Grado en Ingeniería Informática

Trabajo de Final de Grado

Desarrollo del módulo de testing de la aplicación del Suministro Inmediato de Información (SII) para el ERP Business Central

Autor:
Javier Díaz López

Supervisor:
Luis Rius Gumbau
Tutor académico:
María Cristina Campos Sancho

Fecha de lectura: 21 de Junio de 2023
Curso académico 2022/2023

Resumen

En el presente documento se describe el desarrollo de un proyecto para implementar el módulo de testing sobre la aplicación del Suministro Inmediato de Información (SII) en Business Central, el ERP de Microsoft.

El objetivo del proyecto es validar el correcto funcionamiento de la aplicación del SII con las futuras versiones de Business Central.

El proyecto ha sido desarrollado en el entorno de programación Visual Studio Code utilizando el lenguaje AL propiedad de Microsoft. Se ha realizado siguiendo las fases comunes que tienen este tipo de desarrollos utilizando el modelo en cascada.

Este documento se ha redactado durante la estancia en prácticas en la empresa Lãberit Sistemas S.L.

Palabras clave

Microsoft Business Central, Testing, IVA, Agencia tributaria, Codeunit, Lenguaje AL.

Keywords

Microsoft Business Central, Testing, VAT, Tax Agency, Codeunit, AL Language.

Índice general

Índice de figuras

Índice de cuadros

Listings

Capítulo 1

Introducción

1.1. Contexto y motivación del proyecto

En el presente texto encontramos descrito el proyecto realizado durante la estancia de prácticas en la empresa Lãberit Sistemas S.L.[10]

Lãberit es el resultado de la fusión de ocho empresas del grupo Alfatec. Dicha empresa se dedica al desarrollo de software y soluciones tecnológicas en los sectores de las autoridades portuarias, las administraciones públicas, la industria, la sanidad y la automoción. Se trata de una de las empresas líderes en el sector IT(Information Technology)[4] de nuestro país, y con intereses crecientes en Latinoamérica, África, Estados Unidos, Unión Europea y Reino Unido.

La sede se encuentra en Valencia, pero tiene sucursales en lugares como Castellón, Alicante, Murcia, Canarias, etc. E internacionalmente en lugares como Argentina, Colombia, Portugal, Chile. En Lãberit también se estructuran las empresas Beehive, Betalab (Geekshubs), Cablealia, Disid, Inlogiq, Kuara, Qwerty, que actúan en el mercado con marca propia.

Actualmente dispone de una plantilla que supera los 1000 profesionales, principalmente ingenieros técnicos y superiores en informática y telecomunicaciones. Además es reconocido como Partners Oficiales de grandes marcas multinacionales, especialmente Microsoft, Google y Atlassian.

Una de las principales áreas de negocio es la unidad BC Puertos llamada así dentro de la empresa, en la cual se dedican a dar soporte a las empresas denominadas autoridades portuarias, *"organismo público dependiente del Ministerio de Fomento a través de Puertos del Estado, al que corresponden, entre otras competencias, la prestación de los servicios generales, así como la gestión y control de los servicios portuarios para lograr que se desarrollen en condiciones óptimas de eficacia, economía, productividad y seguridad"*[7].

La sucursal de Castellón, donde nos encontramos, se encuentra dentro de la unidad de BC Puertos. Se trabaja con Microsoft Dynamics Business Central 365, *"solución de administración empresarial para pequeñas y medianas empresas que automatiza y optimiza los procesos empresariales y le ayuda a administrar su empresa"*[11] (Business Central), con el cual se puede gestionar la información de las empresas que lo contratan.

Business Central es un ERP (Enterprise Resource Planning) *"sistema que ayuda a automatizar y administrar los procesos empresariales de distintas áreas: finanzas, fabricación, venta al por menor, cadena de suministro, recursos humanos y operaciones"*[14] (ERP) de Microsoft. Dicho esto, esta unidad se encarga de implantar mejoras y dar soporte sobre el ERP de Business Central para las autoridades portuarias.

En la unidad de BC puertos todos los proyectos de desarrollo de software utilizan un proceso de evaluación y verificación para comprobar que cumple con los requisitos de software durante toda la vida del proyecto y permite detectar si hay una nueva versión de Business Central, que los cambios del producto no afectan al módulo. Esto es debido a que, según el estándar de Microsoft, al sacar una nueva versión de Business Central existen campos o funciones que pueden añadirse, modificarse o eliminarse y esto puede dar pie a errores o fallos en futuras mejoras en el desarrollo de la aplicación, es por ello que se hace uso de la ejecución de estos test al utilizar una nueva versión para identificar los cambios o errores en el desarrollo de código de una forma mucho más rápida y eficiente.

Recientemente, Lãberit se ha encontrado con el problema de que las empresas que contratan sus servicios o publican un contrato de oferta pública en un pliego de condiciones (documento que presenta una empresa, en el que se redactan los requisitos a cumplir y la empresa que desea adquirir el contrato debe responder demostrando cómo va a realizar todo lo que propone el contrato sin exceder el presupuesto propuesto por la empresa). Los clientes solicitan de una persona experta en testing para que pueda testear que todo el código que se va desarrollando en el proyecto funciona de manera correcta, ya que, una vez finalizado el proyecto, el cliente no quiere encontrarse con fallos en su aplicación o cambios en migraciones de futuras versiones.

Actualmente existe un proyecto en desarrollo para la aplicación del Suministro Inmediato de Información del IVA (SII) donde, cualquier mejora que se hace en el módulo de desarrollo de la aplicación lleva acompañado un módulo de testing para verificar que las posibles nuevas versiones no afecten a esa mejora.

Dicho esto, la motivación final de este proyecto surge de la necesidad de realizar la verificación para comprobar el correcto funcionamiento de la aplicación del SII y que sirva para la detección de cambios o errores en futuras versiones de Business Central.

En cuanto a la aplicación del SII se trata de un servicio llamado Comunicación del Suministro Inmediato de Información del IVA *"un cambio del sistema de gestión actual del IVA que lleva 30 años funcionando, pues se pasa a un nuevo sistema de llevanza de los libros de registro del Impuesto sobre el Valor Añadido a través de la Sede Electrónica de la Agencia Estatal de Administración Tributaria (AEAT), mediante el suministro cuasi inmediato de los registros de facturación"*[3] (SII). Es decir, es un servicio que hace de intermediario para enviar la información sobre el IVA de las empresas a la agencia tributaria.

Respecto a los beneficios que aporta realizar este proyecto, como ya hemos mencionado de forma breve anteriormente, tenemos los siguientes puntos:

- Ahorro de tiempo y dinero: cuando se implantan nuevas versiones de Business Central a la hora de buscar qué modificaciones afectan a los módulos desarrollados para los clientes.
- Desarrollo más rápido: la detección de cambios necesarios a la hora de implantarlos en las nuevas versiones de forma más eficiente. En caso de no tener estos módulos de testing acompañando a cada mejora, cuando se implanta una nueva versión se tendría que revisar cómo afecta esta nueva versión a cada uno de los módulos que se han desarrollado, a cada campo y función.
- Rediseño rápido: permite detectar dónde hay cambios para rediseñar las mejoras de acuerdo a las nuevas versiones.

1.2. Objetivo y alcance del proyecto

El objetivo del proyecto es desarrollar el módulo de testing en la aplicación del SII para validar el correcto funcionamiento de este y las futuras versiones de Business Central. Para ello antes hay que realizar un estudio interno de la funcionalidad de Business Central y, seguidamente, un exhaustivo estudio del funcionamiento de la propia aplicación del SII y el código desarrollado en la aplicación. Las tareas a realizar en el desarrollo del proyecto son las siguientes:

1. Analizar el funcionamiento del ERP de Business Central en el área que compete a la aplicación del SII.
2. Estudiar el funcionamiento de la aplicación del SII.
3. Desarrollar tests para que el módulo funcione correctamente cuando se implante en nuevas versiones.

Realizando esta serie de tareas, anteriormente mencionadas, conseguimos el objetivo final que es automatizar test en procesos de integración continua para que, en futuras actualizaciones de Business Central, el desarrollador pueda realizar, de una manera más sencilla, la actualización de la aplicación detectando las funciones que han añadido campos o qué variables han cambiado respecto la versión anterior.

1.3. Objetivo y alcance del producto

El objetivo del producto nos permite validar el correcto funcionamiento del código desarrollado en el módulo de desarrollo del SII para que, en futuras versiones de Business Central, cumpliendo los requisitos tanto de este como de Microsoft y profundizar en materia del testing mejorando esta herramienta en los proyectos existentes dando robustez a las soluciones. Cuando surjan nuevas versiones de Business Central, la aplicación del SII solo tendrá que modificar aquellos campos o funciones en las que se detecten cambios una vez ejecutado el test.

Con esto conseguimos un desarrollo de código más eficiente y seguro para que las actualizaciones de código en Business Central, que puedan surgir en un futuro, sean más sencillas de realizar. El objetivo del producto y el alcance funcional en este caso coinciden.

En el producto se trabaja en varias áreas del testing, ya sea desarrollando varios tests desde cero, actualizando una función de test y testeando un servicio. Todo esto comprobando que las funcionalidades y requisitos que debe poseer el producto son las correctas.

