevo.</li></ul>





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

	<p>Tendrá la opción de seleccionar el nuevo tipo de usuario: Repartidor, Administrativo y Encargado (Actor-01, Actor-02 y Actor-03 respectivamente).</p> <ul style="list-style-type: none"><li>▪ Paso 4: El usuario pulsará el botón “Añadir usuario”.</li><li>▪ Paso 5: El sistema nos redirigiera al listado de usuarios en la ventana “Gestión de usuarios” con el nuevo usuario añadido.</li></ul>
<b>Postcondición</b>	Se añadirá un usuario al sistema.
<b>Excepción</b>	<p>Si el usuario no ingresa un campo del formulario el sistema alertará la excepción al usuario mediante una ventana emergente informando la falta de datos.</p> <p>Si la contraseña no coincide con la repetición de la contraseña el sistema alertará la excepción al usuario mediante una ventana emergente informando la no coincidencia de las contraseñas.</p>
<b>Comentarios</b>	Requisitos relacionados: RF-05. El usuario deberá ser de tipo Actor-03

<b>RF-15</b>	<b>Gestión de usuarios. Modificar usuario</b>
<b>Actor asociado</b>	Actor-03
<b>Descripción</b>	El usuario podrá modificar un usuario del sistema.
<b>Precondición</b>	El usuario deberá estar autenticado y estar ubicado en la ventana “Ventana Principal”. El usuario deberá ser un Actor-03.
<b>Secuencia normal</b>	<ul style="list-style-type: none"><li>▪ Paso 1: El usuario pulsa desde el menú principal en el botón “8.-Gestión de usuarios”.</li><li>▪ Paso 2: En la nueva ventana “Gestión de usuarios” el usuario verá el listado de los usuarios registrados en el sistema.</li><li>▪ Paso 3: El usuario deberá seleccionar el registro del usuario que quiera modificar.</li><li>▪ Paso 4: El sistema mostrará al usuario una ventana emergente donde podrá modificar los datos del usuario.</li><li>▪ Paso 5: El usuario pulsará en el botón “Modificar usuario” para registrar los cambios.</li></ul>





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

	<ul style="list-style-type: none"><li>▪ Paso 6: El sistema mostrará una ventana emergente confirmando que los cambios se han realizado correctamente.</li><li>▪ Paso 7: El sistema nos redirigiera al listado de usuarios en la ventana "Gestión de usuarios" con el usuario modificado.</li></ul>
<b>Postcondición</b>	Se modificará un usuario del sistema.
<b>Excepción</b>	<p>Si el usuario no ingresa un campo del formulario el sistema alertará la excepción al usuario mediante una ventana emergente informando la falta de datos.</p> <p>Si la contraseña no coincide con la repetición de la contraseña el sistema alertará la excepción al usuario mediante una ventana emergente informando la no coincidencia de las contraseñas.</p>
<b>Comentarios</b>	Requisitos relacionados: RF-05. El usuario deberá ser de tipo Actor-03

<b>RF-16</b>	<b>Gestión de usuarios. Eliminar usuarios</b>
<b>Actor asociado</b>	Actor-03
<b>Descripción</b>	El usuario podrá eliminar un usuario del sistema.
<b>Precondición</b>	El usuario deberá estar autenticado y estar ubicado en la ventana "Ventana Principal". El usuario deberá ser un Actor-03.
<b>Secuencia normal</b>	<ul style="list-style-type: none"><li>▪ Paso 1: El usuario pulsa desde el menú principal en el botón "8.-Gestión de usuarios".</li><li>▪ Paso 2: En la nueva ventana "Gestión de usuarios" el usuario verá el listado de los usuarios registrados en el sistema.</li><li>▪ Paso 3: El usuario deberá seleccionar el registro del usuario que quiera eliminar.</li><li>▪ Paso 4: El sistema mostrará al usuario una ventana emergente donde podrá comprobar los datos del usuario.</li><li>▪ Paso 5: El usuario pulsará en el botón "Eliminar usuario" para registrar los cambios.</li><li>▪ Paso 6: El sistema mostrará una ventana emergente preguntando al usuario si quiere confirma la acción de eliminar el usuario.</li></ul>





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

	<ul style="list-style-type: none"><li>▪ Paso 7: El usuario realiza la confirmación de la eliminación del usuario.</li><li>▪ Paso 8: El sistema confirma la eliminación del usuario y nos redirigiera al listado de usuarios en la ventana “Gestión de usuarios” para comprobar la eliminación realizada.</li></ul>
<b>Postcondición</b>	Se eliminará un usuario del sistema.
<b>Excepción</b>	Si el usuario no ingresa un campo del formulario, el sistema alertará la excepción al usuario mediante una ventana emergente informando la falta de datos. Si la contraseña no coincide con la repetición de la contraseña, el sistema alertará la excepción al usuario mediante una ventana emergente informando la no coincidencia de las contraseñas.
<b>Comentarios</b>	Requisitos relacionados: RF-05. El usuario deberá ser de tipo Actor-03

A continuación, detallaremos los requisitos funcionales del subproyecto de la aplicación móvil:

<b>RF-17</b>	<b>Iniciar sesión</b>
<b>Actor asociado</b>	Actor-01
<b>Descripción</b>	El usuario podrá acceder al sistema con sus credenciales.
<b>Precondición</b>	El usuario deberá tener credenciales de acceso al sistema. Las credenciales son expedidas por el Actor-03.
<b>Secuencia normal</b>	<ul style="list-style-type: none"><li>▪ Paso 1: El usuario ingresa sus credenciales de acceso.</li><li>▪ Paso 2: El sistema mostrará una ventana emergente dando la bienvenida al usuario.</li><li>▪ Paso 3: El sistema redirigirá al usuario al “Menú principal”.</li></ul>
<b>Postcondición</b>	Se tendrá acceso a todas las funcionalidades contenidas en el menú principal.
<b>Excepción</b>	Si las credenciales son incorrectas el sistema alertará la excepción al usuario mediante una ventana emergente informando el error de acceso. Si un usuario diferente al Actor-01 intenta acceder, el sistema alertará la excepción al usuario mediante una ventana emergente informando la restricción de acceso.





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

<b>Comentarios</b>	El usuario deberá ser de tipo Actor-01
--------------------	--

<b>RF-18</b>	<b>Leer importaciones que llegan</b>
<b>Actor asociado</b>	Actor-01
<b>Descripción</b>	El usuario podrá crear un listado de las importaciones que llegan a las instalaciones industriales.
<b>Precondición</b>	El usuario deberá estar autenticado y estar ubicado en el "Menú principal". El usuario deberá ser un Actor-01.
<b>Secuencia normal</b>	<ul style="list-style-type: none"><li>▪ Paso 1: El usuario pulsará en el elemento del menú llamado "Llegada de importaciones".</li><li>▪ Paso 2: En la siguiente transición "Llegada de importaciones" el usuario pulsará en "Leer código de barras".</li><li>▪ Paso 3: El usuario leerá físicamente el código de barras contenido en las etiquetas de las importaciones.</li><li>▪ Paso 4: El sistema reconocerá la importación y la añadirá al listado en la transición "Llegada de importaciones".</li></ul>
<b>Postcondición</b>	Se creará un listado temporal con la lectura de todas las importaciones que han llegado.
<b>Excepción</b>	Si se duplica la lectura de un mismo código de barras, el sistema alertará la excepción al usuario mediante una ventana emergente informando que la importación ha sido añadida a la lista. Si el código de barras no corresponde a ningún envío, el sistema alertará la excepción al usuario mediante una ventana emergente informando el error de lectura.
<b>Comentarios</b>	Requisitos relacionados: RF-17, RF-19 La lectura de código de barras se realiza mediante la cámara posterior del smartphone.

<b>RF-19</b>	<b>Enviar datos de lectura de importaciones</b>
<b>Actor asociado</b>	Actor-01
<b>Descripción</b>	El usuario podrá actualizar el estado de las importaciones que se han dado entrada.







## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

<b>Precondición</b>	El usuario deberá estar autenticado y estar ubicado en el “Menú principal”. El usuario deberá ser un Actor-01.
<b>Secuencia normal</b>	<ul style="list-style-type: none"><li>▪ Paso 1: El usuario pulsará en el elemento del menú llamado “Actualizar llegadas”.</li><li>▪ Paso 2: En la siguiente transición “Actualizar llegadas” el usuario verá el listado de importaciones leídas por código de barras.</li><li>▪ Paso 3: El usuario pulsará el botón “Subir datos a SIGI”.</li><li>▪ Paso 4: El sistema pedirá confirmación al usuario para subir las lecturas de llegadas al sistema.</li><li>▪ Paso 5: El sistema mostrará un mensaje de confirmación de subida de datos.</li></ul>
<b>Postcondición</b>	Se actualizará el estado de incidencias al envío como “Lectura de llegada X bultos”.
<b>Excepción</b>	Si no existieran envíos leídos, el sistema alertará la excepción al usuario mediante una ventana emergente informando que no es posible actualizar la lista vacía.
<b>Comentarios</b>	Requisitos relacionados: RF-17, RF-18

<b>RF-20</b>	<b>Asignar importaciones a repartidor</b>
<b>Actor asociado</b>	Actor-01
<b>Descripción</b>	El usuario podrá actualizar el estado de las importaciones y añadirlas a su listado de reparto.
<b>Precondición</b>	El usuario deberá estar autenticado y estar ubicado en el “Menú principal”. El usuario deberá ser un Actor-01.
<b>Secuencia normal</b>	<ul style="list-style-type: none"><li>▪ Paso 1: El usuario pulsará en el elemento del menú llamado “Asignar reparto”.</li><li>▪ Paso 2: En la siguiente transición “Asignar reparto” el usuario pulsará el botón “Añadir envío”.</li><li>▪ Paso 3: El usuario leerá físicamente el código de barras contenido en las etiquetas de las importaciones/envíos.</li></ul>





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

	<ul style="list-style-type: none"><li>▪ Paso 4: El sistema añadirá la importación/envío a la lista de asignación de reparto.</li><li>▪ Paso 5: Cuando el usuario haya terminado de añadir los envíos que llevará a reparto pulsará el botón “Terminar asignación”.</li><li>▪ Paso 6: El sistema pedirá confirmación al usuario.</li><li>▪ Paso 7: El usuario confirmará la asignación de reparto.</li><li>▪ Paso 8: El sistema confirmará la asignación de repartos.</li></ul>
<b>Postcondición</b>	Se actualizará el estado de incidencias al envío como “En reparto”.
<b>Excepción</b>	Si no existieran envíos asignados, el sistema alertará la excepción al usuario mediante una ventana emergente informando que no es posible terminar la asignación dado que no existen envíos.
<b>Comentarios</b>	Requisitos relacionados: RF-17

<b>RF-21</b>	<b>Ver listado de reparto</b>
<b>Actor asociado</b>	Actor-01
<b>Descripción</b>	El usuario podrá visualizar los envíos que lleva a reparto.
<b>Precondición</b>	El usuario deberá estar autenticado y estar ubicado en el “Menú principal”. El usuario deberá ser un Actor-01.
<b>Secuencia normal</b>	<ul style="list-style-type: none"><li>▪ Paso 1: El usuario pulsará en el elemento del menú llamado “Listado de reparto”.</li><li>▪ Paso 2: En la siguiente transición “Listado de reparto” el usuario visualizará el listado de envíos que lleva a reparto.</li></ul>
<b>Postcondición</b>	
<b>Excepción</b>	Si no existieran envíos asignados, el listado de reparto aparecerá vacío.
<b>Comentarios</b>	Requisitos relacionados: RF-17
<b>RF-22</b>	<b>Entregar importaciones</b>
<b>Actor asociado</b>	Actor-01





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

<b>Descripción</b>	El usuario podrá confirmar la entrega de los envíos.
<b>Precondición</b>	El usuario deberá estar autenticado y estar ubicado en el “Menú principal”. El usuario deberá ser un Actor-01.
<b>Secuencia normal</b>	<ul style="list-style-type: none"><li>▪ Paso 1: El usuario pulsará en el elemento del menú llamado “Listado de reparto”.</li><li>▪ Paso 2: En la siguiente transición “Listado de reparto” el usuario visualizará el listado de envíos que lleva a reparto.</li><li>▪ Paso 3: El usuario seleccionará el envío que va a entregar al destinatario.</li><li>▪ Paso 4: En la nueva transición “Entregar envío”, el destinatario del envío deberá firmar con su dedo sobre la pantalla del smartphone y rellenará su DNI.</li><li>▪ Paso 5: El usuario pulsará el botón “Entregado”.</li><li>▪ Paso 6: El sistema pedirá la confirmación al usuario de la entrega mediante una ventana emergente.</li><li>▪ Paso 7: El usuario confirmará la entrega</li></ul>
<b>Postcondición</b>	El estado del envío se actualizará a: Entregado y se eliminará del listado de reparto.
<b>Excepción</b>	Si no existieran envíos asignados, el listado de reparto aparecerá vacío y no se podrá realizar ninguna entrega.
<b>Comentarios</b>	Requisitos relacionados: RF-17, RF-20, RF-21

<b>RF-23</b>	<b>Crear incidencias en reparto</b>
<b>Actor asociado</b>	Actor-01
<b>Descripción</b>	El usuario podrá añadir una incidencia a la importación/envío justificando la no posibilidad de entrega.
<b>Precondición</b>	El usuario deberá estar autenticado y estar ubicado en el “Menú principal”. El usuario deberá ser un Actor-01.
<b>Secuencia normal</b>	<ul style="list-style-type: none"><li>▪ Paso 1: El usuario pulsará en el elemento del menú llamado “Listado de reparto”.</li><li>▪ Paso 2: En la siguiente transición “Listado de reparto” el usuario visualizará el listado de envíos que lleva a reparto.</li></ul>





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

	<ul style="list-style-type: none"><li>▪ Paso 3: El usuario seleccionará el envío al que va añadir la incidencia.</li><li>▪ Paso 4: En la nueva transición “Entregar envío”, en el campo Incidencia seleccionará el tipo de incidencia y en el campo Comentario datos relevantes a la incidencia.</li><li>▪ Paso 5: El usuario pulsará sobre el botón “No entregado”.</li><li>▪ Paso 6: El sistema pedirá la confirmación al usuario de la incidencia mediante una ventana emergente.</li><li>▪ Paso 7: El usuario confirmará la creación de la incidencia.</li></ul>
<b>Postcondición</b>	El estado del envío se actualizará a una nueva incidencia y por consiguiente se eliminará del listado de reparto.
<b>Excepción</b>	Si no existieran envíos asignados, el listado de reparto aparecerá vacío y no se podrá realizar ninguna entrega.
<b>Comentarios</b>	Requisitos relacionados: RF-17, RF-20, RF-21, RF-22

### 5.1.4. Requisitos no funcionales

A continuación, detallaremos los requisitos no funcionales del sistema. Estos requisitos comprenden aumentar la calidad y dar un valor añadido al proyecto software.

<b>RNF-01</b>	<b>Usabilidad</b>
<b>Descripción</b>	El sistema deberá ser fácil de usar. Se debe intentar que el tiempo de aprendizaje de uso de la aplicación sea el mejor. El sistema deberá dejar usar atajos para los menús principales de la aplicación, atajos como elementos numerados.
<b>Comentarios</b>	Este requisito es aplicable a los dos subproyectos software: aplicación de escritorio y aplicación móvil.

<b>RNF-02</b>	<b>Disponibilidad</b>
<b>Descripción</b>	El sistema deberá estar siempre activo, evitando mantenimientos largos o caídas del servidor.
<b>Comentarios</b>	Este requisito es aplicable a los dos subproyectos software: aplicación de escritorio y aplicación móvil.