1. Testear que el servicio web de la AEAT no está caído.

2. Testear todos los campos para una factura concreta.

3. Testear que el código del país a enviar, es correcto referente a las tablas que nos proporciona la AEAT.

4. Testear pagos de un tipo de factura concreta.

5. Testear cobros de un tipo de factura concreta.

1.3.1. Alcance organizativo

Respecto un punto de vista organizativo, el producto desarrollado es para uso comercial de la empresa Lãberit Sistemas S.L, más concretamente, la unidad de BC Puertos (Autoridades Portuarias), ya que son los encargados de este desarrollo. Es decir, este producto va a ser utilizado por los desarrolladores de Lãberit que vayan a utilizar este módulo.

1.3.2. Alcance informático

Respecto un punto de vista informático, podemos decir que, el producto que se está realizando tiene un alcance muy delimitado, es decir, el alcance informático del proyecto es que el módulo del SII funcione correctamente cuando se implanten nuevas versiones conectándose con el propio Business Central y realizando las comunicaciones pertinentes con la AEAT.

1.4. Descripción detallada del desarrollo del proyecto

El desarrollo del proyecto consiste en la realización de un módulo de testing para la aplicación del SII, con este módulo detectaremos si ha habido cambios en la aplicación respecto a la versión anterior de Business Central.

Para ello, se van a realizar una serie de desarrollos probando la verificación de datos sobre diferentes facturas siguiendo una determinada metodología. En el siguiente apartado se detallan las herramientas utilizadas.

El proyecto, para el desarrollo de este módulo de testing, ha requerido llevar a cabo un estudio exhaustivo tanto de Business Central como de la aplicación del SII donde, en la fase de formación, ha sido un trabajo complejo, ya que el testing es un proceso novedoso y poco desarrollado actualmente por las empresas.

1.4.1. Herramientas y tecnologías utilizadas

El proyecto consiste, por tanto, en el desarrollo de un módulo de testing que usará las siguientes tecnologías:

- Microsoft Dynamics 365 Business Central, ERP con el que trabaja la empresa.
- Microsoft Projects, para la planificación.
- Visual Studio Code, como entorno de programación.
- Lenguaje AL, para la programación.
- Azure DevOps, para almacenar los proyectos unido al Git para actualizarlo diariamente.
- Jira, para estimar los tiempos usados en cada fase del proyecto.
- MS Teams, para la comunicación con el resto de compañeros.

Microsoft Dynamics 365 Business Central [11] es un software propiedad de Microsoft para la planificación de recursos empresariales. Antiguamente, conocido en versión de escritorio como Microsoft Dynamics NAV.

Microsoft Projects[12] es una herramienta de Microsoft para la administración y planificación de proyectos. Software utilizado para la realización del diagrama de Gantt para tener una organización de las tareas que vamos a realizar del proyecto en la estancia en prácticas.

Visual Studio Code[16] es el entorno de desarrollo que usaremos para modificar el código estándar de Microsoft Dynamics 365 Business Central. Proporcionado por Microsoft para el desarrollo de código, en nuestro caso usando lenguaje de programación AL, también propiedad de Microsoft, se trata de un entorno muy completo, ya que acepta extensiones para docker y depuración de código bastante útiles en la empresa.

Azure DevOps[6] es una herramienta, propiedad de Microsoft, con la que se puede llevar a cabo la gestión de proyectos en la nube junto con Git para el control de versiones en Visual Studio Code con una extensión propia.

Jira[5] software empleado para estimar los tiempos usados en cada fase del proyecto, con este software se van actualizando las incidencias o avances que vamos realizando, día a día, sobre el proyecto que estamos desarrollando.

MS Teams[13] es una aplicación, propiedad de Microsoft, en la que puedes, principalmente, comunicarte tanto por chat como por videoconferencia con el equipo de trabajo. Puede que haya gente trabajando desde casa o desde otra sucursal que no sea en la que nosotros estamos realizando el proyecto y también se transfieren archivos. En nuestra unidad se realizan las reuniones semanales con el resto del equipo BC Puertos que se encuentra en Valencia.

1.5. Estructura de la memoria

En este apartado encontramos descrito cómo se estructura la memoria realizada. Se compone de 5 capítulos divididos en subapartados donde, en cada uno, se detalla con exactitud qué se ha realizado durante la estancia en prácticas en la empresa.

En el Capítulo 1, Introducción, es el capítulo actual en el que se relata, brevemente, como surge el proyecto y como se va a desarrollar con un enfoque técnico.

En el Capítulo 2, Planificación del proyecto, podemos encontrar la definición de la metodología que se usará a lo largo del proyecto, teniendo en cuenta la planificación inicial podemos observar la comparación con la planificación real. Finalmente, una breve explicación del proyecto que vamos a desarrollar, el seguimiento y los costes que este conlleva.

En el Capítulo 3, Análisis y diseño del sistema, se describe el proceso que hemos llevado a cabo para realizar un análisis y diseño minucioso definiendo qué herramientas vamos a usar y cómo las vamos a usar para el desarrollo. Finalmente, la descripción de requisitos y el respectivo análisis de cada uno de ellos.

En el Capítulo 4, Implementación y pruebas, se encuentra, detalladamente, los procedimientos que se han ido realizando para la implementación de código y sus respectivas pruebas para validar el funcionamiento del módulo de testing para la aplicación del SII.

En el Capítulo 5, Conclusiones, se trata del capítulo final, en el que se hace una valoración general del proyecto y la estancia en prácticas aportando una opinión personal del trayecto realizado y aprendido a lo largo de los meses.

Capítulo 2

Planificación del proyecto

2.1. Metodología

Para la realización del proyecto se decide utilizar una metodología predictiva, más concretamente conocida como modelo en cascada[15], este está ordenado por una serie de fases que van una detrás de otra a medida que se van completando a lo largo de la duración del proyecto.

El proyecto está compuesto por las siguientes fases:

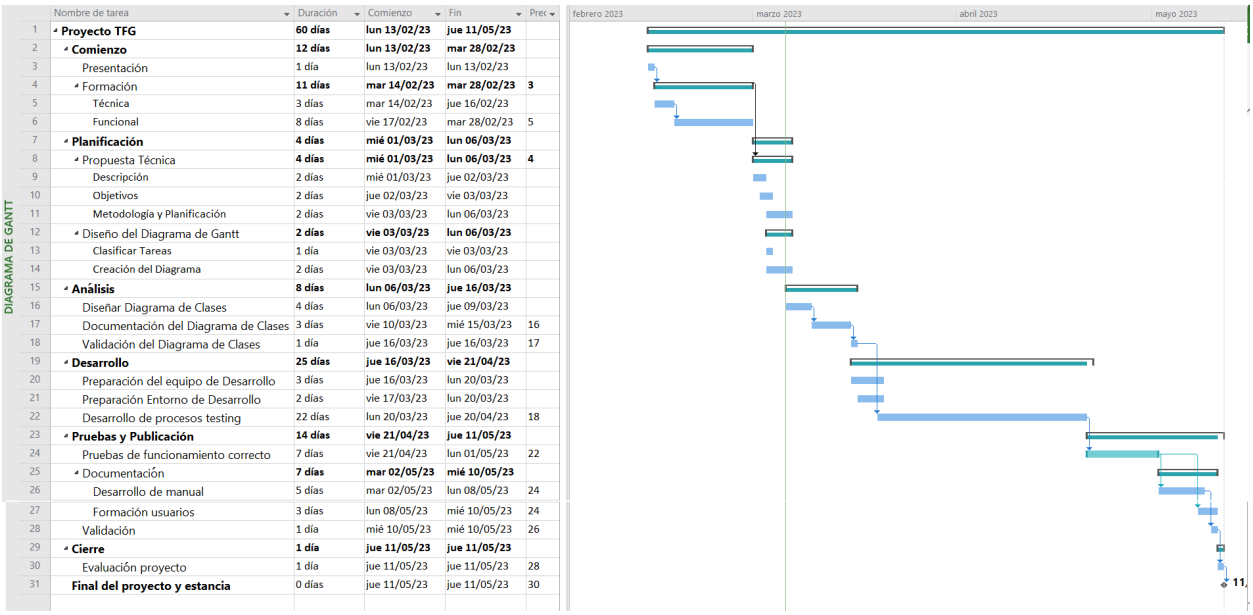
- Planificación.
- Análisis.
- Desarrollo.
- Validación.
- Documentación.

Además se planifica un período de formación inicial puesto que el desarrollo y las herramientas usadas en el ERP de Business Central lo requiere.

2.2. Planificación temporal del proyecto

Como hemos identificado anteriormente, respecto a la metodología que hemos escogido en el proyecto, realizamos un Diagrama de Gantt[9] en el que definimos las fases del proyecto y se estima su duración.
Esto se lleva a cabo con Microsoft Project, una herramienta especializada para la gestión de proyectos. Desarrollaremos el diagrama teniendo en cuenta que la duración de nuestra estancia en prácticas son 300 horas y que realizamos una media diaria de 5 horas, es decir, 25 horas semanales.

En la Figura 2.1 se muestra la planificación inicial.



	Nombre de tarea	Duración	Comienzo	Fin	Pred
1	**Proyecto TFG**	**60 días**	**lun 13/02/23**	**jue 11/05/23**	
2	**Comienzo**	**12 días**	**lun 13/02/23**	**mar 28/02/23**	
3	Presentación	1 día	lun 13/02/23	lun 13/02/23	
4	**Formación**	**11 días**	**mar 14/02/23**	**mar 28/02/23**	**3**
5	Técnica	3 días	mar 14/02/23	jue 16/02/23	
6	Funcional	8 días	vie 17/02/23	mar 28/02/23	5
7	**Planificación**	**4 días**	**mié 01/03/23**	**lun 06/03/23**	
8	Propuesta Técnica	**4 días**	**mié 01/03/23**	**lun 06/03/23**	**4**
9	Descripción	2 días	mié 01/03/23	jue 02/03/23	
10	Objetivos	2 días	jue 02/03/23	vie 03/03/23	
11	Metodología y Planificación	2 días	vie 03/03/23	lun 06/03/23	
12	Diseño del Diagrama de Gantt	**2 días**	**vie 03/03/23**	**lun 06/03/23**	
13	Clasificar Tareas	1 día	vie 03/03/23	vie 03/03/23	
14	Creación del Diagrama	2 días	vie 03/03/23	lun 06/03/23	
15	**Análisis**	**8 días**	**lun 06/03/23**	**jue 16/03/23**	
16	Diseñar Diagrama de Clases	4 días	lun 06/03/23	jue 09/03/23	
17	Documentación del Diagrama de Clases	3 días	vie 10/03/23	mié 15/03/23	16
18	Validación del Diagrama de Clases	1 día	jue 16/03/23	jue 16/03/23	17
19	**Desarrollo**	**25 días**	**jue 16/03/23**	**vie 21/04/23**	
20	Preparación del equipo de Desarrollo	3 días	jue 16/03/23	lun 20/03/23	
21	Preparación Entorno de Desarrollo	2 días	vie 17/03/23	lun 20/03/23	
22	Desarrollo de procesos testing	22 días	lun 20/03/23	jue 20/04/23	18
23	**Pruebas y Publicación**	**14 días**	**vie 21/04/23**	**jue 11/05/23**	
24	Pruebas de funcionamiento correcto	7 días	vie 21/04/23	lun 01/05/23	22
25	**Documentación**	**7 días**	**mar 02/05/23**	**mié 10/05/23**	
26	Desarrollo de manual	5 días	mar 02/05/23	lun 08/05/23	24
27	Formación usuarios	3 días	lun 08/05/23	mié 10/05/23	24
28	Validación	1 día	mié 10/05/23	mié 10/05/23	26
29	**Cierre**	**1 día**	**jue 11/05/23**	**jue 11/05/23**	
30	Evaluación proyecto	1 día	jue 11/05/23	jue 11/05/23	28
31	**Final del proyecto y estancia**	0 días	jue 11/05/23	jue 11/05/23	30

Figura 2.1: Diagrama de Gantt Inicial.

Se pensó, en un principio, realizar una planificación como la vista en las asignaturas a lo largo de la carrera, en él se sigue un proyecto estandar de desarrollo de prácticas y de proyectos.

En el apartado 2.3. Seguimiento del proyecto, se muestra como queda la figura 2.2 de la planificación del desarrollo final.

2.3. Seguimiento del proyecto

El seguimiento de este proyecto se ha controlado semanalmente, a través de reuniones con el supervisor, verificando y validando que se iba cumpliendo la planificación con reuniones mensuales para hacer un balance de todo lo que se avanza y con los informes quincenales que se le envían a la tutora de prácticas.

Los informes quincenales también han sido una parte importante del seguimiento del proyecto, en ellos se iban haciendo pequeños resúmenes de lo que se iba realizando cada quincena. Estos informes son enviados a la tutora del proyecto para que se tenga un control, a largo plazo, de lo que se está realizando de manera progresiva.

Las reuniones mensuales con el supervisor para realizar un balance de todo lo aprendido y realizado en relación al mes anterior sirven para valorar si el proyecto va progresando adecuadamente y el supervisor te hace las recomendaciones para mejorar lo que se ha llevado a cabo.

Además de todo lo mencionado anteriormente, al final de cada fase de desarrollo se verifica y prueba con el supervisor para ver que funcionaba de la manera deseada para poder pasar a la siguiente fase.

Como hemos mencionado en el apartado anterior, algunas partes han cambiado respecto a la planificación inicial, esto es debido a que, primeramente, el planteamiento inicial del proyecto tenía un enfoque bastante complejo que costó mucho delimitar y encauzar hacia los objetivos finales. Se planteó ir poco a poco comenzando por un sencillo test para luego ir ampliando la dificultad de los tests de todo el módulo.

En la figura 2.2 se muestran los cambios respecto al planteamiento inicial de las tareas.

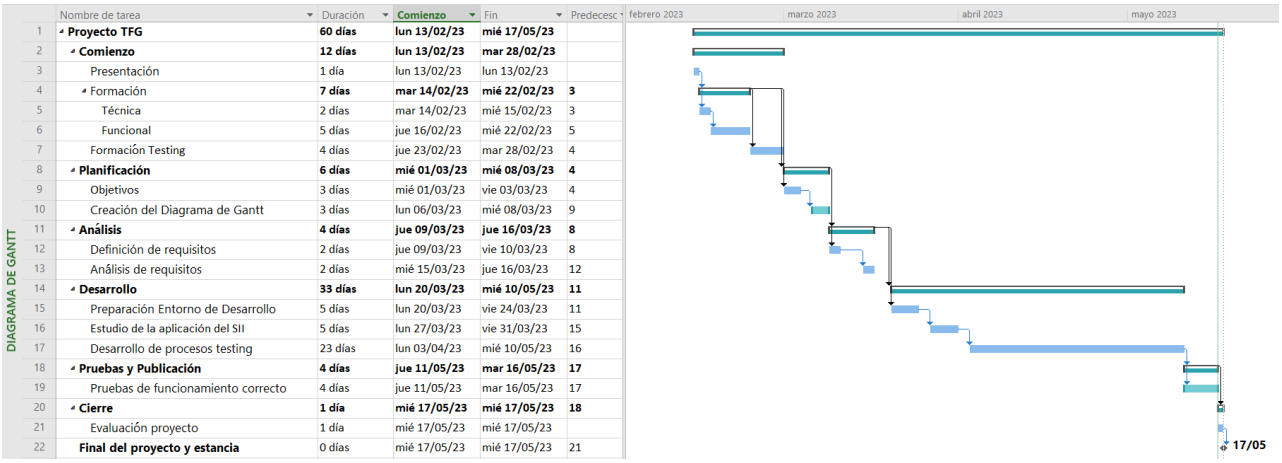


Figura 2.2: Diagrama de Gantt de seguimiento.

A continuación se detallan cada una de las fases del seguimiento.

1. **Comienzo**

   1.1 **Presentación:** El primer día de trabajo, en el que se conoce a los compañeros de la unidad y el supervisor, se tiene una reunión con recursos humanos en la que te explican la historia y el funcionamiento de la empresa.

   1.2 **Formación:** Primeras semanas de trabajo donde te enseñan las herramientas y el entorno con el que trabaja la empresa.

      1.2.1 **Funcional:** La primera toma de contacto con Business Central en la que te enseñan la interfaz con la que vas a trabajar y a realizar facturas de compra y de venta.

      1.2.2 **Técnica:** Visionado de videotutoriales y realización de ejercicios generales en los que te enseñan a cómo crear tablas, páginas y campos para entender el funcionamiento de la programación en el lenguaje AL.

   1.3 **Formación Testing:** Masterclass de testing que se centra en la estructura de tests y como realizar las futuras pruebas. Además de una serie de ejercicios en los que tienes que investigar navegando por el resto de tablas y páginas del estándar para entender cómo se comunican entre ellas.

2. **Planificación**

   2.1 **Objetivos:** Identificación de los objetivos del proyecto a desarrollar.

   2.2 **Creación del Diagrama de Gantt:** Esquema general del proyecto en el que se enumeran las actividades o tareas a realizar durante la estancia en la empresa.

3. **Análisis**

   3.1 **Definición de requisitos:** Documento en el cuál se identifican los requisitos que deberá tener nuestro proyecto final.

   3.2 **Análisis de requisitos:** Estudio de los requisitos anteriormente identificados.

4. **Desarrollo**

   4.1 **Preparación entorno de desarrollo:** Instalación de todo el software necesario para llevar a cabo el correcto desarrollo del módulo de testing, como viene a ser el propio ERP, el entorno de programación y otras herramientas necesarias para ello.

   4.2 **Estudio de la aplicación del SII:** Exhaustivo análisis del código de la aplicación y su funcionamiento.

   4.3 **Desarrollo de procesos testing:** Desarrollo de la extensión de la aplicación de test definidos anteriormente en el análisis. El grueso del proyecto ya que aquí vamos a invertir la mayor cantidad de tiempo.

5. **Pruebas y Publicación**

   5.1 **Probar Desarrollo:** Pruebas realizadas en la interfaz de Business Central para corroborar que el desarrollo del proyecto es el correcto.