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

<b>RNF-03</b>	<b>Estabilidad</b>
<b>Descripción</b>	El sistema deberá ser robusta y no propensa a fallos. El sistema deberá mantener una línea de trabajo estable.
<b>Comentarios</b>	Este requisito es aplicable a los dos subproyectos software: aplicación de escritorio y aplicación móvil.

<b>RNF-04</b>	<b>Rendimiento</b>
<b>Descripción</b>	El sistema deberá ofrecer todas sus funcionalidades con el menor consumo de los recursos disponibles para ello, el sistema deberá ser eficiente en su desarrollo.
<b>Comentarios</b>	Este requisito es aplicable a los dos subproyectos software: aplicación de escritorio y aplicación móvil.

### 5.1.5. Requisitos de información

A continuación, mostraremos de manera detallada los requisitos de información del sistema.

<b>RI-01</b>	<b>Usuarios</b>
<b>Descripción</b>	El sistema almacenará los datos o información de las personas que accederá a la aplicación.
<b>Requisitos asociados</b>	RF-05 RF-06 RF-08 RF-11 RF-12 RF-13 RF-14 RF-17
<b>Datos</b>	Nombre Contraseña DNI Domicilio ID repartidor Matricula





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

	Teléfono
<b>Comentarios</b>	Este requisito es aplicable a los dos subproyectos software: aplicación de escritorio y aplicación móvil.

<b>RI-02</b>	<b>Envío</b>
<b>Descripción</b>	El sistema almacenará los datos o información de las importaciones que gestionará el sistema
<b>Requisitos asociados</b>	RF-01 RF-02 RF-03 RF-04 RF-06 RF-08 RF-09 RF-11 RF-12 RF-18 RF-19 RF-20 RF-21 RF-22 RF-23
<b>Datos</b>	Dirección destinatario Dirección remitente Nombre destinatario Nombre remitente Número de bultos Número de envío Peso Reembolso Teléfono destinatario Teléfono remitente Entrega: Fecha de entrega, firma, hora de entrega, DNI consignatario, repartidor que entrega la mercancía.





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

	Incidentes: Tipo de incidencia, descripción de la incidencia, fecha y hora.
<b>Comentarios</b>	Este requisito es aplicable a los dos subproyectos software: aplicación de escritorio y aplicación móvil.

### 5.2. Modelado y diseño

En el anterior punto (5.1 Análisis y requisitos), hemos analizado el sistema para crear todos los tipos de requerimientos necesarios para realizar la construcción del proyecto. A partir de estos requerimientos crearemos el modelado y diseño del sistema.





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

Primero mostraremos un diagrama de clases creado con la herramienta Visual Studio (véase punto 4.2) en el cual veremos cómo se va a especificar el modelo de datos en nuestras aplicaciones.

Posteriormente, mostraremos el diagrama del modelo lógico de la base de datos en donde se mostrará la estructura y relaciones necesarias para que nuestro sistema se pueda llevar a cabo.

Finalmente, mostraremos como hemos creado los prototipos de las interfaces graficas de las dos aplicaciones. Este prototipado se ha realizado con la aplicación Balsamiq Mockups 3 (véase punto 4.1).

### 5.2.1. Diagrama de clases del sistema

A continuación, mostraremos el diagrama de clases del sistema creado:

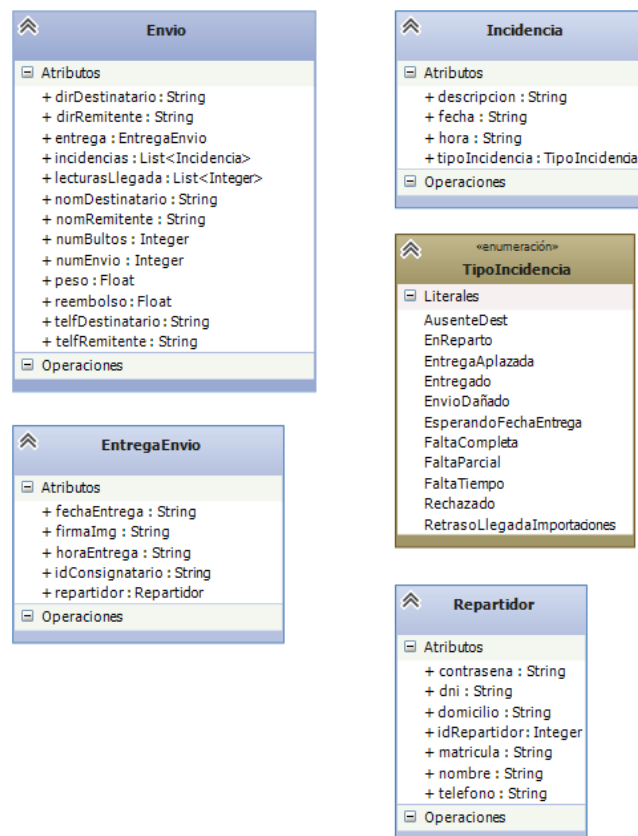


Figura 62 - Diagrama de clases del sistema SIGI

Podemos comprobar que todo el sistema está unido en cuanto la información se refiere, es decir, la aplicación de escritorio usará y consumirá la misma información que la aplicación móvil. No habrá información redundante.





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

Estas clases formarán parte del modelo lógico de negocio de la aplicación y por tanto existirán clases e instancias a estas clases como objetos dentro del paradigma de la Programación Orientada a Objetos, POO.

En el capítulo de Análisis y requisitos, véase para ser más precisos punto 5.1.5 Requisitos de información, especificado los datos claves de la entidad Envío y Usuario.

Hay que aclarar que la clase “Repartidor” corresponde a todos los usuarios registrados en el sistema y que para identificar si un usuario es Administrativo, Encargado o Repartido, en el campo “matricula” deberá estar como ADMINISTRATIVO, ENCARGADO o XXXX-AAA (formato matricula europea).

Por otro lado, la clase “TipoIncidencia” es una clase de tipo Enumeración la cual contendrá los tipos de incidencias que habrá por defecto en el sistema. Se ha diseñado de esta manera ya que los datos relativos a los tipos de incidencias no dinámicos y deben ser robustos a la hora de ser usados.

Las clases “EntregaEnvio” e “Incidencias” son dos clases parecidas en cuanto a su finalidad. Por un lado, la clase “EntregaEnvio” se usará para registrar la información relativa a la entrega de un envío con sus datos relevantes. Por otro lado, la clase “Incidencias” registrará información relativa a eventos posteriores a la creación de un “Envío”, hay que remarcar que un Envío puede contener muchos eventos.

Para la realización de los diagramas UML dentro de Visual Studio (véase capítulo 3.1.9 y 4.2) hemos creado una solución vacía llamada “SIGI”. Desde aquí hemos creado un proyecto de tipo “Proyecto de modelado” en donde hemos creado el diagrama de clases y el diagrama de casos de uso ya expuesto en el capítulo 5.1.3 Requisitos funcionales.

### 5.2.2. Diagrama lógico de la base de datos

Dado que ya sabemos el modelo de datos que contendrá nuestra aplicación a continuación mostraremos la estructura de las entidades y relaciones para la persistencia de los datos en nuestra base de datos.





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

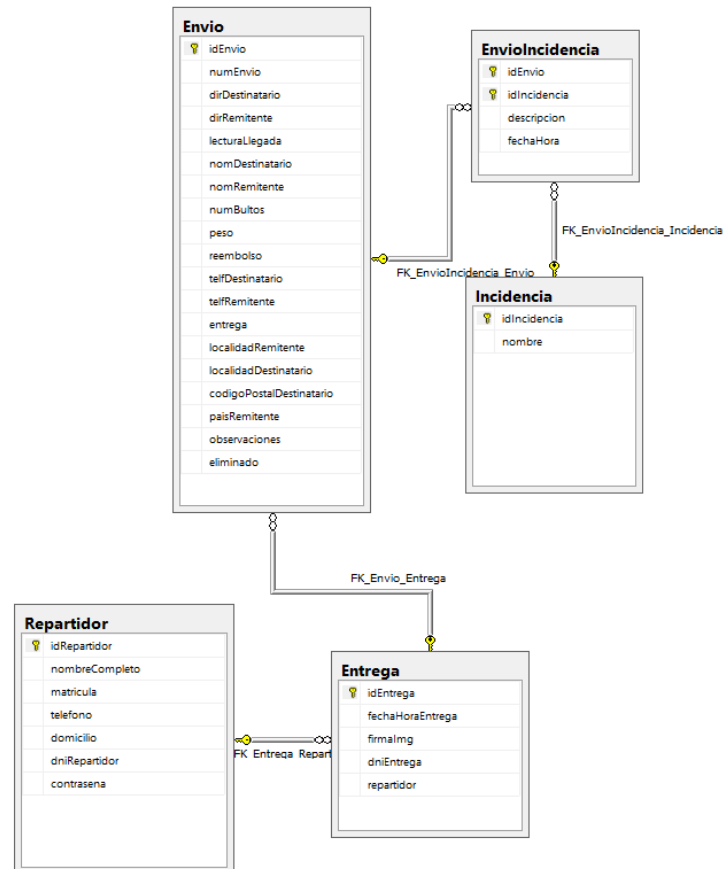


Figura 63 - Diagrama lógico de la base de datos en el sistema SIGI

Las entidades son Entrega, Repartidor, Incidencia, Envío y una entidad relacional llamada EnvioIncidencia. Por otro lado, en las relaciones, podemos comentar que las entidades Repartidor, Entrega y Envío están relacionadas 1..1 dado que un repartidor realizará una entrega de un envío.

Por otro lado, vemos que la entidad relacional EnvioIncidencia está relacionada 1..1 por la entidad Envío e Incidencia hecho que crea esta entidad temporal EnvioIncidencia: lo que quiere decir que un envío puede tener muchas incidencias y una incidencia puede estar en más de un envío.

El diagrama de lógico de la base de datos ha sido creado por la herramienta SQL Server 2014 Management Studio (véase capítulo 4.5).

### 5.2.3. Prototipado de las interfaces gráficas

En este punto del capítulo mostraremos varios ejemplos de los bocetos realizados para generar las interfaces gráficas, tanto de la aplicación de escritorio como de la aplicación móvil.



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

Los dos prototipos se han creado gracias a la herramienta Balsamiq Mockups 3 descrita en el capítulo 4.1 de este documento.

Quiero mencionar que el prototipado se creó en una etapa temprana del proyecto por lo que se ha ido generados cambios en el prototipo durante el desarrollo del diseño y, por consiguiente, puede que existan pequeños cambios que difieran entre la interfaz gráfica final y el prototipo.

Para la aplicación de escritorio se ha generado un total de 13 ventanas, relacionadas entre sí para generar un comportamiento de enlace funcionalidad.

A continuación, mostraremos la ventana principal de la aplicación:

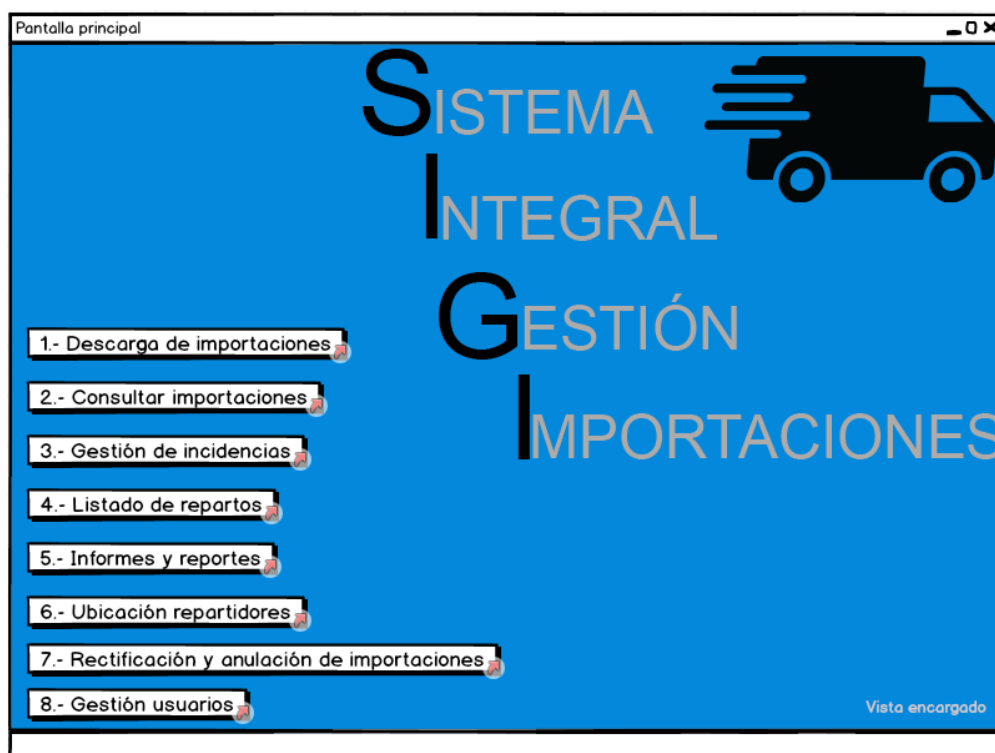


Figura 64 - Prototipo ventana principal de la aplicación de escritorio

Podemos ver que el estilo visual de colores predomina el color azul con código hexadecimal en el estándar RGB #0488DB. Vemos un menú principal numerado y un logo generado que será símbolo de nuestra aplicación. Todas las ventanas tienen un tamaño de 800x600 píxeles, medida elegida para que no ocupe todo el espacio de trabajo de la pantalla de trabajo del ordenador dado que no es necesario.

En la Figura 65, mostraremos una vista resumen de todas las ventanas de la aplicación:



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

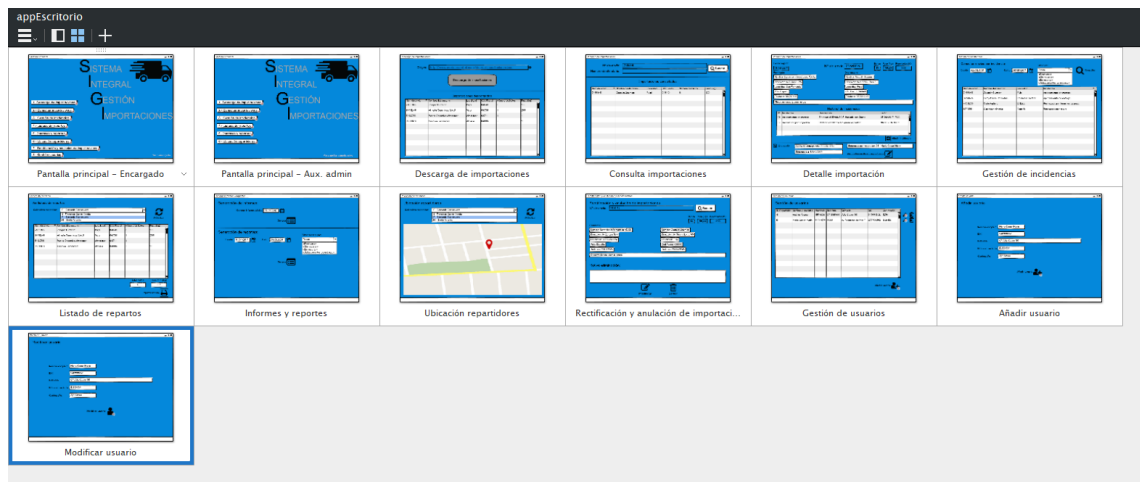


Figura 65 - Vista resumida de todas las ventanas prototipadas de la aplicación de escritorio

La aplicación móvil ha seguido el mismo proceso, se han creado las diferentes ventanas o transición posibles, al igual que la aplicación de escritorio, mostraremos el menú/ventana principal y una vista resumen de todas las ventanas posibles del prototipo:

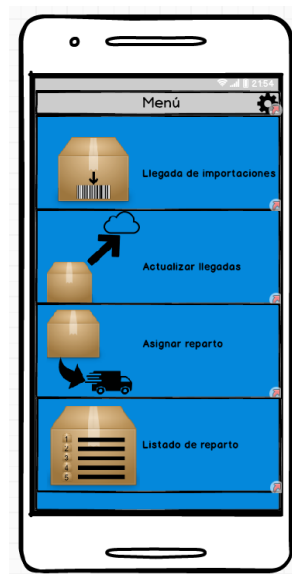


Figura 66 - Prototipo ventana/menú principal de la aplicación móvil

En la Figura 66, vemos como seguimos manteniendo el tema visual de la interfaz gráfica, con un fondo azul y letras negras, iconos grandes para facilitar la usabilidad (requisito RNF-01).