6. **Cierre**

   6.1 **Evaluación del proyecto:** Una vez finalizado el proyecto se realiza la evaluación del mismo donde se prueba el correcto desarrollo de este y una evaluación del rendimiento del alumno por parte del supervisor y jefe de unidad.

7. **Fin de proyecto**

2.4. Gestión de recursos humanos

En cuanto a la gestión de recursos humanos debemos definir el personal implicado en el proyecto con sus respectivos roles. Definimos así los siguientes roles:

- Jefe de proyecto: persona encargada de dirigir el proyecto y delegar las tareas al resto de personal de la empresa que se van a realizar a lo largo del proyecto.
- Supervisor: persona encargada de controlar el seguimiento del programador a lo largo del proyecto y guiarle en aquellas tareas que más le dificulten.
- Programador junior: persona encargada de la programación del proyecto y que debe realizar las tareas encargadas por el supervisor.

En el cuadro 2.1 vemos reflejado un resumen con los nombres de las personas encargadas y sus respectivos roles:

Roles	**Nombre del Personal**
Jefe de proyecto:	Luis Rius Gumbau
Supervisor:	Luis Rius Gumbau
Programador junior:	Javier Díaz López

Cuadro 2.1: Roles del proyecto

2.5. Costes

En la siguiente sección hablaremos de la estimación de costes y recursos que hemos utilizado en el desarrollo del proyecto.

Indicamos todos los recursos tecnológicos en cuanto a hardware y licencias necesarias para llevar a cabo el proyecto, a continuación los recursos humanos que son necesarios para entender lo que le costaría a la empresa estos servicios. Con todo esto hay que tener en cuenta que el total de horas de estancia en prácticas es de 300 horas, es decir la valoración se hará en ese tiempo.

Comenzamos describiendo los recursos hardware que nos ha proporcionado para trabajar de una manera más eficiente y cómoda:

- Equipo: Ordenador portátil DELL Vostro
  - Procesador: Intel(R) Core(TM) i5-7200U CPU
  - RAM: 16,0 GB
  - Disco: 250 GB
  - Gráfica: Intel(R) HD Graphics 620
- Monitor: AOC 236LM00014
- Ratón: Logitech M185
- Teclado: Logitech K120

Como no trabajamos con una aplicación propia hay que calcular también el precio de las licencias que tenemos contratadas. En este caso, la empresa Lãberit Sistemas S.L tiene contratada la licencia de MS Dynamics 365 Business Central de la que nosotros hacemos uso diario. Existen dos tipos de licencias: Dynamics 365 Business Central Essentials valorada en 65,50 euros y la licencia Dynamics 365 Business Central Premium con un valor de 93,60 euros. En nuestro caso hacemos uso de la versión premium.

Finalmente, respecto a los recursos humanos debemos de calcular los costes del programador junior que somos en este caso nosotros con un coste 300 euros al mes y del supervisor en el cual estimamos un coste de unos 18 euros la hora.

A continuación, con la información recabada, obtenemos una estimación del coste total que generamos en el proyecto.
Véase en los cuadros 2.2 y 2.3.

Costes proyecto			
Recursos	Cantidad	Coste	Subtotal
Portátil DELL	1	650 €	(650/(4*12))*3 = 40,65€
Monitor AOC	1	99 €	(99/(4*12))*3 = 6,20 €
Teclado Logitech	1	20 €	(20/(4*12))*3 = 1,25€
Ratón	1	10 €	(10/(4*12))*3 = 0,65€
Total:			**48,75 €**

Cuadro 2.2: Tabla de hardware informático

Costes proyecto			
Recursos	Horas	Mensual	Subtotal
Licencia BC	300	93,60 €	280,80€
Programador Junior	300	300 €	900 €
Supervisor programador	300	1.500 €	4.500€
Total:			**5.680,80 €**

Cuadro 2.3: Licencias y personal

Con los datos obtenidos en las tablas anteriores se estima que la cantidad total del proyecto supondrá a la empresa un coste de **5.729,55 €**.

2.6. Riesgos

En esta sección estableceremos los riesgos que nos pueden surgir durante el desarrollo del proyecto. Se trata de un punto crítico durante la realización de este debido a que cualquier riesgo que se active podría suponer un impedimento para el correcto progreso del proyecto.

Los riesgos que hemos identificado para este proyecto son los siguientes:

- Responsable del proyecto no está presente.
- Falta de conocimietos de programación.
- Falta de conocimiento en el manejo de un ERP.
- Cambios respecto a la definición de requisitos iniciales.

Una vez identificados todos los riesgos vamos a proceder a realizar el respectivo análisis de cada uno de ellos.
Véase en los cuadros 2.4, 2.5, 2.6 y 2.7.

Análisis de riesgos	
Riesgo 01:	Responsable de proyecto no está presente.
	Para paliar este riesgo se tienen reuniones quincenales con el supervisor para tener un seguimiento más correcto a lo largo del proyecto.

Cuadro 2.4: Análisis del riesgo 01

Análisis de riesgos	
Riesgo 02:	Falta de conocimientos de programación.
	Para minimizar los riesgos, el estudiante debe formarse de manera correcta y seguir los consejos del supervisor realizando la formación en el lenguaje AL y funcionamiento de Business Central.

Cuadro 2.5: Análisis del riesgo 02

Análisis de riesgos	
Riesgo 03:	Falta de conocimiento en el manejo de un ERP.
	Para paliar este riesgo se proporciona formación y acceso a videotutoriales y webs para el aumento de conocimientos.

Cuadro 2.6: Análisis del riesgo 03

Análisis de riesgos	
Riesgo 04:	Cambios respecto a la definición de requisitos iniciales.
	Para minimizar este riesgo se definen los bloques de requisitos y se implementan por orden de dificultad, del bloque más sencillo al más complejo.

Cuadro 2.7: Análisis del riesgo 04

Capítulo 3

Análisis y diseño del módulo de testing para la aplicación del SII

En este capítulo vamos a definir el análisis y diseño del módulo de testing para la aplicación del SII. Para ello primero empezaremos definiendo los requisitos del proyecto y después continuaremos con una exhaustiva definición de los datos de cada factura. En el apartado de diseño partimos del ejemplo de un test para explicar el diseño software y la interfaz que usaremos para su ejecución.

3.1. Definición de requisitos

La definición de requisitos consiste en especificar las condiciones que tiene que cumplir el producto.

Como requisito genérico tenemos que comprobar dónde ha habido cambios o dónde afecta una nueva versión de Business Central al módulo del SII. Por tanto, hay que conocer el módulo del SII, su funcionalidad, qué tipos de datos gasta y hay que verificar que siguen siendo compatibles con la nueva versión de Business Central y, en caso de incompatibilidad, mostrar dónde hay errores para que sean corregidos.

Como se ha comentado anteriormente, el módulo del SII envía el IVA de las facturas generadas por Business Central al sistema de la agencia tributaria. En este apartado vamos a entrar en detalle de cuáles son los datos de las facturas que se envian de Business Central a la agencia tributaria mediante el SII.

Las facturas generadas en Business Central se clasifica mediante tres campos: **"Tipo libro registro"**, **"Clave Tipo Factura"** y **"Clasificación SII"**. Estos son jerárquicos, por tanto, existen muchas combinaciones posibles.

Cada factura que genera Business Central se clasifica con un campo llamado **"Tipo libro registro"** según el tipo de factura que vayamos a comunicar a la agencia tributaria. Los dos últimos tipos son facturas especiales RECC (régimen especial criterio de caja), en el siguiente capítulo entraremos más en detalle sobre este tipo de facturas. Los tipos de factura que se clasifican mediante el campo **"Tipo libro registro"** se muestran en la figura 3.1.

Facturas según Tipo libro registro
Facturas Emitidas
Facturas Recibidas
Determinadas Operaciones Intracomunitarias
Bienes inversión
Cobros y Pagos en Metálico
Cobros Facturas Emitidas(RECC).
Pagos Facturas Recibidas(RECC).

Figura 3.1: Tipos de factura según libro registro.

Además, dentro de cada **"Tipo libro registro"**, las facturas tienen distintos subtipos, estos se identifican mediante el campo denominado **"Clave Tipo Factura"** que contiene los valores que se muestran en la figura 3.2.

Clave	Descripción clave de tipo de facturación	
F1	Factura	
F2	Factura Simplificada	
F3	Factura Emitida En Sustitución De Facturas Simplificadas Facturadas Y Declaradas Con Anterioridad	
F4	Asiento Resumen De Facturas	
F5	Importaciones (DUA)	
F6	Otros Justificantes Contables Y Documentos Justificativos Del Derecho A La Deducción	
R1	Factura Rectificativa	Error fundado en derecho y Art. 80 Uno y Dos LIVA
R2		Art 80 Tres LIVA - concurso
R3		Art 80 Cuatro LIVA - deuda incobrable
R4		Resto
R5		Factura rectificativa simplificada

Figura 3.2: Tipos de factura según clave.

Esto a su vez, como Business Central realiza la tarea de comunicar con el SII las facturas con IVA, estas facturas se clasifican dentro de un campo llamado **"Clasificación SII"** que contiene los valores de S1, S2, E1 hasta E6, No Sujeta por localización y No Sujeta Art.7-14. Estos se muestran en la figura 3.3.

Descripción Tipo Desglose IVA (Clasificación SII)	
Sujeta/ No exenta	S1 Sujeta – No Exenta;
	S2 Sujeta – No Exenta - Inv. Suj. Pasivo.
Causa de Exención operación Sujeta y Exentas	E1 Exenta por el artículo 20
	E2 Exenta por el artículo 21
	E3 Exenta por el artículo 22
	E4 Exenta por el artículo 24
	E5 Exenta por el artículo 25
	E6 Exenta por Otros.
No Sujeta Por Reglas De Localización	
No Sujeto Art.7-14 y Otros	

Figura 3.3: Tipos según la clasificación del SII.

Es decir, cada factura a parte de estar clasificada dentro de un tipo en el campo **"Tipo libro registro"** lleva consigo un código, tanto para el campo **"Clave Tipo Factura"** como el campo **"Clasificación SII"**, estos dos últimos pueden combinarse de todas las formas posibles. Todos los tipos vienen definidos por el estándar de la AEAT[1] para su respectiva comunicación.

Como existen tantas combinaciones de posibles tipos de facturas ya que son tres niveles y todas las combinaciones posibles, hemos comenzado por aquellas más habituales.

Debido a que existen muchos tipos de validaciones para todas las combinaciones y no hay tiempo suficiente en la estancia para la realización de todas ellas, en los requisitos aparecen los tests concretos que vamos a realizar para las facturas que son más empleadas por los clientes.

Por lo tanto, en el proyecto habrá que desarrollar tests para facturas de diferentes tipos, a continuación se describen los requisitos que deben cumplir cada uno estos:

- **RQ01:** validar que el servicio web de la AEAT no está caído para poder comunicar con el SII.
- **RQ02:** validar todos los campos obligatorios dentro de lo establecido por el departamento de informática de la agencia tributaria tiene un valor, para el **"Tipo libro registro"**: *Facturas Emitidas* y el subtipo de factura concreto *F2S1.*
- **RQ03:** validar que el país introducido existe y se encuentra dentro de la Unión Europea que es lo establecido por el departamento informático de la AEAT, para el **"Tipo libro registro"**: *Facturas Emitidas* y el subtipo de factura concreto *F2S1.*
- **RQ04:** validar todos los campos para cada tipo diferente de factura que puede realizarse según el **"Tipo libro registro"**: *Pagos Facturas Recibidas(RECC)* y todos sus subtipos. Como se ha mencionado antes, no es posible en el tiempo de estancia realizar para todos los subtipos, es por ello que hemos escogido los siguientes: *F1S1, F1S2 y F1E1.*
- **RQ05:** validar todos los campos para cada tipo diferente de factura que puede realizarse según el **"Tipo libro registro"**: *Cobros Facturas Emitidas(RECC)* y todos su subtipos. En este caso, las facturas realizadas serán las que lleven por código *F2S1, F1S2, F1NSArt7 y R1S1.*

3.2. Definición y análisis de los datos a validar por tipo de factura

Teniendo en cuenta los cinco requisitos que se han mencionado, vamos a analizar para cada uno de los tipos de factura qué datos se requiere validar.

Los requisitos de campos vienen dados por el formato de los documentos establecidos por el departamento de informática tributaria (documento entregado por la empresa). Para ello, por parte de la empresa, se nos facilitó un documento de la AEAT en el que se describen detalladamente los requisitos de cada campo. A continuación, cada requisito irá asociado a una imagen en la que se describen estos campos.

Primeramente tenemos la figura en la que se describe la leyenda para los futuros campos que vamos a analizar:

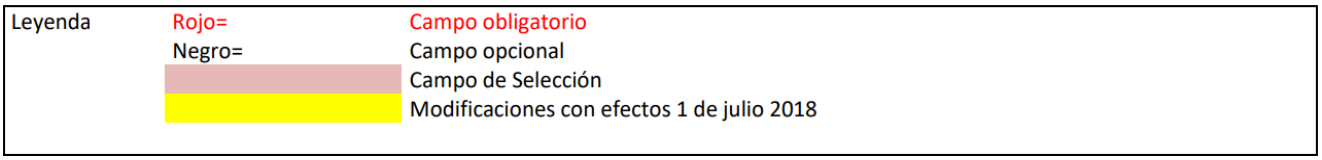


Figura 3.4: Leyenda de los campos.

En ella se nos hace saber que solo habrá que validar aquellos campos que estén en color rojo, que sean nodos hoja (no contengan ningún hijo), es decir aquellos que a su derecha no contengan más campos.

El **RQ01** es un poco distinto a todos, ya que en este vamos a testear que el servicio web de la AEAT no está caído. Esto se puede comprobar viendo el comportamiento de una factura al comunicar con el SII el IVA de la factura y ver que se ha modificado respecto a antes de la comunicación. En este caso, vemos que los campos "Comunication ID" y "Status" cambian después de la comunicación.

Respecto al **RQ02** tenemos las figuras 3.5, 3.6 y 3.7 que muestran todos los campos del **"Tipo libro registro"**: *Facturas Emitidas* y, como hemos mencionado anteriormente, los campos que aparezcan en rojo y no tengan hijos serán los que se deben validar.

BLOQUE	DATOS/ AGRUPACIÓN	DATOS/ AGRUPACIÓN	DATOS/ AGRUPACIÓN	DATOS/ AGRUPACIÓN	DATOS/ AGRUPACIÓN	DATOS/ AGRUPACIÓN	DATOS			DESCRIPCIÓN	FORMATO LONGITUD LISTA
Cabecera	IDVersionSii									Identificación de la versión del esquema utilizado para el intercambio de información	Alfanumérico(3) L19
	Titular	NombreRazon								Nombre-razón social del Titular del libro de registro de facturas expedidas	Alfanumérico(120)
		NIFRepresentante								NIF del representante del titular del libro de registro	FormatoNIF(9)
		NIF								NIF asociado al titular del libro de registro	FormatoNIF(9)
	TipoComunicacion									Tipo de operación (alta, modificación)	Alfanumérico(2) L0
RegistroLRfacturasEmitidas	PeriodoLiquidacion	Ejercicio								Ejercicio	Numérico(4)
		Periodo								Periodo Liquidación	Alfanumérico(2) L1
	IDFactura	IDEmisorFactura	NIF							NIF asociado al emisor de la factura.	FormatoNIF(9)
		NumSerieFacturaEmisor								Nº Serie+Nº Factura que identifica a la factura emitida (en su caso primera factura del asiento resumen)	Alfanumérico(60)
		NumSerieFacturaEmisorResumenFin								Nº Serie+Nº Factura que identifica a la ultima factura cuando el Tipo de Factura es un asiento resumen de facturas	Alfanumérico(60)
		FechaExpedicionFacturaEmisor								Fecha de expedición de la factura	Fecha(dd-mm-yyyy)
	FacturaExpedida	TipoFactura								Especificación del tipo de factura: factura completa, factura simplificada, factura emitida en sustitución de facturas simplificadas, asiento resumen o factura rectificativa.	Alfanumérico(2) L2_EMI
		TipoRectificativa								Campo que identifica si el tipo de factura rectificativa es por sustitución o por diferencia	Alfanumérico(1) L5
		FacturasAgrupadas	IDFacturaAgrupada	NumSerieFacturaEmisor						Nº Serie+Nº Factura que identifica a la factura emitida	Alfanumérico(60)
				FechaExpedicionFacturaEmisor						Fecha de expedición de la factura	Fecha(dd-mm-yyyy)
		FacturasRectificadas	IDFacturaRectificada	NumSerieFacturaEmisor						Nº Serie+Nº Factura que identifica a la factura emitida	Alfanumérico(60)
				FechaExpedicionFacturaEmisor						Fecha de expedición de la factura	Fecha(dd-mm-yyyy)
		ImporteRectificacion	Base Rectificada							Base imponible de la factura/facturas sustituidas	Decimal(12,2)
			Cuota Rectificada							Cuota repercutida o soportada de la factura/facturas sustituidas	Decimal(12,2)
			CuotaRecargoRectificado							Cuota recargo de equivalencia de la factura/facturas sustituidas	Decimal(12,2)
		FechaOperacion								Fecha en la que se ha realizado la operación siempre que sea diferente a la fecha de expedición	Fecha(dd-mm-yyyy)
		ClaveRegimenEspecialOTrascendencia								Clave que identificará el tipo de régimen del IVA o una operación con trascendencia tributaria	Alfanumérico(2) L3.1
		ClaveRegimenEspecialOTrascendenciaAdicional1								Clave adicional que identificará el tipo de régimen del IVA o una operación con trascendencia tributaria	Alfanumérico(2) L3.1
		ClaveRegimenEspecialOTrascendenciaAdicional2								Clave adicional que identificará el tipo de operación o el régimen especial con trascendencia tributaria	Alfanumérico(2) L3.1
		NumRegistroAcuerdoFacturacion								Número de registro obtenido al enviar la autorización en materia de facturación o de libros registro	Alfanumérico(15)
		ImporteTotal								Importe total de la factura	Decimal(12,2)
		BaseImponibleACoste								Para grupos de IVA	Decimal(12,2)
		DescripcionOperación								Descripción del objeto de la factura	Alfanumérico(500)
		RefExterna								Referencia Externa. Dato adicional de contenido libre enviado por algunas	Alfanumérico(60)

Figura 3.5: Campos del libro registro facturas emitidas 1.