La Figura 67 muestra una vista resumida de todas las transiciones prototipadas en la aplicación móvil:



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

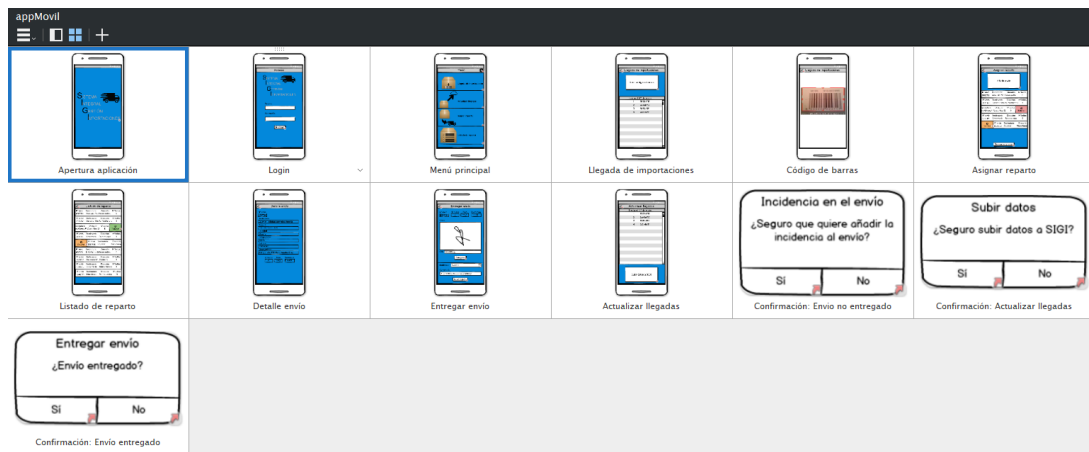


Figura 67 - Vista resumen de todas las transiciones de la aplicación móvil

Vemos que también existen prototipos de ventana o alertas emergentes para confirmar acciones de la aplicación.

### 5.3. Implementación y desarrollo aplicación de escritorio

En este punto del capítulo abordaremos las diferentes fases que hemos realizado o dividido para desarrollar la aplicación de escritorio. He de mencionar que la implementación se ha realizado íntegramente con la herramienta de desarrollo Visual Studio (véase capítulo 4.2).

#### 5.3.1. Creación y estructura del proyecto

Para empezar, vamos a describir como hemos creado el proyecto, configurado y estructurado para la puesta en marcha del desarrollo o implementación del mismo.

Dentro de Visual Studio hemos creado una solución vacía en donde hemos ido añadiendo los diferentes proyectos. Para el desarrollo de la aplicación de escritorio hemos creado tres tipos de proyectos:

- 🚚 Proyecto/aplicación de servicios web Restful con Windows Communication Foundation
- 🚚 Proyecto para pruebas/test para la aplicación de servicios web Resful
- 🚚 Proyecto/aplicación cliente con Windows Presentation Foundation





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

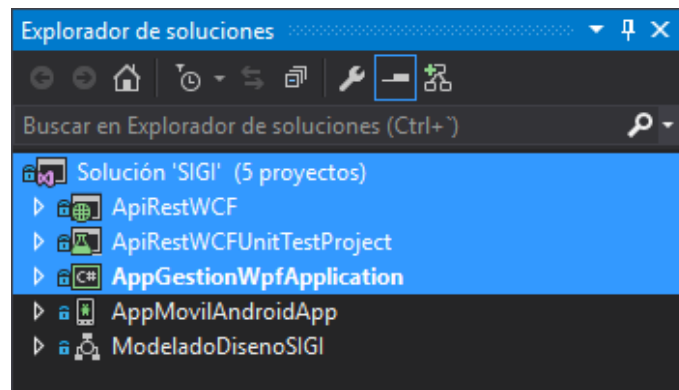


Figura 68 - Explorador de soluciones en Visual Studio

En la Figura 68 vemos resaltado los 3 proyectos que hemos creado para el desarrollo de la aplicación.

### 5.3.2. Configuración repositorio con GIT

Antes de comenzar a implementar el sistema, hemos unido nuestro repositorio creado y comentado en el capítulo 4.4 de Visual Studio Team Services. Para ello hemos copiado el enlace generado por el repositorio en Team Services:

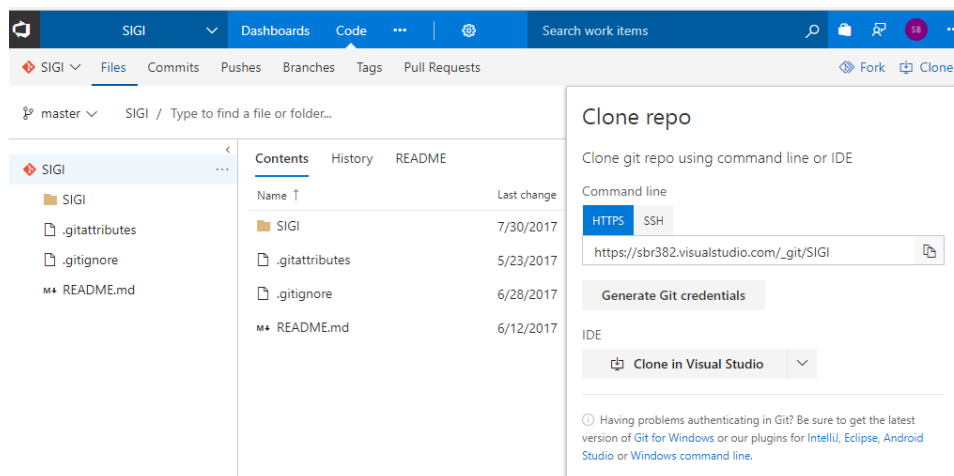


Figura 69 - Enlace/link repositorio GIT en Visual Studio Team Services

Team Services nos da la posibilidad si lo preferimos de enlazar nuestro proyecto con solo pulsar el botón “Clone in Visual Studio”.

Una vez enlazado el repositorio en Visual Studio, este se autoconfigurará en nuestro proyecto y tendremos en nuestra disposición la herramienta “Team explorer”, la cual nos ayudará a la hora de proteger nuestro proyecto, subir y guardar los cambios en el repositorio, crear ramificaciones y otras configuraciones que queramos añadir a nuestro repositorio:





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

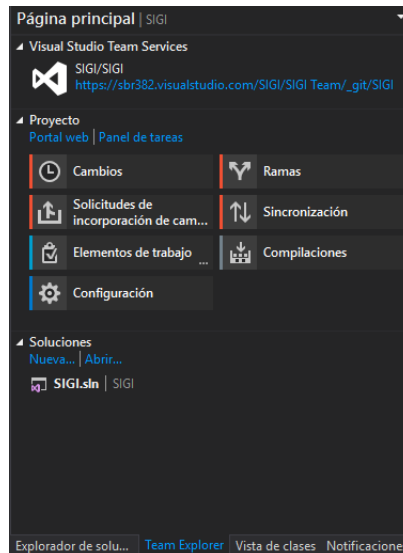


Figura 70 - Vista herramienta Team Explorer dentro de Visual Studio

### 5.3.3. Aplicación de servicios web Restful

En nuestra solución hemos creado un proyecto llamado ApiRestWCF basado en las plantillas llamadas Aplicación de Servicio Web de Visual Studio. Este proyecto creará el servicio web Rest. Durante el desarrollo del punto de este capítulo nos centraremos en este proyecto.

Vamos a utilizar la tecnología de Microsoft llamada Windows Communication Foundation, en el capítulo 3.1.8 hemos descrito en que se basa y como funciona.

El servicio web Restful se alimentará de la base de datos creada en el servicio en la nube de Azure (véase capítulo 4.3: Microsoft Azure) y creará los controladores que utilizarán las aplicaciones cliente para administrar los datos.

La idea es crear un servicio web que las aplicaciones clientes consuman o consulten los datos de la base de datos.

En el capítulo 5.2.2 Diagrama lógica de la base de datos vimos que existen las siguientes entidades: Entrega, Repartidor, Incidencia, Envío y una entidad relacional llamada EnvíoIncidencia. Para cada entidad hemos creado un servicio de Restful.

A continuación, detallaremos las etapas realizadas para crear el servicio web Rest:

**🚚 Conexión con los datos.** Lo primero que debemos es crear un diagrama o clase de datos de SQL a LINQ. Desde el Explorador de servidores añadiremos la cadena de conexión de nuestra base de datos en Azure. Introduciremos el nombre del servidor y



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

nuestras credenciales. Una vez reconocidas nuestras credenciales podemos seleccionar la base de datos:

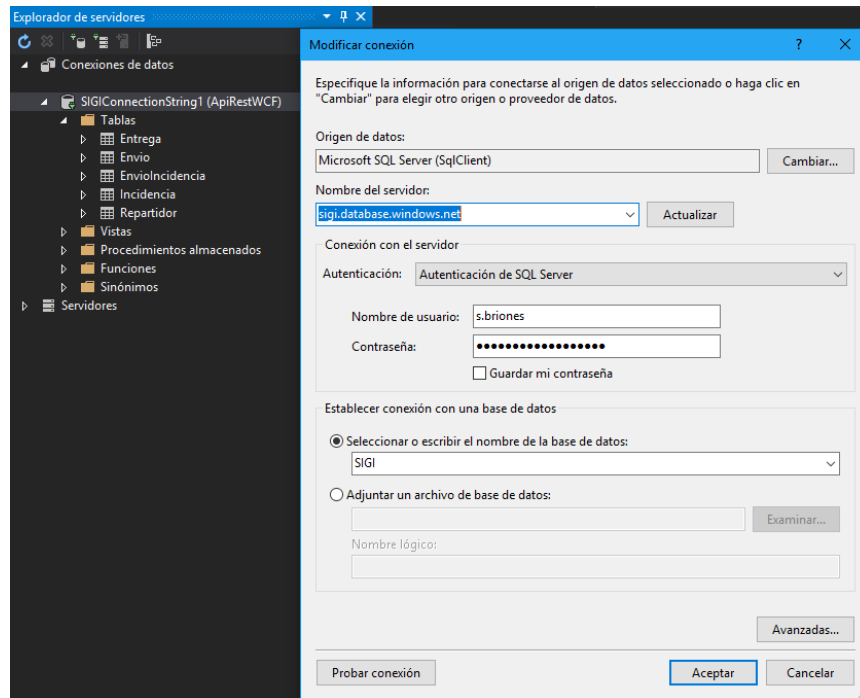


Figura 71 - Configuración cadena de conexión entre Azure y Visual Studio

Seguidamente crearemos un elemento en nuestro proyecto, pulsamos con el botón derecho y añadimos elemento: Clases de Linq to SQL. Esto nos creará un diagrama vacío para añadir las tablas de la base de datos de Azure, por tanto, arrastraremos esas tablas al diagrama:

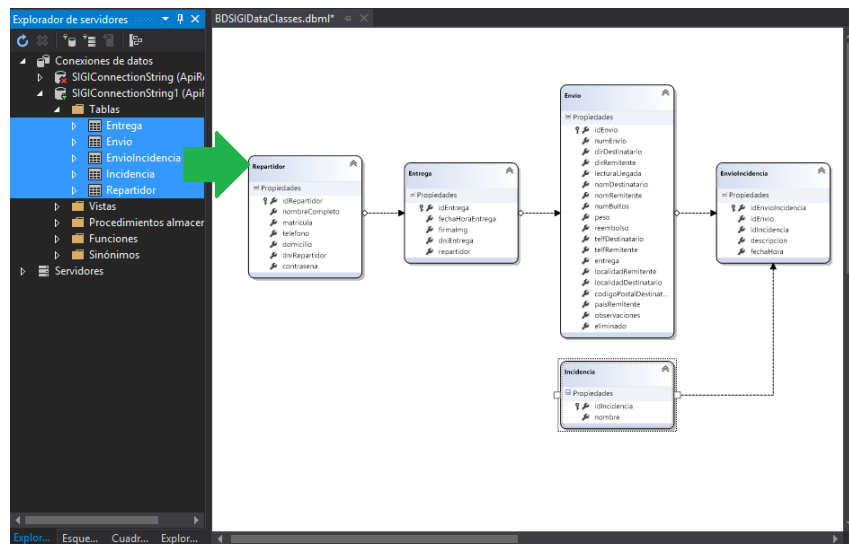


Figura 72 - Creación de modelo de clases en Visual Studio





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

Si guardamos el diagrama, automáticamente se autogenerará “código behind”, este código autogenerado serán los modelos o clases que utilizaremos para codificar nuestro programa. Si observamos el Explorador de soluciones, se nos creará una clase por entidad añadida:

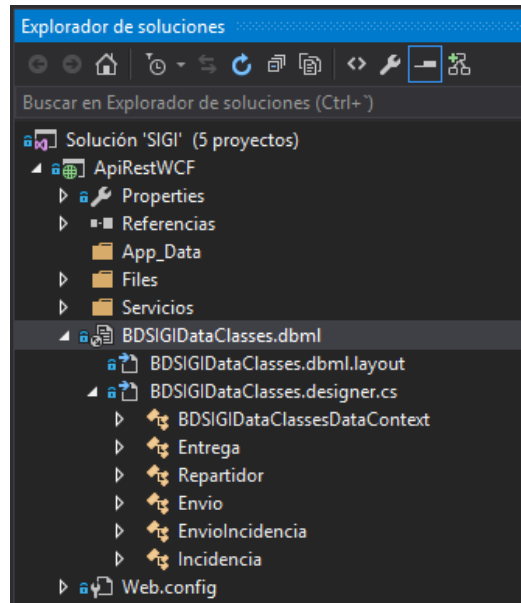


Figura 73 - Explorador de soluciones con código autogenerado

**Creación del servicio.** Acto seguido de añadir las clases y datos de la base de datos que vamos a manipular, crearemos los servicios. He creado para tener organizado el proyecto, un servicio por entidad.

Un servicio Rest lleva consigo un contrato o Interfaz de clases a utilizar por obligación, si esta interfaz de clases contiene un método de consulta, el servicio lo debe implementar. Sin más dilación veremos cómo hemos creado el contrato de servicio para realizar la consulta de todos los envíos contenidos en el sistema:

```
[ServiceContract]
public interface IEnvioService
{
    [OperationContract]
    [WebGet(UriTemplate = "/getEnvios",
    ResponseFormat = WebMessageFormat.Json,
    RequestFormat = WebMessageFormat.Json,
    BodyStyle = WebMessageBodyStyle.Wrapped)
    ]
    List<Envio> getEnvios();
}
```



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

Vemos que la interfaz IEnvioService declara un método llamado getEnvios() que la clase EnvioService deberá implementar de manera que se transmita a partir de la URL: /getEnvios y que envíe y devuelva listado de Envios en formato Json.