	FacturaSimplificadaArticulos7.2_7.3								Factura simplificada Artículo 7,2 Y 7,3 RD 1619/2012. Si no se informa este campo se entenderá que tiene valor "N	L26
	EntidadSucedida	NombreRazon							Nombre-razón social de la entidad sucedida como consecuencia de una operación de reestructuración	Alfanumérico(120)
		NIF							NIF asociado a la entidad sucedida como consecuencia de una operación de reestructuración	FormatoNIF(9)
	RegPrevioGGEEoREDEMEoCompetencia								Identificador que especifica aquellos registros de facturación con dificultades para enviarse en plazo por no tener constancia del cambio de condición a GGEE, de la inclusión en REDEME o de un cambio en la competencia inspectora.	Alfanumérico(1) L28
									Si no se informa este campo se entenderá que tiene valor "N".	
	Macrodato								Identificador que especifica aquellas facturas con importe de la factura superior a un umbral de 100.000.000 euros. Si no se informa este campo se entenderá que tiene valor "N".	Alfanumérico(1) L29
	DatosInmueble	DetalleInmueble	SituaciónInmueble						Identificador que especifica la situación del inmueble	Numérico(1) L6
			ReferenciaCatastral						Referencia catastral del inmueble	Alfanumérico(25)
	ImporteTransmisionInmueblesSujetoAIVA								Importe percibido por transmisiones de inmuebles sujetas a IVA	Decimal(12,2)
	EmitidaPorTercerosODestinatario								Identificador que especifica si la factura ha sido emitida por un tercero o por el destinatario. Si no se informa este campo se entenderá que tiene valor "N".	Alfanumérico(1) L10
	FacturaSinIdentifDestinatarioAritculo6.1.d								Factura sin identificación destinatario artículo 6.1.d) RD 1619/2012 Si no se informa este campo se entenderá que tiene valor "N".	L27
	Contraparte	NombreRazon							Nombre-razón social de la contraparte de la operación (cliente) de facturas expedidas	Alfanumérico(120)
		NIFRepresentante							NIF del representante de la contraparte de la operación	FormatoNIF(9)
		NIF							Identificador del NIF contraparte de la operación (cliente) de facturas expedidas	FormatoNIF(9)
		IDOtro	CodigoPais						Código del país asociado contraparte de la operación (cliente) de facturas expedidas	Alfanumérico(2) (ISO 3166-1 alpha-2 codes) L17
			IDType						Clave para establecer el tipo de identificación en el pais de residencia	Alfanumérico(2) L4
			ID						Número de identificación en el país de residencia	Alfanumérico(20)

Figura 3.6: Campos del libro registro facturas emitidas 2.

		TipoDesglose	DesgloseFactura	Sujeta	Exenta	DetalleExenta	CausaExencion			Campo que especifica la causa de la exención	Alfanumérico(2) L9
							BaseImponible			Importe en euros correspondiente a la causa de exención	Decimal(12,2)
					NoExenta	TipoNoExenta				Tipo de operación sujeta y no exenta para la diferenciación de inversión de sujeto pasivo	Alfanumérico(2) L7
						DesgloseIVA	DetalleIVA	TipoImpositivo		Porcentaje aplicado sobre la Base Imponible para calcular la cuota.	Decimal(3,2)
								BaseImponible		Magnitud dineraria sobre la cual se aplica un determinado tipo impositivo	Decimal(12,2)
								CuotaRepercutida		Cuota resultante de aplicar a la base imponible un determinado tipo impositivo	Decimal(12,2)
								TipoRecargoEquivalencia		Porcentaje asociado en función del tipo de IVA	Decimal(3,2)
								CuotaRecargoEquivalencia		Cuota resultante de aplicar a la base imponible el tipo de recargo de equivalencia	Decimal(12,2)
				NoSujeta	ImportePorArticulos7_14_Otros					Importe en euros si la sujeción es por el art. 7,14, otros	Decimal(12,2)
					ImporteTAIReglasLocalizacion					Importe en euros si la sujeción es por operaciones no sujetas en el TAI por reglas de localización	Decimal(12,2)
			DesgloseTipoOperacion	PrestacionServicios	Sujeta	Exenta	DetalleExenta	CausaExencion		Campo que especifica la causa de la exención	Alfanumérico(2) L9
								BaseImponible		Importe en euros correspondiente a la causa de exención	Decimal(12,2)
						NoExenta	TipoNoExenta			Tipo de operación sujeta y no exenta para la diferenciación de inversión de sujeto pasivo	Alfanumérico(2) L7
							DesgloseIVA	DetalleIVA	TipoImpositivo	Porcentaje aplicado sobre la Base Imponible para calcular la cuota.	Decimal(3,2)
									BaseImponible	Magnitud dineraria sobre la cual se aplica un determinado tipo impositivo	Decimal(12,2)
									CuotaRepercutida	Cuota resultante de aplicar a la base imponible un determinado tipo impositivo	Decimal(12,2)
					NoSujeta	ImportePorArticulos7_14_Otros				Importe en euros si la sujeción es por el art. 7,14, otros	Decimal(12,2)
						ImporteTAIReglasLocalizacion				Importe en euros si la sujeción es por operaciones no sujetas en el TAI por reglas de localización	Decimal(12,2)
				Entrega	Sujeta	Exenta	DetalleExenta	CausaExencion		Campo que especifica la causa de la exención	Alfanumérico(2) L9
								BaseImponible		Importe en euros correspondiente a la causa de exención	Decimal(12,2)
						NoExenta	TipoNoExenta			Tipo de operación sujeta y no exenta para la diferenciación de inversión de sujeto pasivo	Alfanumérico(2) L7
							DesgloseIVA	DetalleIVA	TipoImpositivo	Porcentaje aplicado sobre la Base Imponible para calcular la cuota.	Decimal(3,2)
									BaseImponible	Magnitud dineraria sobre la cual se aplica un determinado tipo impositivo	Decimal(12,2)
									CuotaRepercutida	Cuota resultante de aplicar a la base imponible un determinado tipo impositivo	Decimal(12,2)
									TipoRecargoEquivalencia	Porcentaje asociado en función del tipo de IVA	Decimal(3,2)
									CuotaRecargoEquivalencia	Cuota resultante de aplicar a la base imponible el tipo de recargo de equivalencia	Decimal(12,2)
					NoSujeta	ImportePorArticulos7_14_Otros				Importe en euros si la sujeción es por el art. 7,14, otros	Decimal(12,2)
						ImporteTAIReglasLocalizacion				Importe en euros si la sujeción es por operaciones no sujetas en el TAI por reglas de localización	Decimal(12,2)

Figura 3.7: Campos del libro registro facturas emitidas 3.

Debemos hacer uso del esquema que aparece en la figura 3.8 para trazar el camino de los campos reales a validar, teniendo en cuenta que queremos una factura del tipo *F2S1* que significa *F2*(Factura Simplificada) y *S1*(Sujeto No Exento) solo habrá que validar todos los campos que antes aparecian en rojo y no tienen hijos, es decir, los campos que sigan el recorrido de DesgloseFactura, Sujeta y No Exenta.