En el método getEnvios() de la clase EnvioService implementaremos la función de consulta de la entidad de relación de Linq:

```
public class EnvioService: IEnvioService
{
    BDSIGIDataClassesDataContext conexBD = new BDSIGIDataClassesDataContext();

    public List<Envio> getEnvios()
    {
        List<Envio> resultado = null;
        resultado = conexBD.Envio.ToList();

        if (resultado != null)
            return resultado;
        else
            throw new Exception();
    }
}
```

La clase BDSIGIDataClassesDataConext es la clase creada a partir del diagrama que hemos generado de la base de datos de Azure. Esta clase contiene todas las clases del modelo y la gestión para realizar las consultas a la base de datos.

Hemos creado un servicio Rest que llamando desde la url <http://localhost/getEnvios> nos mostrará una consulta de todos los envíos registrados en el sistema.

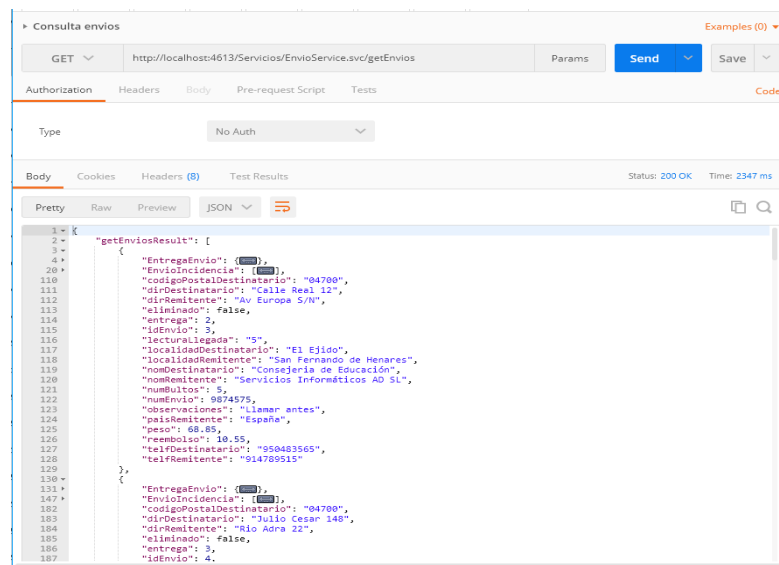


Figura 74 - Consulta a la base de datos mediante Postman



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

Gracias a la herramienta Postman, véase capítulo 4.6 Postman, realizaremos una consulta de todos los envíos. El resultado de la consulta es en formato Json tal y como se acordó en el contrato del servicio creado EnvioService.svc:

```
{
  "getEnviosResult": [
    {
      "EntregaEnvio": {},
      "EnvioIncidencia": [],
      "codigoPostalDestinatario": "04700",
      "dirDestinatario": "Calle Real 12",
      "dirRemitente": "Av Europa S/N",
      "eliminado": false,
      "entrega": 2,
      "idEnvio": 3,
      "lecturalLlegada": "5",
      "localidadDestinatario": "El Ejido",
      "localidadRemitente": "San Fernando de Henares",
      "nomDestinatario": "Consejeria de Educación",
      "nomRemitente": "Servicios Informáticos AD SL",
      "numBultos": 5,
      "numEnvio": 9874575,
      "observaciones": "Llamar antes",
      "paisRemitente": "España",
      "peso": 68.85,
      "reembolso": 10.55,
      "telfDestinatario": "950483565",
      "telfRemitente": "914789515"
    },
    {
      "EntregaEnvio": {},
      "EnvioIncidencia": [],
      "codigoPostalDestinatario": "04700",
      "dirDestinatario": "Julio Cesar 148",
      "dirRemitente": "Rio Adra 22",
      "eliminado": false,
      "entrega": 3,
      "idEnvio": 4,
      "lecturalLlegada": "1",
      "localidadDestinatario": "El Ejido",
      "localidadRemitente": "Viator",
      "nomDestinatario": "Carla Agüero",
      "nomRemitente": "TNT Almería",
      "numBultos": 1,
      "numEnvio": 1236547,
      "observaciones": "Entregar por la mañana",
      "paisRemitente": "España",
      "peso": 15,
      "reembolso": 0,
      "telfDestinatario": "617438285",
      "telfRemitente": "950173838"
    }
  ]
}
```

La creación de este servicio Rest para la consulta de envíos es un ejemplo de todas las peticiones HTTP posibles que ha sido aplicadas a las demás entidades de la base de datos.

Una vez creadas y realizadas las pruebas pertinentes hemos desplegado el servicio web a Azure. Esto quiere decir que dé lugar de ejecutar el servicio web en localhost





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

(en nuestro equipo), ya directamente es ejecutado en el servidor de Azure, la url de este servidor es: <http://despliquesigi.azurewebsites.net>

Las peticiones HTTP creadas son las siguiente enumeradas:

### Envío:

- Ver todos los envíos:
  - Tipo de petición/verbo HTTP: **GET**
  - URL:  
<http://despliquesigi.azurewebsites.net/Servicios/EnvioService.svc/getEnvios>
- Ver el detalle de un envío:
  - Tipo de petición/verbo HTTP: **GET**
  - URL:  
<http://despliquesigi.azurewebsites.net/Servicios/EnvioService.svc/getEnvios/{id-envio}>
  - Parámetros de la url:
    - {id-envio}: identificador del envío en el sistema
- Añadir un envío al sistema:
  - Tipo de petición/verbo HTTP: **POST**
  - URL:  
<http://despliquesigi.azurewebsites.net/Servicios/EnvioService.svc/addEnvio>
  - Cuerpo de la petición HTTP (datos ejemplo):

```
{
  "newEnvio":{
    "codigoPostalDestinatario": "04770",
    "dirDestinatario": "Aurea 2 12",
    "dirRemitente": "Paseo de la Castellana S/N",
    "localidadDestinatario": "Adra",
    "localidadRemitente": "Madrid",
    "nomDestinatario": "Rogelio Mayor",
    "nomRemitente": "Drop Shot",
    "numBultos": 1,
    "numEnvio": 1234567,
    "paisRemitente": "España",
    "peso": 1,
    "reembolso": 0.0,
    "telfDestinatario": "617438285",
    "telfRemitente": "917852456",
    "observaciones": "Llamar y entregar antes de las 13hrs.",
    "eliminado": 0
  }
}
```

- Modificar un envío en el sistema:
  - Tipo de petición/verbo HTTP: **PUT**



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

- URL:  
<http://despliguesigi.azurewebsites.net/Servicios/EnvioService.svc/setEnvio/{id-envio}>
- Parámetros de la url:
  - {id-envio}: identificador del envío en el sistema
- Cuerpo de la petición HTTP (datos ejemplo):

```
{
  "setEnvio": {
    "EntregaEnvio": null,
    "codigoPostalDestinatario": "04640",
    "dirDestinatario": "Ctra Huercal-overa a Pulpi S/N",
    "dirRemitente": "Asertash street 2",
    "eliminado": true,
    "idEnvio": 60,
    "lecturaLlegada": null,
    "localidadDestinatario": "Pulpi",
    "localidadRemitente": "Beijing",
    "nomDestinatario": "Mkito Catal importaciones SL",
    "nomRemitente": "samples for you",
    "numBultos": 1,
    "numEnvio": 8457745,
    "observaciones": "Llamar antes",
    "paisRemitente": "China",
    "peso": 17,
    "reembolso": 50,
    "telfDestinatario": "+34 617438285",
    "telfRemitente": "917852456"
  }
}
```

- Eliminar un envío en el sistema:
  - Tipo de petición/verbo HTTP: **DELETE**
  - URL:  
<http://despliguesigi.azurewebsites.net/Servicios/EnvioService.svc/cleanEnvio/{id-envio}>
  - Parámetros de la url:
    - {id-envio}: identificador del envío en el sistema

### Incidencia:

- Ver todas las incidencias:
  - Tipo de petición/verbo HTTP: **GET**



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

- URL:  
<http://despliquesigi.azurewebsites.net/Servicios/IncidenciaService.svc/getIncidencias>
- Ver el detalle de una incidencia:
  - Tipo de petición/verbo HTTP: **GET**
  - URL:  
<http://despliquesigi.azurewebsites.net/Servicios/IncidenciaService.svc/getIncidencias/{id-incidencia}>
  - Parámetros de la url:
    - {id-incidencia}: identificador de la incidencia en el sistema
- Añadir una incidencia al sistema:
  - Tipo de petición/verbo HTTP: **POST**
  - URL:  
<http://despliquesigi.azurewebsites.net/Servicios/IncidenciaService.svc/addIncidencia>
  - Cuerpo de la petición HTTP (datos ejemplo):

```
{
  "newIncidencia":
    {
      "nombre": "En reparto"
    }
}
```

- Modificar una incidencia en el sistema:
  - Tipo de petición/verbo HTTP: **PUT**
  - URL:  
<http://despliquesigi.azurewebsites.net/Servicios/IncidenciaService.svc/setIncidencia/{id-incidencia}>
  - Parámetros de la url:
    - {id-incidencia}: identificador de la incidencia en el sistema
  - Cuerpo de la petición HTTP (datos ejemplo):

```
{
  "setIncidencia":
    {
      "nombre": "nueva incidencia X"
    }
}
```



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

- Eliminar una incidencia en el sistema:
  - Tipo de petición/verbo HTTP: **DELETE**
  - URL:  
<http://despliquesigi.azurewebsites.net/Servicios/IncidenciaService.svc/cleanIncidencia/{id-incidencia}>
  - Parámetros de la url:
    - {id-incidencia}: identificador de la incidencia en el sistema

### Repartidor:

- Ver todos los repartidores/usuarios:
  - Tipo de petición/verbo HTTP: **GET**
  - URL:  
<http://despliquesigi.azurewebsites.net/Servicios/RepartidorService.svc/getRepartidores>
- Ver el detalle de un usuario:
  - Tipo de petición/verbo HTTP: **GET**
  - URL:  
<http://despliquesigi.azurewebsites.net/Servicios/RepartidorService.svc/getRepartidores/{id-repartidor}>
  - Parámetros de la url:
    - {id-repartidor}: identificador del usuario o repartidor en el sistema
- Añadir un usuario al sistema:
  - Tipo de petición/verbo HTTP: **POST**
  - URL:  
<http://despliquesigi.azurewebsites.net/Servicios/RepartidorService.svc/addRepartidor>
  - Cuerpo de la petición HTTP (datos ejemplo):

```
{
  "newRepartidor":
  {
    "contrasena": "987654",
    "dniRepartidor": "987654",
    "domicilio": "C/ Clavel 56, 1ªA",
    "matricula": "8974HXS",
    "nombreCompleto": "Jesús Artés",
    "telefono": "652145789"
  }
}
```





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

- Modificar un usuario en el sistema:
  - Tipo de petición/verbo HTTP: **PUT**
  - URL:  
<http://despliquesigi.azurewebsites.net/Servicios/RepartidorService.svc/setRepartidor/{id-repartidor}>
  - Parámetros de la url:
    - {id-repartidor}: identificador del usuario en el sistema
  - Cuerpo de la petición HTTP (datos ejemplo):

```
{
  "setRepartidor":
  {
    "contrasena": "1234",
    "dniRepartidor": "54099165Z",
    "domicilio": "Julio Cesar 36 El Ejido",
    "idRepartidor": 1,
    "matricula": "1181GDY",
    "nombreCompleto": "Andrés Aleman",
    "telefono": "671884554"
  }
}
```

- Eliminar un usuario en el sistema:
  - Tipo de petición/verbo HTTP: **DELETE**
  - URL:  
<http://despliquesigi.azurewebsites.net/Servicios/RepartidorService.svc/cleanRepartidor/{id-repartidor}>
  - Parámetros de la url:
    - {id-repartidor}: identificador del usuario en el sistema

### Entregas:

- Ver todas las entregas:
  - Tipo de petición/verbo HTTP: **GET**
  - URL:  
<http://despliquesigi.azurewebsites.net/Servicios/EntregaService.svc/getEntregas>
- Ver el detalle de una entrega:
  - Tipo de petición/verbo HTTP: **GET**
  - URL:  
<http://despliquesigi.azurewebsites.net/Servicios/EntregaService.svc/getEntregas/{id-entrega}>
  - Parámetros de la url:
    - {id-entrega}: identificador de la entrega en el sistema





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

- Añadir un usuario al sistema:
  - Tipo de petición/verbo HTTP: **POST**
  - URL:  
<http://despliguesigi.azurewebsites.net/Servicios/EntregaService.svc/addEntrega>
  - Cuerpo de la petición HTTP (datos ejemplo):

```
{
  "newEntrega":
  {
    "firmalmg": "SADASDADASD.png",
    "dniEntrega": "12345678S",
    "repartidor": 1
  }
}
```

- Modificar una entrega en el sistema:
  - Tipo de petición/verbo HTTP: **PUT**
  - URL:  
<http://despliguesigi.azurewebsites.net/Servicios/EntregaService.svc/setEntrega/{id-entrega}>
  - Parámetros de la url:
    - {id-entrega}: identificador de la entrega en el sistema
  - Cuerpo de la petición HTTP (datos ejemplo):

```
{
  "setEntrega":
  {
    "fechaEntrega": "31/05/2017",
    "firmalmg": "firma_541696969.png",
    "horaEntrega": "21:00",
    "dniEntrega": "B0489214X",
    "repartidor": 4
  }
}
```

- Eliminar una entrega en el sistema:
  - Tipo de petición/verbo HTTP: **DELETE**
  - URL:  
<http://despliguesigi.azurewebsites.net/Servicios/EntregaService.svc/cleanEntrega/{id-entrega}>
  - Parámetros de la url:
    - {id-entrega}: identificador de la entrega en el sistema

### Archivos (fotos de las firmas):

- Ver una firma de una entrega:
  - Tipo de petición/verbo HTTP: **GET**



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

- URL:  
<http://despliquesigi.azurewebsites.net/Servicios/FileUploadService.svc/Files/{nombre-archivo}/{extensión-archivo}>
- Parámetros de la url:
  - {nombre-archivo}: Nombre del archivo a descargar. El archivo que descargaremos será la firma de una entrega.
  - {extensión-archivo}: Extensión del archivo a descargar. Como será una imagen es “.png”.
- Subir una firma al sistema:
  - Tipo de petición/verbo HTTP: **POST**
  - URL:  
<http://despliquesigi.azurewebsites.net/Servicios/FileUploadService.svc/UploadFile?fileName={nombre-archivo}>
  - Parámetros de la url:
    - {nombre-archivo}: Nombre del archivo a subir. El archivo que subiremos será una imagen en png (la extensión se añadirá automáticamente).
  - Cuerpo de la petición HTTP: El cuerpo será un objeto de tipo Stream. La clase Stream genera archivos binarios. [9]

Para la realización de esta aplicación o servicio web hemos seguido el libro “Programming WCF Services” de la editorial O’Reilly, desde aquí nos muestra como crear los servicios con los diferentes tipos de peticiones HTTP, cómo realizar el consumo de los servicios de WCF y cómo integrarlo en Microsoft Azure [10]

### 5.3.4. Aplicación cliente con WPF

Para crear la aplicación de escritorio hemos creado un proyecto basado en la tecnología Windows Presentation Foundation o WPF, esta tecnología ha sido comentada en el capítulo 3.1.7.

#### 5.3.4.1. Patrón de diseño. Modelo-vista-modelo de la vista

Hemos utilizado el patrón de diseño Modelo-vista-modelo de la vista o MVVM (model-view-viewmodel). La característica de este patrón es intentar desacoplar lo máximo posible la interfaz de usuario de la lógica de la aplicación:

- **Modelo:** El modelo será reutilizado del proyecto ya creado basado en WCF. Para ello deberemos añadir las referencias en las propiedades del proyecto. Hemos dado click derecho en “Referencias” y hemos añadido el subproyecto “ApiRestWCF”:





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

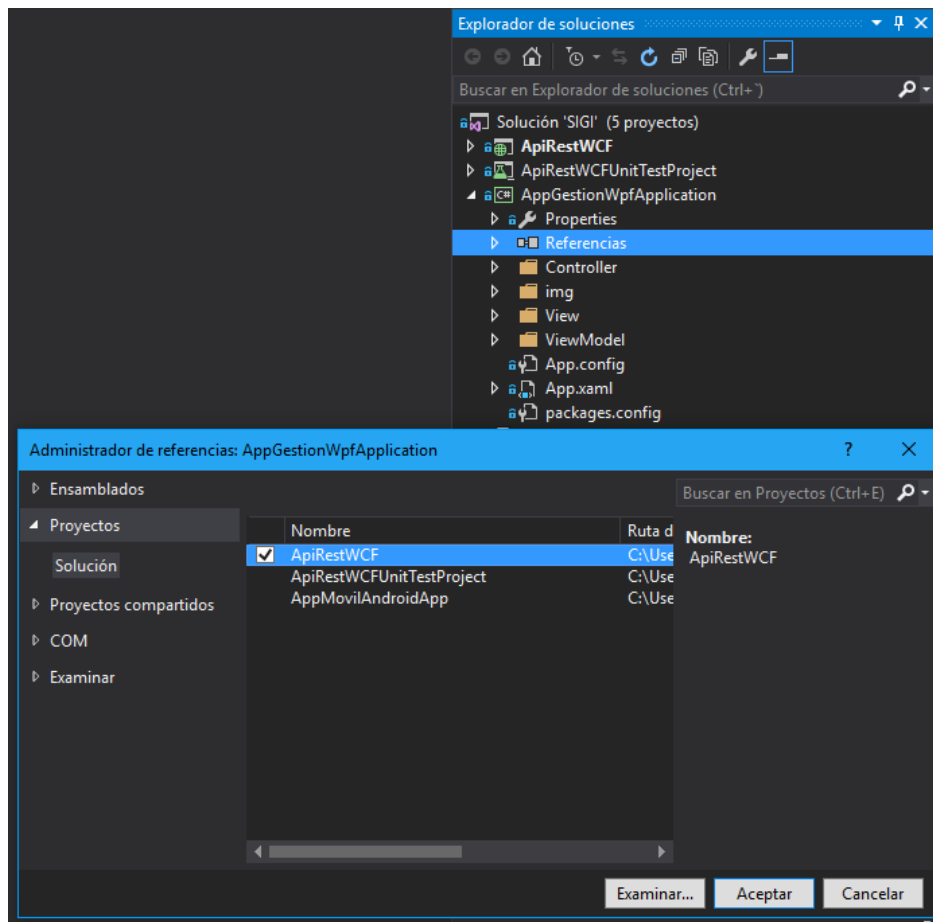


Figura 75 - Añadimos modelo en la aplicación de escritorio

De esta manera importamos y reciclamos las clases ya creadas en el anterior proyecto.

**Vistas.** La genialidad de WPF hace posible que dividamos la vista del modelo de la vista de tal manera podemos manipular la vista sin tocar la lógica de la aplicación. Hemos creado vistas independientes de cada ventana de la aplicación. Cuando creamos una ventana se crea un archivo “.xaml” que contiene en formato “.xml” el contenido de la presentación de la ventana.

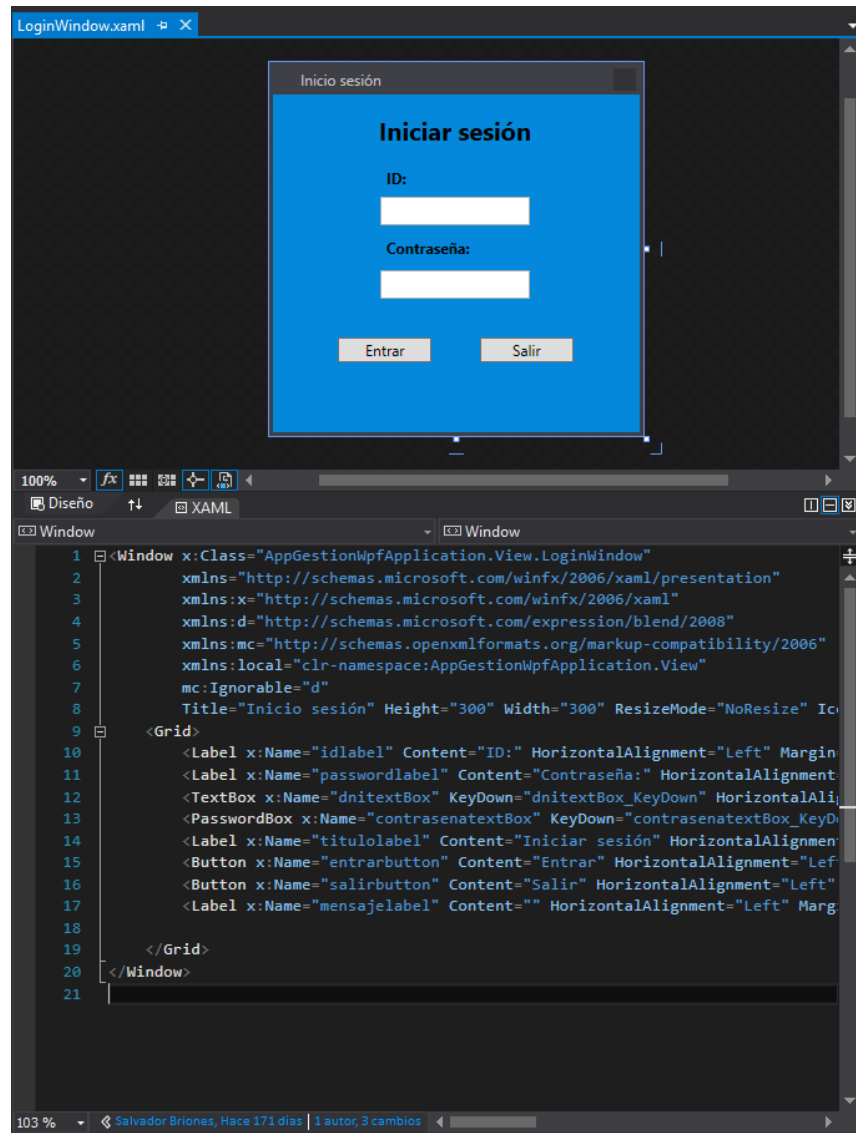



Figura 76 - Vista del patrón MVVM en la aplicación de escritorio

En las Figura 76, podemos observar la división entre el diseño de los elementos visuales y su definición en forma de XML. La vista diseño es una representación de los elementos creados en XAML. Cada etiqueta XML corresponde a un elemento gráfico de la interfaz gráfica de la ventana.

 **Modelo de la vista.** Esta parte del patrón de diseño MVVM contiene toda la lógica de la presentación o vista. Hace de intermediario entre la lógica y la vista. Para darnos a la idea de su función digamos que es donde se define el comportamiento cuando



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

creamos el evento al pulsar un botón de la interfaz gráfica. Cada vista tiene un modelo de la vista y en el proyecto está definido como una clase C#.

A continuación, veremos cómo hemos resumido este patrón de diseño en nuestra aplicación de escritorio:

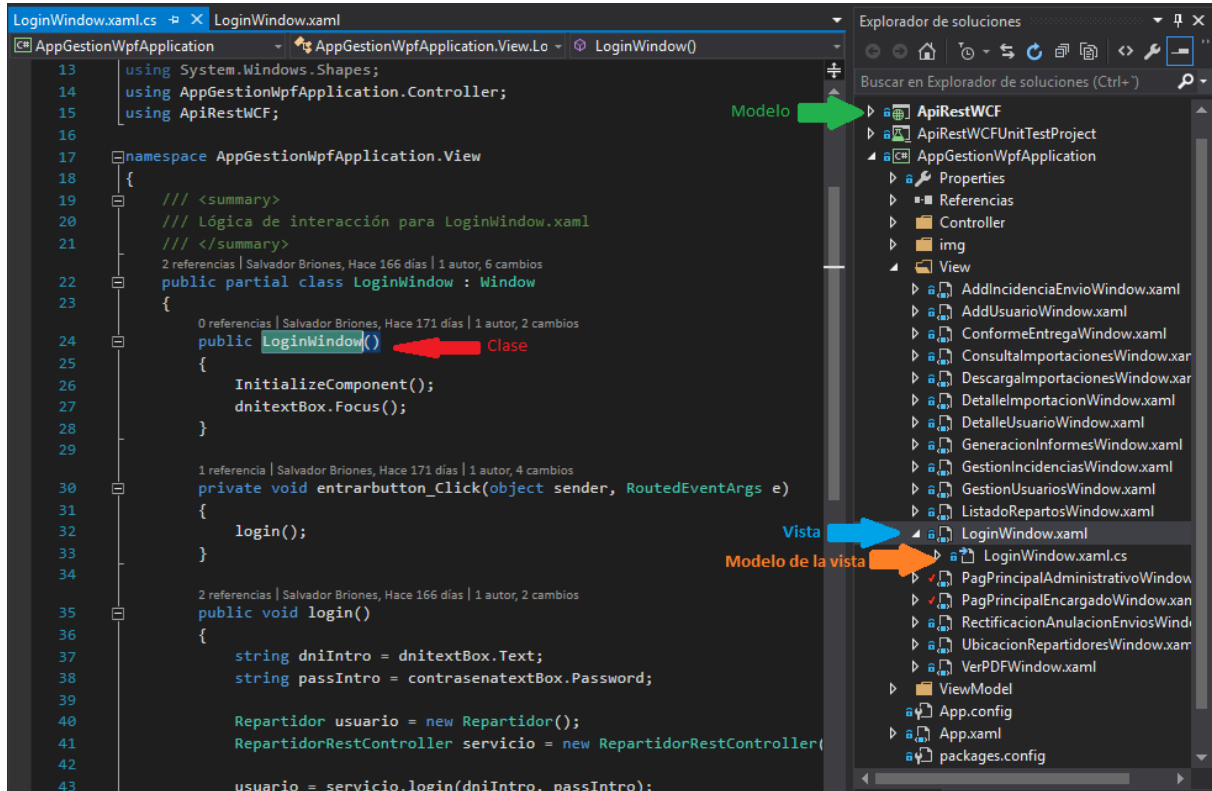


Figura 77 - Patrón de diseño MVVM en la aplicación de escritorio

### 5.3.4.2. Creación de documentos PDF

Dentro de los objetivos y requisitos funcionales del proyecto (capítulo 5.1.3) es la de impresión y generación de documentos de informes y reportes. Los documentos PDF o Portable Document Format se caracteriza por ser multiplataforma ya que puede ser presentado en los principales sistemas operativos, pero, sobre todo, la característica que nos ha sido útil es en los archivos PDF el formato no se pierde con el envío a otros sistemas informáticos, es decir, el contenido se queda en de la misma organización en la que se creó.

Para la generación y creación de documentos PDF nos hemos apoyado de una librería externa Open Source llamada “iTextSharp”. iTextSharp se encuentra de manera gratuita desde el administrador de paquetes NuGet dentro de Visual Studio. Inicialmente la herramienta fue desarrollada en lenguaje Java pero gracias a la empresa After Logic ha trasladado y traducido la librería a lenguaje C#. Han creado también documentación en inglés de la que me ha seguido de ayuda para la creación de los documentos. [11]





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

Una de las funcionalidades como ya hemos dicho es la de imprimir el listado de reparto de un repartidor, ver que envíos lleva asignados en reparto. En la siguiente figura, Figura 78, podemos ver el listado de repartos que tiene (ficticiamente) un de los repartidores:

Nº envío	Nombre destinatario	Localidad	Código postal	Nº bultos	Peso (kg)
9874512	Banco BBVA sucursal 5647	Almeria	04001	1	1
2147895	Oxigeno 60 SL	Viator	04240	8	17

Total bultos: 9  
Peso total (kg): 18


Imprimir listado  [Volver al menu](#)

Figura 78 - Captura de la ventana Listado de repartos en SIGI

Vemos que están todos los datos necesarios que necesita el repartidor, no obstante, estos datos solo los pueden ver los empleados de tipo Administrativo o Encargado. Para que el repartidor tenga un listado físico detallando la mercancía que lleva, se ha creado la funcionalidad de imprimir este listado:



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

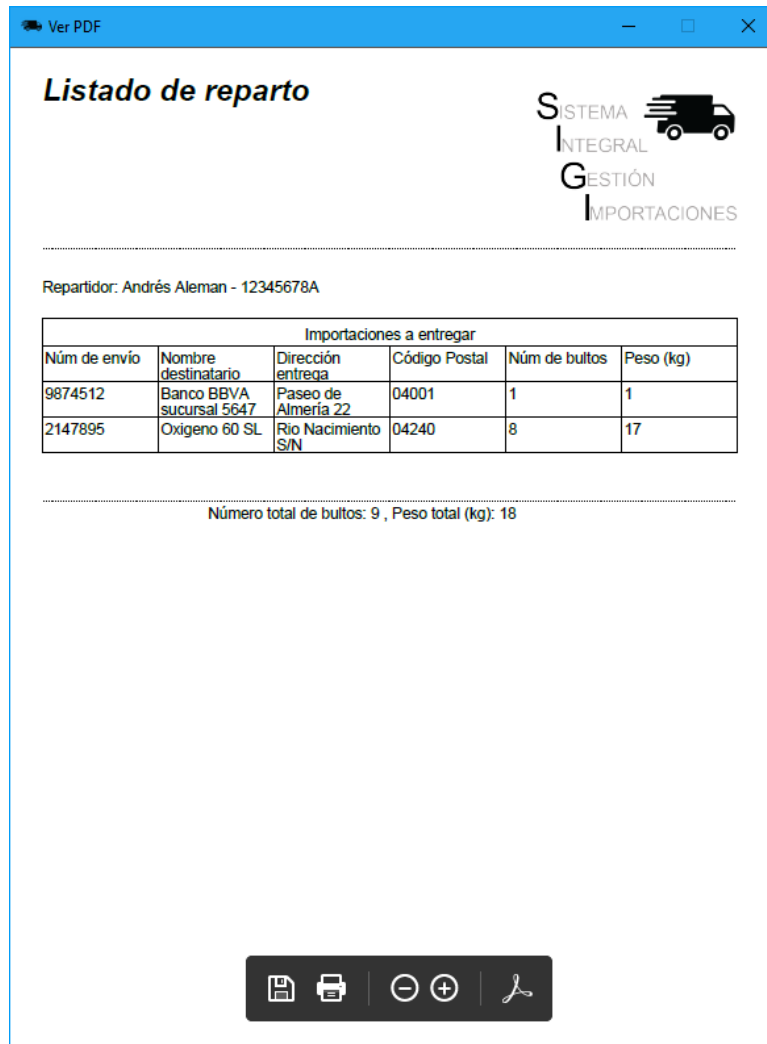


Figura 79 - Documento PDF generado del listado de reparto en SIGI

Podemos ver que genera un documento PDF con los mismos datos que aparecía en el sistema.