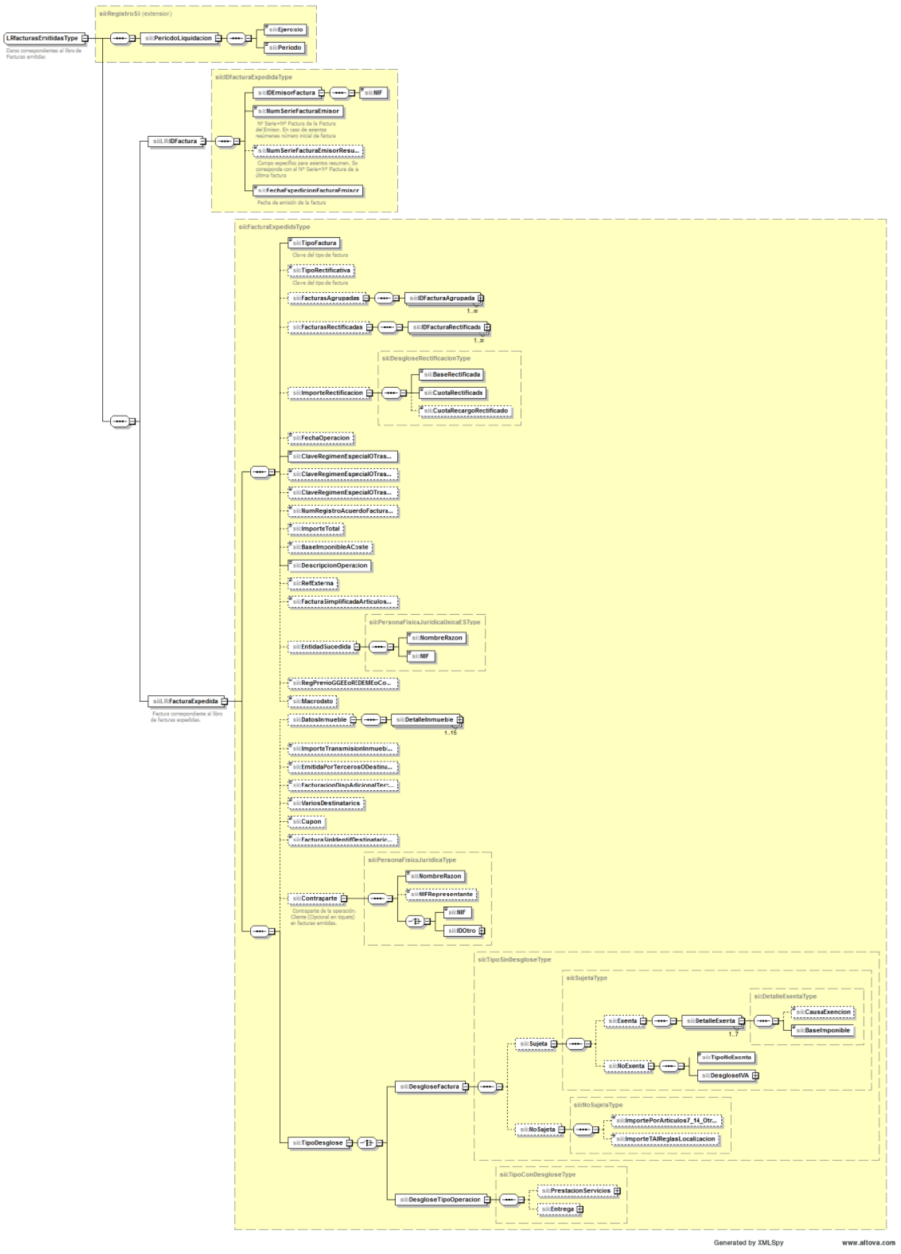


Figura 3.8: Recorrido XML para la factura F2S1.

En resumen, los campos que se deben validar se exponen a continuación: IDVersiónSII, NombreRazón, NIF(Titular), TipoComunicación, Ejercicio, Período, NumSerieFacturaEmisor, FechaExpediciónFacturaEmisor, TipoFactura, ClaveRégimenEspecialOTranscendencia, DescripciónOperación, SituaciónInmueble, NombreRazón(Contraparte), NIF(Contraparte), IDType, ID, TipoNoExenta y BaseImponible.

En cuanto al **RQ03**, al introducir un valor en el campo "Country Code: Code[2]", la codeunit informará de un error si dicho código no se encuentra en la lista de países válidos como se muestra en la figura 3.9. En caso de que exista nos podrá dar una segunda excepción si el país no forma parte de la Unión Europea, es decir, el test será correcto si el valor que introducimos es un país válido y se encuentra dentro de la Unión Europea. En este caso aunque sea una factura del **"Tipo libro registro"**: *Facturas Emitidas* del subtipo *F2S1*, solo se pide que validemos el campo "Country Code" por eso no hacemos uso de las imágenes del RQ01.

VALORES	DESCRIPCIÓN
DE	ALEMANIA
AT	AUSTRIA
BE	BELGICA
BG	BULGARIA
CZ	CHECA, REPUBLICA
CY	CHIPRE
HR	CROACIA
DK	DINAMARCA
SK	ESLOVAQUIA
SI	ESLOVENIA
EE	ESTONIA
FI	FINLANDIA
FR	FRANCIA
GR	GRECIA
HU	HUNGRIA
IE	IRLANDA
IT	ITALIA
LV	LETONIA
LT	LITUANIA
LU	LUXEMBURGO
MT	MALTA
NL	PAISES BAJOS
PL	POLONIA
PT	PORTUGAL
GB	REINO UNIDO
RO	RUMANIA
SE	SUECIA

Figura 3.9: Países válidos para el RQ02

Para el **RQ04** hacemos uso de la figura 3.10 en la que tenemos los campos que representan el **"Tipo libro registro"** del Suministro de Pagos para facturas registradas en el Libro de registro de Facturas Recibidas *(Pagos Facturas Recibidas(RECC))*. En este caso, no hay un recorrido que seguir como en el RQ01, ya que se hacen test para todos los tipos de factura, entonces se validan todos los campos.

BLOQUE	DATOS/ AGRUPACIÓN	DATOS/ AGRUPACIÓN	DATOS/ AGRUPACIÓN	DATOS/ AGRUPACIÓN	DESCRIPCIÓN	FORMATO LONGITUD LISTA
Cabecera	IDVersionSii				Identificación de la versión del esquema utilizado para el intercambio de información	Alfanumérico(3) L19
	Titular	NombreRazon			Nombre-razón social del Titular del libro de registro de facturas recibidas	Alfanumérico(120)
		NIFRepresentante			NIF del representante del titular del libro de registro	FormatoNIF(9)
		NIF			NIF asociado al titular del libro de registro	FormatoNIF(9)
RegistroLRPagos	IDFactura	IDEmisorFactura	NombreRazon		Nombre-razón social del emisor de la factura	Alfanumérico(120)
			NIF		Identificador del NIF del emisor de la factura	FormatoNIF(9)
			IDOtro	CodigoPais	Código del país asociado a la contraparte de la factura	Alfanumérico(2) (ISO 3165-1 alpha-2 codes) L17
				IDType	Clave para establecer el tipo de identificación en el país de residencia	Alfanumérico(2) L4
				ID	Número de identificación en el país de residencia	Alfanumérico(20)
		NumSerieFacturaEmisor			Nº Serie+Nº Factura que identifica a la factura emitida (en su caso primera factura del asiento resumen)	Alfanumérico(60)
		FechaExpedicionFacturaEmisor			Fecha de expedición de la factura	Fecha(dd-mm-yyyy)
	Pagos	Pago	Fecha		Fecha de realización del pago	Fecha(dd-mm-yyyy)
			Importe		Importe pagado	Decimal(12,2)
			Medio		Medio de pago utilizado	Alfanumérico(2) L11
			Cuenta_O_Medio		Cuenta bancaria o medio de pago utilizado	Alfanumérico(34)

Figura 3.10: Campos del Suministro de Pagos del Libro de registro de Facturas Recibidas

En cuanto a los campos que se deben validar se encuentran los siguientes: IDVersiónSII, NombreRazón(Titular), NIF(Titular), NombreRazón(Contraparte), NIF(Contraparte), IDType, ID, NumSerieFacturaEmisor, FechaExpediciónFacturaEmisor, Fecha, Importe y Medio.

Finalmente, respecto al **RQ05**, consultamos la figura 3.11 donde se observan los campos que componen el **"Tipo libro registro"** del Suministro de Cobros para facturas registradas en el Libro de registro de Facturas Expedidas *(Cobros Facturas Emitidas(RECC))*. En este caso, como en el anterior requisito, se hacen tests para todos los tipos de facturas.

BLOQUE	DATOS/ AGRUPACIÓN	DATOS/ AGRUPACIÓN	DATOS/ AGRUPACIÓN	DATOS/ AGRUPACIÓN	DESCRIPCIÓN	FORMATO LONGITUD LISTA
Cabecera	IDVersionSii				Identificación de la versión del esquema utilizado para el intercambio de información	Alfanumérico(3) L19
	Titular	NombreRazon			Nombre-razón social del Titular del libro de registro de facturas expedidas	Alfanumérico(120)
		NIFRepresentante			NIF del representante del titular del libro de registro	FormatoNIF(9)
		NIF			NIF asociado al titular del libro de registro	FormatoNIF(9)
RegistroLRCobros	IDFactura	IDEmisorFactura	NIF		Identificador del NIF del emisor de la factura	FormatoNIF(9)
		NumSerieFacturaEmisor			Nº Serie+Nº Factura que identifica a la factura emitida (en su caso primera factura del asiento resumen)	Alfanumérico(60)
		FechaExpedicionFacturaEmisor			Fecha de expedición de la factura	Fecha(dd-mm-yyyy)
	Cobros	Cobro	Fecha		Fecha de realización del cobro	Fecha(dd-mm-yyyy)
			Importe		Importe cobrado	Decimal(12,2)
			Medio		Medio de cobro utilizado	Alfanumérico(2)

Figura 3.11: Campos del Suministro de Cobros del Libro de registro de Facturas Expedidas.

Los campos que se deben validar son: IDVersiónSII, NombreRazón, NIF(Titular), NIF(Contraparte), NumSerieFacturaEmisor, FechaExpediciónFacturaEmisor, Fecha, Importe y Medio.

3.3. Diseño de software

En cuanto al diseño de la arquitectura, en el desarrollo actual no existe, debido a que todo el software desarrollado se encuentra en la parte del servidor. No se trata de una aplicación completa en la que hay parte servidor y parte del cliente sino de una aplicación para desarrolladores, ya que solo en la parte del servidor es donde estos se llevan a cabo.

Como hemos descrito anteriormente, el proyecto consiste en realizar unas pruebas de validación. Para realizar este desarrollo haremos uso de codeunits, una distinta para cada requisito, es decir, realizaremos un total de cinco codeunits.