Podemos ver que se ha insertado en el encabezado del documento una imagen del logo de la aplicación, una tabla para visionar mejor los datos de los envíos en reparto y demás datos relevantes para el repartidor.

Cada elemento del PDF se ha ido insertado por párrafos. A continuación, explicaremos de forma detallada con comentarios (letra color verde) un fragmento de la implementación realizada para la generación de este documento PDF:



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

```
private void crearPDF(string rutaPDF)
{
    try
    {
        //Generamos un documento con medida A4 y márgenes de 25mm.
        Document = new Document(PageSize.A4, 25, 25, 25, 25);
        //Creamos el archivo binario donde se guardará el documento
        FileStream = new FileStream(rutaPDF, FileMode.OpenOrCreate);
        PdfWriter writer = PdfWriter.GetInstance(document, fileStream);
        document.Open();

        //Creamos el primer párrafo o encabezado del documento
        Paragraph parrafoTitulo = new Paragraph();
        //Establecemos el estilo de fuente a usar
        Font fuenteTitulo = FontFactory.GetFont(FontFactory.HELVETICA, 24,
Font.BOLDITALIC);
        //Damos título al documento
        Chunk titulo = new Chunk("Listado de reparto", fuenteTitulo);

        //Añadimos una línea separadora horizontal
        DottedLineSeparator dottedline = new DottedLineSeparator();
        dottedline.Offset = -2;
        dottedline.Gap = 2f;

        //Añadimos el título al párrafo
        parrafoTitulo.Add(titulo);

        //Creamos la imagen
        iTextSharp.text.Image jpg =
iTextSharp.text.Image.GetInstance(System.IO.Directory.GetCurrentDirectory() + "\\\"
+ "LOGO.png");

        //Redimensionamos la imagen
        jpg.ScaleToFit(160f, 140f);
        //Damos espacio blanco a la imagen
        jpg.SpacingBefore = 10f;
        jpg.SpacingAfter = 1f;
        //Alineamos la imagen
        jpg.Alignment = Element.ALIGN_RIGHT;

        //Añadimos la imagen al párrafo
        parrafoTitulo.Add(jpg);
        //Añadimos la línea horizontal al párrafo
        parrafoTitulo.Add(dottedline);

        //Añadimos el párrafo al documento
        document.Add(parrafoTitulo);
    }
}
```

Este fragmento de código corresponde a la implementación realizada para crear el encabezado del documento PDF.







### 5.3.4.3. Integración con Google Maps

En el capítulo 5.1.3. Requisitos funcionales detallamos como objetivo y requisito funcional el consultar la ubicación de un repartidor.

El gigante Google tiene numerosas herramientas de desarrollo, uno de sus productos más usados es Google Maps. Google Maps es una herramienta que ofrece imágenes de mapas, fotografías por satélite de todo el mundo, simulaciones de ruta entre diferentes ubicaciones y numerosas funcionalidades más ofrecidas en la geolocalización de dispositivos.






Google Maps API es un servicio web simple para el uso de las herramientas de Google Maps. En concreto, nosotros usaremos la API Static Maps API [12].

Static Maps API implementa mapas como imágenes en aplicaciones y sitios web según los parámetros de URL enviados a través de una solicitud de HTTPS estándar.

Básicamente, lo que hemos desarrollado es crear una imagen generada a partir de una URL, ésta URL es dinámica y genera un mapa según la dirección introducida.

<https://maps.googleapis.com/maps/api/staticmap?size=685x380&zoom=15&maptype=roadmap&markers=size:mid%7Ccolor:red%7C{DIRECCIÓN}>

Los parámetros por modificar son:

-  size: Especificamos el tamaño que tendrá la imagen. 685x380píxeles
-  zoom: La ampliación para tener un menor o mayor detalle del mapa. x15 de zoom
-  maptype: El tipo de mapa. Mapa de carretera: roadmap
-  markers: El formato del marcador que señalará la ubicación. Marcador mediano de color rojo: size:mid-color:red
-  {DIRECCION}: La ubicación física, pueden ser coordenadas o una dirección postal física, por ejemplo: Calle Almería 33, El Ejido.

El último parámetro es registrado por la aplicación móvil que guarda de manera automática la ubicación del repartidor cada vez que utiliza la aplicación.

A continuación, mostraremos un ejemplo desde la aplicación de escritorio:



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

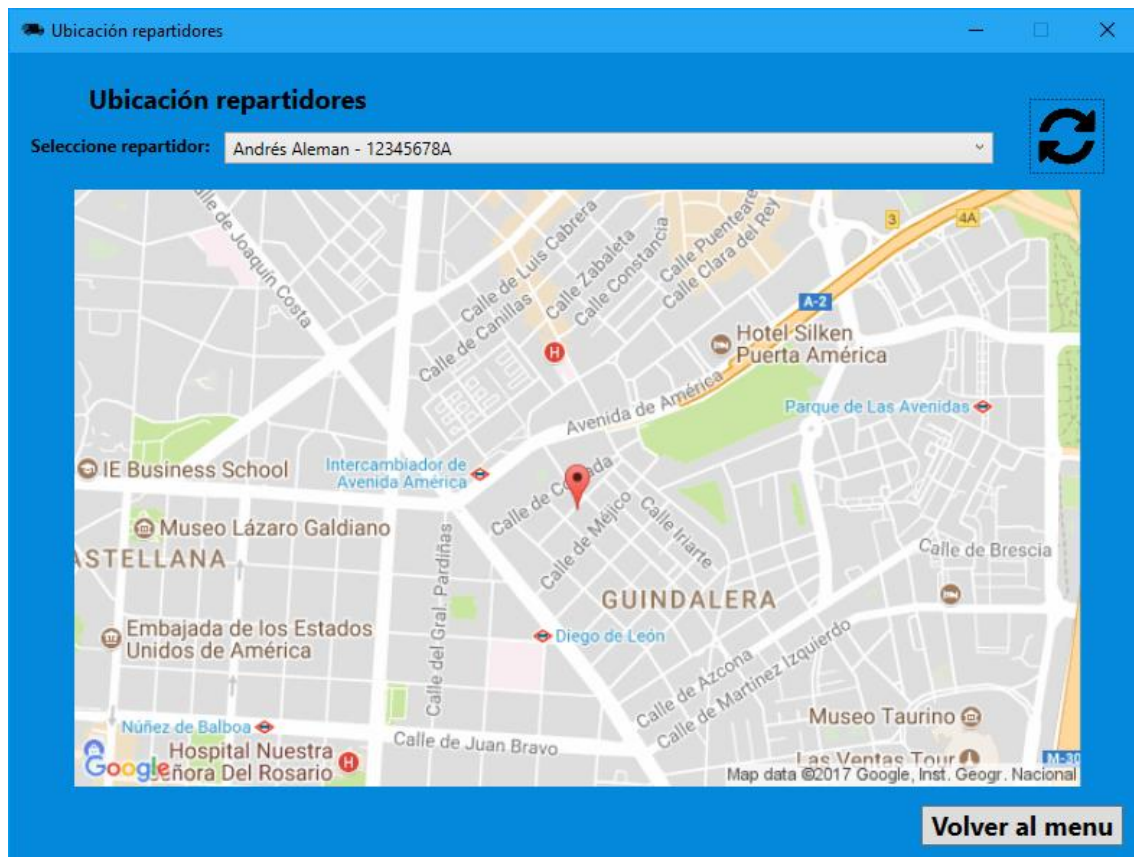


Figura 80 - Captura de la ventana Ubicación repartidores en SIGI

Podemos ver que el repartidor se encuentra en la ubicación que muestra el marcador de color rojo, si seleccionamos otro repartidor nos mostraría su ubicación también.

### 5.3.4.4. Consumo de datos del servicio web Restful

Como ya hemos visto en el capítulo 5.3.3, hemos creado un servicio web basado en Rest en el cual, cuando nosotros realizamos una petición HTTP desde una URL este nos devuelve la petición con los datos en formato JSON.

Hemos realizado consultas de datos este servicio web para después formatearlos en la aplicación cliente de escritorio. En nuestro proyecto hemos creado una carpeta llamada “Controller”, en esta carpeta hemos creado las clases que interpretan y realizan las consultas para el servicio web.

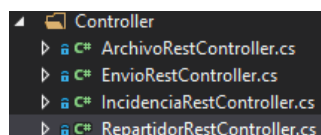


Figura 81 - Distribución controladores en el proyecto WPF



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

Podemos ver que para cada entidad o tabla de la base de datos existe una clase que hace de controlador con el servicio Rest.

Por ejemplo, para realizar la consulta de todos los usuarios del sistema, hemos realizado una petición HTTP a la URL:

<http://despliguesigi.azurewebsites.net/Servicios/RepartidorService.svc/getRepartidores>

Se ha creado un método para que realice la consulta y que devuelva todos los usuarios en forma de lista parametrizada es decir como List<Repartidor>:

```
//Método que realiza el formateado de los datos, de Json a una lista de repartidores.
private List<Repartidor> jsonToRepartidores(string json)
{
    List<Repartidor> result = new List<Repartidor>();

    json = eliminarRaiz(json, "getRepartidoresResult");
    MemoryStream ms = new MemoryStream(Encoding.Unicode.GetBytes(json));
    DataContractJsonSerializer ser2 = new
DataContractJsonSerializer(typeof(List<Repartidor>));
    result = (List<Repartidor>)ser2.ReadObject(ms);
    ms.Close();
    return result;
}
//Método que realiza la petición HTTP
public List<Repartidor> getRepartidores()
{
    List<Repartidor> resultado = new List<Repartidor>();
    HttpClient client = new HttpClient();
    HttpResponseMessage wcfResponse = client.GetAsync(this.urlServicio +
"getRepartidores").Result;
    HttpContent stream = wcfResponse.Content;
    var data = stream.ReadAsStringAsync();

    resultado = jsonToRepartidores(data.Result);
    return resultado;
}
```

Una vez que tenemos los datos, hemos mostrado los datos en forma de tabla gracias al DataBinding de WPF:



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

Nombre empleado	Puesto del empleado	Teléfono	Nº de identificación
Andrés Aleman	1181GDY	671884554	12345678A
Mario Cesar Marin	5124HZV	694512574	X7977051X
Salvador Briones	ADMINISTRATIVO	671884554	1234
Néstor Agüero	ENCARGADO	679953679	123456
María Jose García	ADMINISTRATIVO	950173838	89745615C

Figura 82 - Captura de pantalla Gestión de usuarios en SIGI

En la Figura 82, vemos que los usuarios aparecen en una tabla en la que cada columna aparece los datos más relevantes del usuario.

En este caso, se trata de un ejemplo de la obtención de datos desde el servicio web Rest a la aplicación cliente, pero durante el desarrollo de la aplicación se ha realizado el uso de peticiones a los demás servicios creados para envíos, incidencias, entregas, imágenes.





## 5.4. Implementación y desarrollo aplicación móvil

En este capítulo describiremos los puntos más relevantes realizados en el desarrollo de la aplicación móvil. Al igual que la aplicación de escritorio, el desarrollo integro lo hemos realizado con la herramienta Visual Studio (véase capítulo 4.2.).

Explicaremos la forma utilizada para estructurar este proyecto, el uso de las diferentes tecnologías, el uso de librerías externas para algunas funcionalidades principales de la aplicación y uso de la aplicación como aplicación cliente y su enlace con el servicio web Rest.

### 5.4.1. Creación y estructura del proyecto

El desarrollo de la aplicación móvil se ha realizado en un solo proyecto. A partir de la solución ya creada, hemos añadido un proyecto de tipo “Xamarin.Android”. Durante la creación y el desarrollo del proyecto hemos seleccionado las propiedades de este, una de estas propiedades es seleccionar la versión mínima en la que se ejecutará la aplicación: hemos seleccionado como versión mínima Android 4.2 o Nivel 17 API. Hemos elegido esta versión dado que actualmente está en el punto de ruptura de uso dentro de los dispositivos móviles existentes [13].

El patrón de diseño utilizado para este proyecto ha sido una Modelo Vista Controlador o MVC:

- 🚚 **Modelo.** Se ha remasterizado el modelo para acoplarlo al proyecto dado que las librerías utilizadas en el proyecto ApiRestWCF no son compatibles con el proyecto de Xamarin Android.
- 🚚 **Vista.** En Android no existen ventanas, pero si Actividades, estas actividades muestran contenidos en la pantalla para que el usuario pueda interactuar. Así pues, entre más estados de uso tenga la aplicación más actividades o vistas tendrá. Hay que mencionar que las vistas en Android son independientes de los datos o modelos y por tanto cada vista tiene un modelo y una vista. Es un tanto parecido al modelo MVVM que hemos visto en el anterior capítulo 5.3.1.
- 🚚 **Controlador.** Se han creado una serie de controladores para que interactúen con los medios externos. En este caso se han creado para realizar las peticiones HTTP del servicio web Rest.





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

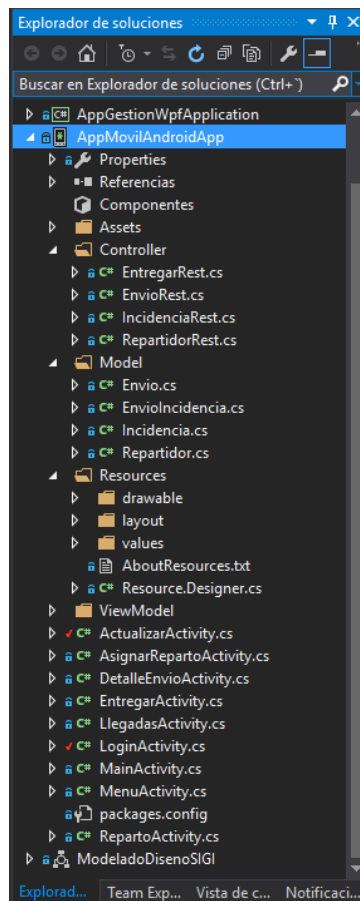


Figura 83 - Estructura del proyecto de la aplicación móvil

### 5.4.2. Uso de Xamarin en el proyecto

Como hemos mencionado en el capítulo 3.1.5. de Tecnologías, Xamarin es un sistema y plataforma para el desarrollo de aplicaciones móviles. Su potencia está en el desarrollo en un mismo lenguaje para diferentes plataformas móvil.

No obstante, vamos a desarrollar una sola plataforma utilizando esta herramienta, hemos desarrollado un proyecto Xamarin-Android. Las características de este TFG han hecho que nos planteamos el desarrollo en más plataformas dado que los objetivos ya están fijados y los recursos temporales ya están concretados.

Por tanto, la ventaja que nos dará esta herramienta es la de crear un proyecto en el mismo lenguaje que se ha hecho con los demás proyectos en Visual Studio, es decir, C#.

La curva de aprendizaje con Xamarin ha sido mayor dado que ya conocíamos el lenguaje nativo de programación y el modelo de vistas está basado en XML, también conocido.

Xamarin está integrado en la versión de Visual Studio 2015 con la actualización Update 3 por defecto, por tanto, no hemos tenido que instalar ninguna herramienta más.



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

La creación de las interfaces gráficas ha sido bastante cómoda dado que existe una herramienta de creación de elementos visuales bastante completa.

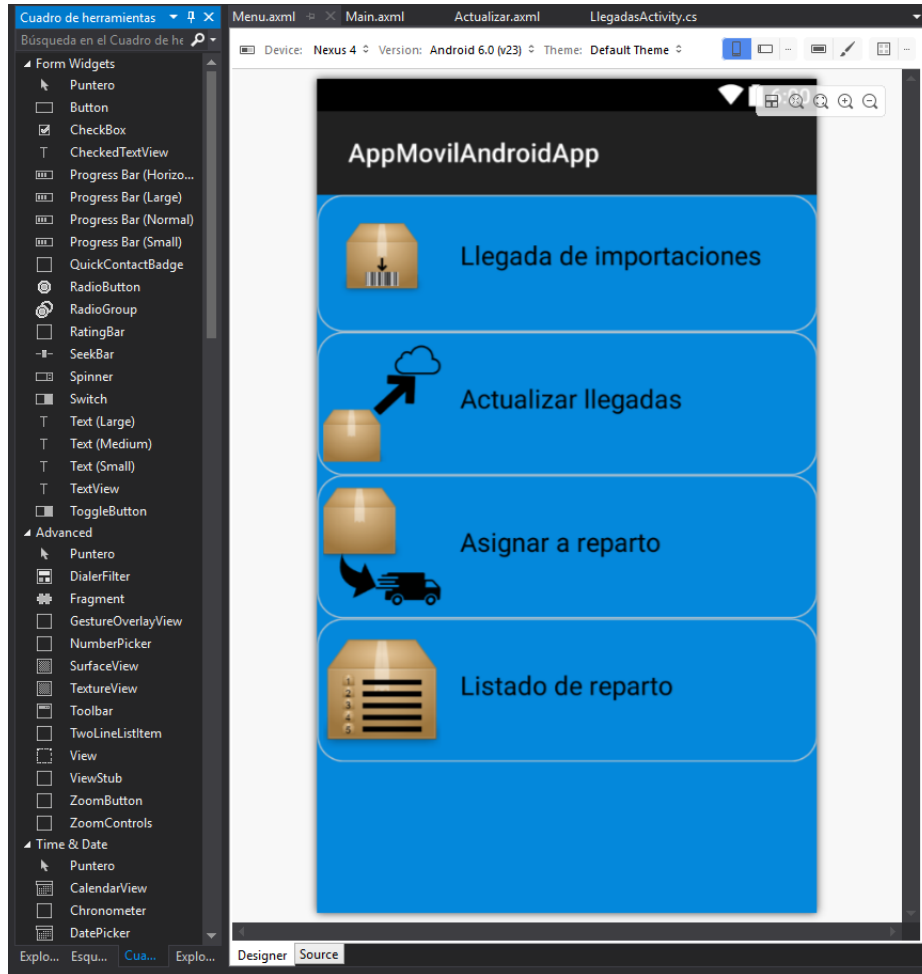


Figura 84 - Creación de interfaces gráficas con Xamarin en Visual Studio

Podemos ver, al igual que WPF, existen dos tipos de vistas de edición, una vista gráfica donde podemos añadir elementos gráficos y una vista de código fuente donde podemos añadir los elementos con etiquetas XML.

Al igual que pasa en el desarrollo nativo de Android en otros entornos de desarrollo integral, las actividades contienen Layout que son las vistas gráficas a su lado están las Activity que son las clases que contienen la lógica de la aplicación.



### 5.4.3. Uso de código de barras

A efecto prácticos y poniéndonos en situación con las tareas a llevar con la aplicación móvil, una de estas tareas era la de controlar y entregar las importaciones que llegaban a las instalaciones. Cuando las importaciones llegan desde la aplicación móvil deberemos dar entrada o registrar la llegada del envío, para agilizar esta tarea, se pretende realizar la lectura del código de barras de la etiqueta adherida en la importación.

Este código de barras contiene el número de envío de la importación y el número de bulto de la importación. Para mantener un formato único de lectura, se han generado código de barras con el siguiente formato:

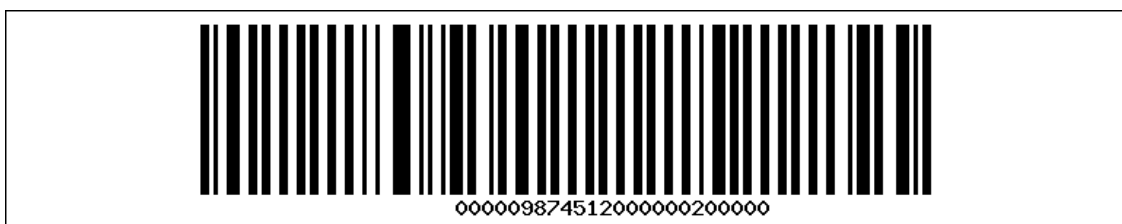


Figura 85 - Ejemplo de código de barras

En la Figura 85, muestra un ejemplo del formato del código de barras. Se trata de la importación con número de envío: 9874512 y pertenece al bulto número 2. En este caso, para completar la llegada correcta de la importación se deberá leer la etiqueta del bulto faltan de este envío.

Por otro lado, en el capítulo 4.7, describimos el modelo de Smartphone a utilizar, se trata de un terminal móvil que contiene una cámara fotográfica trasera y posibilita la lectura de código de barras de los envíos.

Para el desarrollo de proyecto, hemos utilizado una librería de libre uso llamada FastAndroidCamera. Esta librería ha sido descargada desde el administrador de paquetes NuGet de Visual Studio. Al instalarla en nuestro proyecto, se descarga un proyecto prueba donde muestran ejemplo de uso.

Para que funcionará en nuestro proyecto, tuvimos que añadir la librería como referencia externa e inicializar la clase que realiza la lectura:

```
MobileBarcodeScanner.Initialize(Application);  
var scanner = new ZXing.Mobile.MobileBarcodeScanner();  
scanner.BottomText = "Pulse la pantalla para enfocar";  
var result = await scanner.Scan();
```

La variable "result" guarda el valor del código de barras leído.





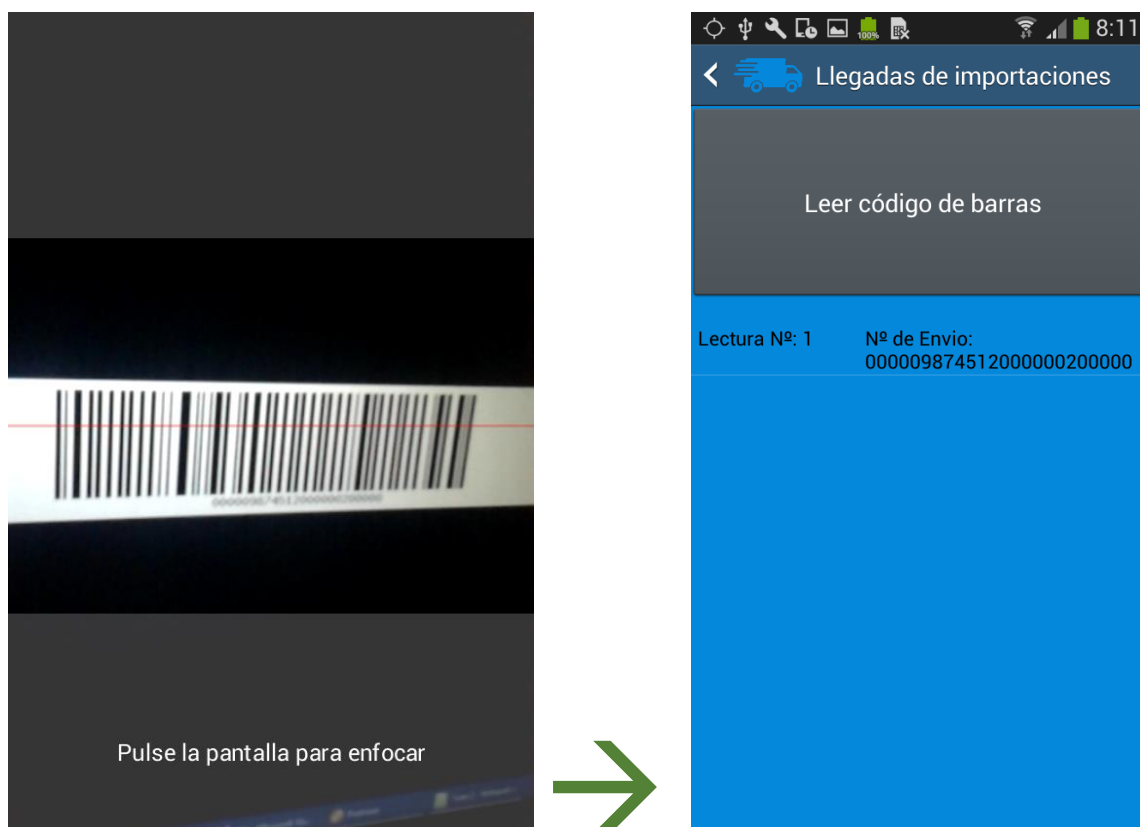


Figura 86 - Lectura de código de barras desde la aplicación móvil

Cuando el código es leído por la cámara el valor es transmitido a la aplicación para que realice el procesamiento que realiza en esa función usada. La función de lectura de código de barras puede ser para dar entrada, para actualizar la llegada del envío o si es para asignar el envío a reparto.

#### 5.4.4. Realización y creación de firmas grafológicas

Cuando se realiza la entrega de un envío se da por finalizado el servicio. El servicio realizado es exportar la mercancía desde el almacén de origen de la ciudad de origen, realizar el tránsito de la importación hasta la llegada al almacén de destino de la ciudad de destino para que finalmente se realice el reparto y entrega del envío.

Cuando se entrega la mercancía, es necesario que el consignatario o persona quien recibe la mercancía dé conformidad a la entrega y por tanto de por finalizado el servicio por parte de la empresa de transporte. Una conformidad de entrega es correcta cuando se manifiesta por algún tipo de escrito u otro tipo de vía de comunicación que el servicio se ha realizado y entregado a una persona, empresa o razón física.



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

Para justificar la entrega a una persona, empresa o razón física es necesario su número de identificación fiscal ya sea NIF, DNI o NIE en España y una firma grafológica de la persona que recibió la mercancía que, es quien tiene el relevo de custodia del envío.

Una vez justificada la obligación de incluir la funcionalidad de incluir un sistema de firma grafológica en la aplicación móvil, explicaremos como hemos implementado y desarrollado esta función.

Por medio de una librería Open Source de Xamarin llamada Xamarin.Controls.SignaturePad hemos realizado la captura, guardado y visualización de una firma simple. Mediante el administrador de paquetes NuGet de Visual Studio hemos descargado e instalado esta librería a nuestro proyecto móvil.

Una vez realizado los pasos anteriores de instalación, hemos implementado la librería gracias a la documentación aportada por la librería.

```
private void cargarModuloFirma()
{
    SignaturePadView signature;
    signature = FindViewById<SignaturePadView>(Resource.Id.signatureView);

    signature.Caption.Text = "Firma de entrega";
    signature.Caption.SetTypeface(Typeface.Serif, TypefaceStyle.BoldItalic);
    signature.Caption.SetTextSize(global::Android.Util.ComplexUnitType.Sp, 13f);
    signature.BackgroundColor = Android.Graphics.Color.Rgb(255, 255, 200); // a light
yellow.
    signature.StrokeColor = Android.Graphics.Color.Black;

    var layout = new RelativeLayout.LayoutParams(ViewGroup.LayoutParams.MatchParent,
ViewGroup.LayoutParams.MatchParent);
    layout.AddRule(LayoutRules.CenterInParent);
    layout.SetMargins(20, 20, 20, 20);
    signature.BackgroundImageView.LayoutParameters = layout;

    // You can change paddings for positioning...
    var caption = signature.Caption;
    caption.SetPadding(caption.PaddingLeft, 1, caption.PaddingRight, 25);

    // get our button from the layout resource,
    // and attach an event to it
    editTextDni = FindViewById<EditText>(Resource.Id.editTextInputDniEntregar);
    botonEntregado = FindViewById<Button>(Resource.Id.botonBntEntregar);
    botonEntregado.Click += envioEntregado_Click;
}
```

En el fragmento de código anterior vemos como configuramos el formato y tamaño del recuadro de firma. Posteriormente la firma se guarda como un objeto Bitmap o mapa de bits que seguidamente comprimimos en formato PNG para ser subida al servidor con la entrega.





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

A continuación, mostraremos la interfaz gráfica que muestra el uso de la firma a la hora de entregar un envío:



Figura 87 - Firma de conformidad de entrega en la aplicación móvil

Podemos ver que existe la posibilidad de borrar la firma pulsando en “Clear”.

### 5.4.5. Consumo de datos del servicio web Rest

Al igual que en la aplicación cliente de escritorio, en la aplicación móvil vamos a comunicarnos con el servicio web Rest para extraer y consultar los datos necesarios para el uso de la aplicación.

En el capítulo 5.3.3., hemos creado un servicio web basado en Rest en el cual, cuando nosotros realizamos una petición HTTP desde una URL este nos devuelve la petición con los datos en formato JSON.

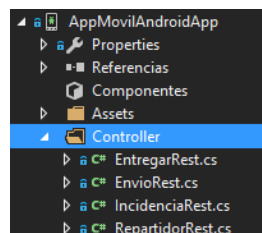


Figura 88 - Estructura controladores en la aplicación móvil



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

En la Figura 88, vemos como hemos estructurado los controladores que hará de intermediarios entre la lógica de la aplicación en este proyecto y el servicio web Rest creado y desplegado en la nube Azure.

```
private List<Incidencia> getIncidencias<T>()
{
    try
    {
        Url =
        "http://despliguesigi.azurewebsites.net/Servicios/IncidenciaService.svc/";
        var uri = new Uri(Url + "getIncidencias");

        var response = client.GetAsync(uri).Result;
        if (response.IsSuccessStatusCode)
        {
            string json = response.Content.ReadAsStringAsync().Result;
            JObject jo = JObject.Parse(json);
            List<Incidencia> listado = jo.SelectToken("getIncidenciasResult",
            false).ToObject<List<Incidencia>>();

            return listado;
        }
    }
    catch (Exception ex)
    {
        System.Diagnostics.Debug.WriteLine(ex.Message);
    }
    return default(List<Incidencia>);
}
```

Como podemos ver, este método nos devuelve en forma de lista todas las incidencias del sistema, con los datos, lo único que tenemos que hacer es formatearlos en la aplicación.

Este listado es utilizado para dar incidencia por no entregar el envío, está formateado en forma de lista de selección tal y como muestra la figura:

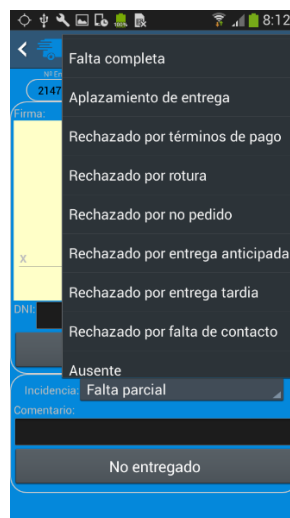


Figura 89 - Listado de incidencias desde la aplicación móvil



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

Al igual que se ha realizado la petición de datos del servicio web Rest desde la aplicación móvil para consultar la lista de incidencias en el sistema, también se han realizado consultas de todo tipo para que las funcionalidades de la aplicación funcionen correctamente.

### 5.5. Pruebas

Para el testeo y pruebas de las aplicaciones se han utilizado dos tipos métodos. El primero es el más rudimentario, pero de mucha utilidad y ha sido la utilización de amigos y conocidos los cuales han proporcionado un feedback continuo sobre cosas que no funcionaban correctamente o que no les gustaba.

El segundo tipo de método ha sido más formal, hemos utilizado para el testeo de datos con el servidor la herramienta Postman.

```
1 - {
2 -   "getRepartidoresResult": [
3 -     {
4 -       "contrasena": "1234",
5 -       "dniRepartidor": "12345678A",
6 -       "domicilio": "40,4376657399354, -3,67383994537177",
7 -       "idRepartidor": 1,
8 -       "matricula": "1181GDY",
9 -       "nombreCompleto": "Andrés Aleman",
10 -      "telefono": "671884554"
11 -    },
12 -    {
13 -      "contrasena": "Aa154784",
14 -      "dniRepartidor": "X7977851X",
15 -      "domicilio": "36.775582, -2.688764",
16 -      "idRepartidor": 2,
17 -      "matricula": "5124HZV",
18 -      "nombreCompleto": "Mario Cesar Marin",
19 -      "telefono": "694512574"
20 -    },
21 -    {
22 -      "contrasena": "1234",
23 -      "dniRepartidor": "1234",
24 -      "domicilio": "36,7762884151328, -2,80618483464015",
25 -      "idRepartidor": 3,
26 -      "matricula": "ADMINISTRATIVO",
27 -      "nombreCompleto": "Salvador Briones",
28 -      "telefono": "671884554"
29 -    },
30 -    {
31 -      "contrasena": "123456",
32 -      "dniRepartidor": "123456",
33 -      "domicilio": "DOMICILIO PRIVADO",
34 -      "idRepartidor": 4,
35 -      "matricula": "ENCARGADO",
36 -      "nombreCompleto": "Néstor Agüero",
37 -      "telefono": "679953679"
38 -    },
39 -    {
40 -      "contrasena": "123456",
41 -      "dniRepartidor": "89745615C",
42 -    }
43 -  ]
44 - }
```

Figura 90 - Testeo de datos con la herramienta Postman

Para la comprobación del funcionamiento del servicio web Rest con la tecnología WCF hemos creado test unitarios de Visual Studio:





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

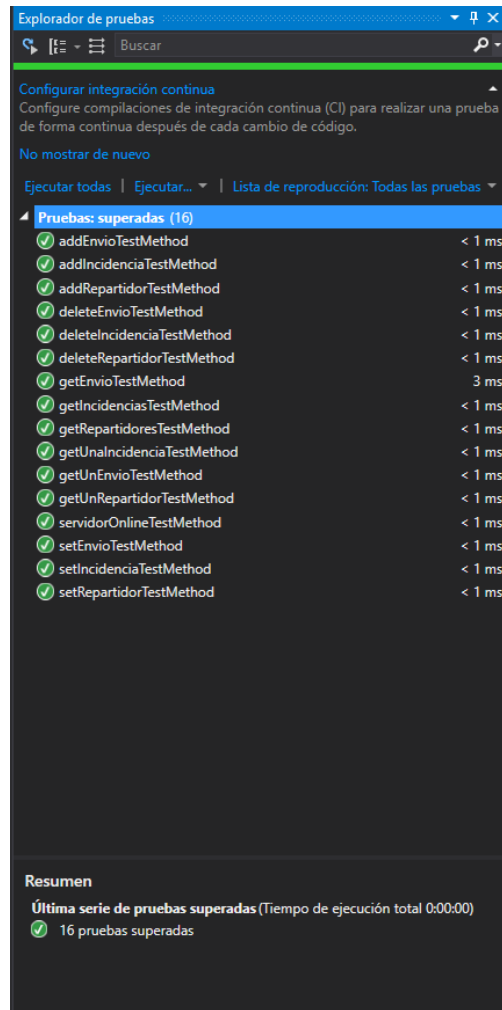


Figura 91 - Explorador de pruebas en Visual Studio



## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

### 5.6. Despliegue y producción de SIGI

Durante el desarrollo del sistema completo se ha ido implementado las diferentes funcionalidades del sistema. Mediante el uso de las dos herramientas Visual Studio y Visual Studio Team Service hemos creado y desplegado nuestro sistema en Azure.

#### 5.6.1. Despliegue del sistema

En el capítulo 4.4 hemos configurado y unido nuestros proyectos para que se realice la integración continua del proyecto completo. La integración continua se ha configurado para que se active siempre y cuando las pruebas/test sean correctos y cuando las construcciones de los proyectos se realicen sin problemas.

En la herramienta Team Services es muy completa y realiza un análisis exhaustivo a la hora de realizar la construcción del proyecto.

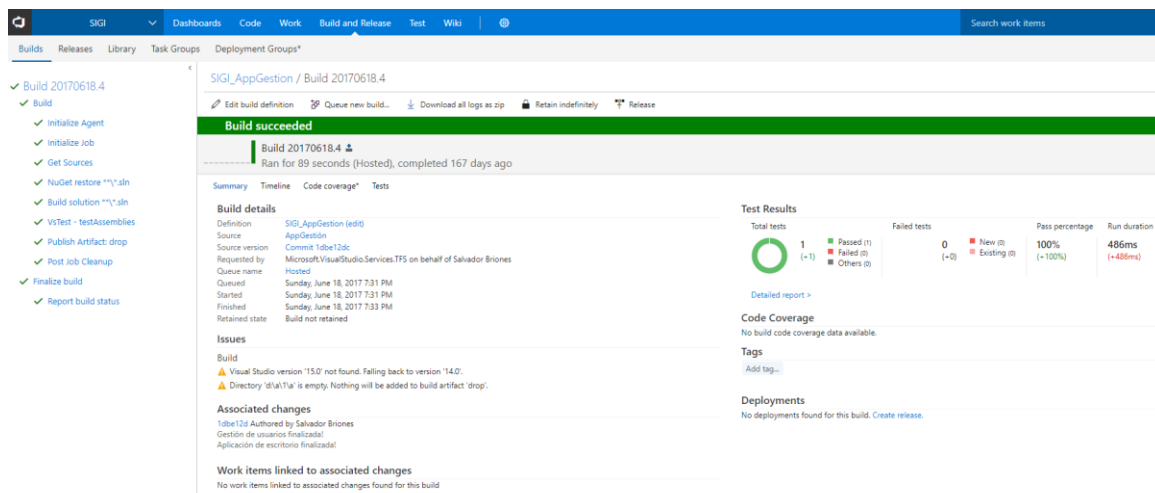


Figura 92 - Ejemplo construcción del proyecto

En la parte izquierda del informe muestra las tareas realizadas que anteriormente hemos configurado. Podemos ver que inicializa el sistema de construcción, descarga el repositorio, restaura los paquetes NuGet, y la solución de Visual Studio, ejecuta los test de Visual Studio y finalmente crea los artefactos de construcción.

Vemos cuando empieza y finaliza la construcción y el tiempo que ha transcurrido. Por otro lado, hace un análisis de los test realizados en el proyecto.

También muestra las incidencias producidas durante la construcción y el último commit o cambios realizados en el repositorio.

Si la construcción es correcta, el sistema automatiza la integración continua y realiza el despliegue configurado en Azure:





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

The screenshot shows the 'All release definitions' page in Azure DevOps. The table below represents the data shown in the interface:

Title	Release Definition	Environments	Build	Branch	Created	Created By	Description
Release-47	New Azure App Service...	✓	20170730.1 (Build)	master	7/30/2017	Salvador Briones	Triggered by SIGI_Construcción para desarrollo ...
Release-46	New Azure App Service Dep...	✓	20170728.1 (Build)	master	7/28/2017	Salvador Briones	Triggered by SIGI_Construcción para desarrollo ...
Release-45	New Azure App Service Dep...	✓	20170725.4 (Build)	master	7/25/2017	Salvador Briones	Triggered by SIGI_Construcción para desarrollo ...

Figura 93 - Historial de despliegues del sistema

The screenshot shows the 'New Azure App Service Deployment 06-Jun / Release-47' details page. The deployment summary is as follows:

Step	Action
Environment 1	...
Pre-deployment approval	✓
Run on agent	✓
Initialize Agent	✓
Initialize Job	✓
Download Artifacts	✓
Deploy Azure App Service	✓
Post-deployment approval	✓

Deployment summary: Environment 1

Deployment Status	<b>SUCCEEDED</b>
Sub status	Succeeded
Trigger	Automated - After release creation
Requested by	Microsoft.VisualStudio.Services.ReleaseManagement
Requested for	Salvador Briones
Queued time	4 months ago
Completed time	4 months ago

Issues

Warnings (1)

- No artifacts are available in the build 83.

Figura 94 -Vista detalle del despliegue del proyecto

Podemos ver las tareas de despliegue en Azure, básicamente se trata de copiar los artefactos generados durante la construcción correcta y desplegarlos en el App Service creado en Azure (véase capítulo 4.3.)







# 6. CONCLUSIONES Y TRABAJO FUTURO

## 6.1. Conclusiones

Este trabajo desde un primer momento ha sido un gran reto personal ya que supone la representación y justificación del esfuerzo realizado durante todo el Grado. Al igual, supone un trabajo muy satisfactorio a nivel profesional dado que me doy cuenta de que la ingeniería del software es una rama que me fascina y me completa como persona.

He intentado que todos los conocimientos adquiridos durante estos años en el Grado queden reflejados en este trabajo. A su vez, he querido ser autodidacta y, añadir nuevas tecnologías y herramientas al proyecto para así comparar de primera mano las diferentes alternativas de soluciones para resolver un problema existente.

Los conocimientos adquiridos en el Grado han ido construyendo mi perfil profesional. No obstante, durante la realización de este trabajo, he tenido problemas en el desarrollo del proyecto, este hecho ha producido que yo mismo tenga que solucionarlo convirtiéndome poco a poco en mejor profesional.

Al principio de este trabajo se marcaron unos objetivos fijos que eran la realización de un proyecto software que nos facilitara las tareas de control, gestión y entrega de importaciones de una empresa de transporte. Estos objetivos se han cumplido, además de esto, se han añadido funciones que durante el desarrollo se han visto que era viables y dan un valor añadido a la calidad de este trabajo. La idea de crear este proyecto fue generada a partir de mi labor profesional en el mundo de la logística y vi de primera persona las necesidades que existían y pueden existir para realizar una gestión integral de importaciones. Creo que este sistema cumple todas las expectativas necesarias para que este puesto en funcionamiento como solución a las necesidades que se planearon desde un principio.

Este proyecto es flexible y puede ser aplicable a cualquier empresa de logística ya que, cuenta con todas las herramientas necesarias para que todos los procesos funcionen de manera adecuada y el servicio a ofrecer sea correcto.





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

Otro valor añadido que he intentado dar al proyecto es aplicar el uso de las tecnologías .Net en la totalidad del proyecto, ha sido un objetivo por cumplir y un reto personal. Me gustaría liberar parte del proyecto a aquellas personas que necesiten soporte en realizar algún tipo de proyecto o seguir la misma línea de éste. Dado que en un futuro me gustaría comercializar el proyecto, liberare sólo la parte de gestión de importaciones, es decir, la aplicación de escritorio. Solo me gustaría liberar esta parte dado que como trabajo futuro quiero plantear la posibilidad de seguir desarrollando el proyecto de la aplicación móvil.

Por otro lado, espero que este trabajo sea de ayuda o sirva de base para futuros trabajo a otras personas a nivel educativo o profesional, he intentado que sea lo más completo posible, pero, como bien sabemos, todo trabajo por el hombre es mejorable.

### 6.2. Trabajo futuro

A continuación, comentaremos el posible trabajo futuro a hacer en el proyecto:

- 🚚 **Página web.** Como trabajo futuro sería la adaptación a cualquier plataforma, esto lo podríamos lograr si transformamos el sistema gracias a la tecnología .NET o creáramos otra aplicación cliente de tipo web, de esta manera podría funcionar en cualquier plataforma o sistema operativo.
- 🚚 **Ubicación en tiempo real.** La ubicación actual del repartidor es relativa al uso de la aplicación. Si modificáramos y adaptáramos la aplicación podríamos ver el recorrido durante el reparto de un chofer.
- 🚚 **Desarrollo para iOS y Windows Phone.** Gracias a Xamarin tenemos la posibilidad de trasladar la aplicación a las plataformas móviles de Apple y Windows esto crearía una mayor adaptabilidad a las infraestructuras de una empresa.





## Bibliografía

- [1] **El comercio exterior en la economía española.** Periódico El País. Publicado en febrero 2015, consultado en noviembre 2017:  
[https://elpais.com/elpais/2015/02/18/media/1424285656\\_143881.html](https://elpais.com/elpais/2015/02/18/media/1424285656_143881.html)
- [2] **Las exportaciones españolas logran un récord de 254.530 millones de euros en 2016.** Periódico El País. Febrero 2017, consultado en octubre 2017:  
[https://economia.elpais.com/economia/2017/02/20/actualidad/1487587129\\_677591.html](https://economia.elpais.com/economia/2017/02/20/actualidad/1487587129_677591.html)
- [3] **Aumenta la subcontratación logística, aunque la inversión disminuye.** Diario on-line para el sector del transporte. Cadena de Suministro. Publicado en diciembre 2010, consultado en noviembre 2017: <http://www.cadenadesuministro.es/noticias/aumenta-la-subcontratacion-logistica-aunque-la-inversion-disminuye/>
- [4] **Android ya es el sistema operativo más usado del mundo.** Periódico El País. Publicado en abril 2017, consultado en noviembre 2017:  
[https://elpais.com/tecnologia/2017/04/04/actualidad/1491296467\\_396232.html](https://elpais.com/tecnologia/2017/04/04/actualidad/1491296467_396232.html)
- [5] **Manifiesto for Agile Software Development.** Kent Beck. Publicado en 2001, consultado en noviembre 2017: <http://agilemanifesto.org/>
- [6] **Microsoft Imagine – Universidad de Almería.** Microsoft. Consultado en noviembre 2017:  
[https://e5.onthehub.com/WebStore/ProductsByMajorVersionList.aspx?cmi\\_cs=1&cmi\\_mnuMain=bdba23cf-e05e-e011-971f-0030487d8897&ws=6ec1b2c8-749b-e011-969d-0030487d8897&vsro=8](https://e5.onthehub.com/WebStore/ProductsByMajorVersionList.aspx?cmi_cs=1&cmi_mnuMain=bdba23cf-e05e-e011-971f-0030487d8897&ws=6ec1b2c8-749b-e011-969d-0030487d8897&vsro=8)
- [7] **Microsoft Imagine para estudiantes.** Microsoft. Consultado en noviembre 2017:  
<https://azure.microsoft.com/es-es/pricing/member-offers/imaginel/>
- [8] **Planes de App Service.** Microsoft. Consultado en noviembre 2017:  
<https://azure.microsoft.com/es-es/pricing/details/app-service/plans/>
- [9] **Clase Stream. .Net Framework.** Microsoft. Consultado en noviembre 2017:  
[https://msdn.microsoft.com/es-es/library/system.io.stream\(v=vs.110\).aspx](https://msdn.microsoft.com/es-es/library/system.io.stream(v=vs.110).aspx)
- [10] **Programming WCF Services: Mastering WCF and the Azure AppFabric Service Bus.** O'Reilly. Juval Lowy. Septiembre 2010.
- [11] **iTextSharp Documentation. After Logic.** Consultado en noviembre 2017:  
<https://afterlogic.com/mailbee-net/docs-itextsharp/>
- [12] **Static Maps API. Google Maps API.** Consultado en noviembre 2017:  
<https://developers.google.com/maps/documentation/static-maps/?hl=es-419>





## SIGI: SISTEMA INTEGRAL PAR LA GESTIÓN DE IMPORTACIONES

[13] Platform Versions. Google Android Developers. Consultado en diciembre 2017:

<https://developer.android.com/about/dashboards/index.html#Platform>





El objetivo de este Trabajo Fin de Grado es dar una solución a aquellas empresas de logística que necesitan implantarse en posiciones geográficas y puede suponer un coste excesivo. Con la utilización de este sistema, estas empresas podrán subcontratar a otras empresas y tener una gestión íntegra de las importaciones sin que lleve un sobre coste en infraestructuras como instalaciones o flotas industriales.