Las codeunit son objetos que se encargan de contener el código en el lenguaje AL para que otros objetos que contienen Business Central puedan ejecutarlo como son las funciones y eventos.

A cada codeunit que vamos a desarrollar la hemos decidido nombrar teniendo en cuenta la notación que utiliza la empresa para el resto de proyectos. Dicho esto la lista de codeunits que tenemos es la siguiente:

- codeunit01 → SIITESTServicioWebAEAT
- codeunit02 → SIITESTCreatedAndCaptureF2S1
- codeunit03 → SIITESTComprobarPaís
- codeunit04 → SIITESTPagosFacturasEmitidas
- codeunit05 → SIITESTFacturasRecibidasCobros

Como hemos mencionado en el capítulo anterior hemos tenido que identificar unos campos que procedían de un documento de la AEAT. Para conocer la procedencia de estos campos dentro de Business Central hemos hecho uso de los recursos proporcionados por la empresa, en la codeunit dentro del módulo del SII llamada "SIISendbyServerJobQueue.Codeunit.al" encontramos todos los nombres de los campos y asociados a su respectiva tabla dentro del estándar de Microsoft.

Los SIITEST es el nombre que le damos al test según lo determinado por la empresa. Se basan en codeunits que se estructuran dentro del entorno de programación, como se muestra en el listing 3.1.

```
codeunit 89019 "SIITESTEjemplo"
{
    Subtype = Test;
    Permissions = tabledata "VAT Entry" = rimd;

    var

    [Test]
    procedure Test01()
    var
    begin

    end;
}
```

Listing 3.1: Estructura de test

Cada codeunit se identifica por un ID único y un nombre, estos ID se encuentran dentro de un rango definido en la configuración, en este caso el rango de ID dentro de la configuración de la aplicación del SII va desde el ID 89000 hasta el 89050. En caso de completar todos los ID's solo habría que ampliar el rango en la configuración. Seguidamente se explica brevemente la configuración general de la codeunit establecida por el estándar de Microsoft.

Cada test que se realiza viene encabezado por la consigna [Test] indicando así que la próxima función que se realice se trata de un test. A continuación, hacemos referencia a la función y al nombre del test. Seguidamente habrá que declarar todas las variables locales dentro de 'var' (más arriba hemos realizado lo mismo pero con las variables globales). Nuestro código vendrá encapsulado entre las consignas begin end.

3.4. Diseño de las interfaces

En esta sección se hablan de las interfaces desarrolladas a lo largo del proyecto pero en nuestro caso no existe una interfaz como tal porque nuestro desarrollo solo se basa en el entorno de programación, pero hacemos uso de una interfaz del estándar de Business Central llamada "Test Tool" donde podemos ver el listado de codeunits desarrolladas para testear y desde donde se validaron los futuros tests de forma automática.

En la figura 3.12 tenemos el diseño de la interfaz que usaremos para la futura validación y verificación, con esto comprobaremos si los tests automáticos superan todas las pruebas.

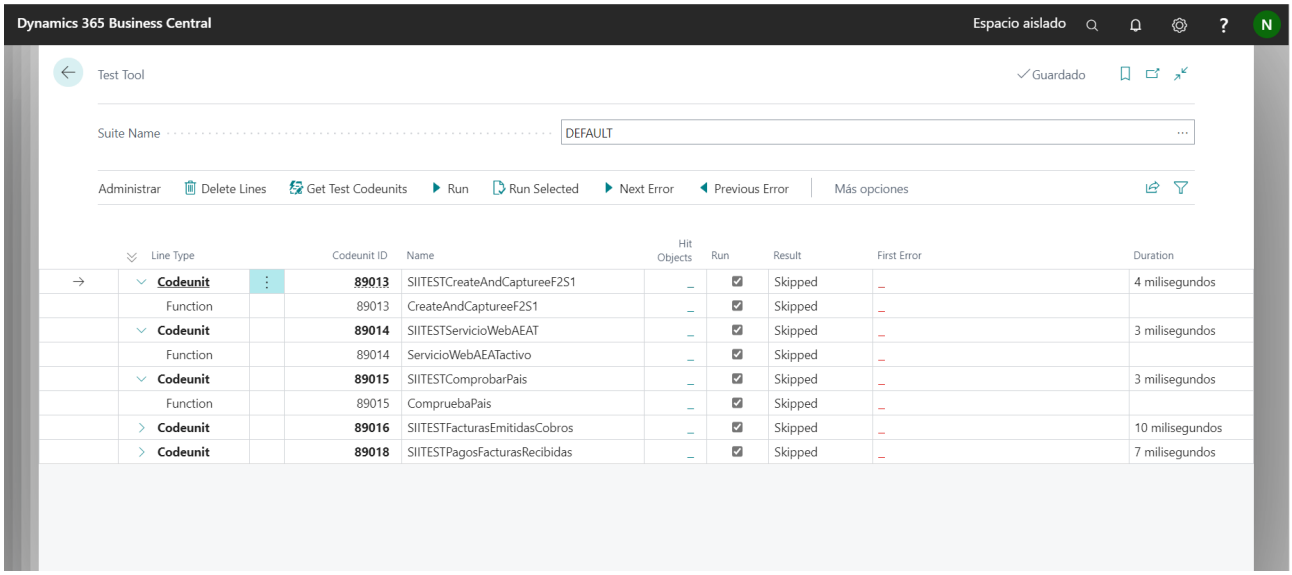


Figura 3.12: Interfaz Test Tool.

Capítulo 4

Implementación y pruebas

En el capítulo actual vamos a proceder a realizar la descripción técnica de lo que hemos realizado y desarrollado a lo largo del proyecto con todas las decisiones que se han ido tomando y las respectivas instalaciones de las aplicaciones y software utilizados. Por último, explicaremos las pruebas realizadas en la verificación y validación del producto.

4.1. Instalación y preparación del entorno

En esta sección hablaremos del software necesario para el desarrollo del proyecto y cómo lo hemos instalado.

El desarrollo se va a llevar a cabo en el entorno de Visual Studio Code, para ello debemos instalar la aplicación, ya que Microsoft Business Central hace uso de su propio lenguaje de programación llamado AL. Para el almacenamiento de la información de Business Central se utiliza un contenedor (Docker) que la compañía nos proporciona.

Para instalar la aplicación del SII haremos uso de los scripts descritos a continuación:

- InstallOrUpdateDockerEngine.ps1: este script se utiliza para instalar la versión del docker más reciente en nuestro equipo, requiere de un sistema operativo Windows 10 para su funcionamiento.
- InstallPortainerForDocker.ps1: dicho script instala el contenedor llamado “portainer.io”, se debe indicar el puerto sobre el que se va a trabajar.
- InstallUpGradeBcContainerHelper.ps1: el script instala o actualiza los comandos necesarios para instalar Business Central.
- SetupBCContainerLocalContainerNameBC19.ps1: con este script creamos un nuevo contenedor de Business Central, este será sobre el que se trabajará a partir de ahora. Se deben indicar tanto la versión de Business Central que se descarga y el tipo (online o local). Para su correcto funcionamiento hacemos uso de una licencia proporcionada por la empresa.

Una vez tenemos todo instalado dentro de Visual Studio Code instalaremos las extensiones necesarias para una programación más fluida y eficiente. Estas extensiones son las siguientes:

- AL Extension Pack[18]: paquete de extensiones que contiene al completo las extensiones que se necesitan para el correcto desarrollo del proyecto con el lenguaje AL.
- AL Language Tools[19]: extensión que nos permite utilizar atajos para agilizar la programación eficiente de objetos en AL.
- AL Object Designer[20]: esta extensión es la más importante, ya que nos permite el acceso al código fuente del estándar proporcionado por Business Central. Esto nos sirve para visualizar el funcionamiento de todas las funciones y campos que contiene el estándar y poder hacer las respectivas modificaciones o desarrollo del nuevo código.
- Git Graph[17]: dicha extensión permite usar una vista gráfica sobre la gestión del sistema de control de versiones en Git.

Finalmente, creamos un repositorio en Azure DevOps para activar el control de versiones desde nuestro Visual Studio para poder tener nuestro proyecto siempre en la nube y que el supervisor pueda revisarlo regularmente.

4.2. Estructura del código

La aplicación del SII está estructurada en dos módulos, como todas las extensiones que se desarrollan para el Marketplace de Business Central. Estos se dividen en un módulo para el desarrollo de la aplicación y otro módulo para el testing de la misma.

La organización general de carpetas en los dos módulos es prácticamente idéntica, existe una pequeña diferencia dentro del módulo del testing y en él aparece una sección extra llamada test, donde se desarrollan las codeunits para comprobar el funcionamiento correcto de la aplicación.

Dicho esto, el código realizado a lo largo de la estancia se encuentra en la carpeta de 'test' dentro del área de trabajo del módulo del SII. Se trata de 5 codeunits independientes, cada una con sus respectivas funciones y comprobaciones, es decir, no hay una codeunit dependiente de la anterior.

El contenido importante que hay que destacar dentro de la carpeta 'sii_test' se encuentra descrito a continuación:

- .alpackages: esta carpeta es creada al descargar los símbolos del sistema, esto se hace al conectar por primera vez el entorno de desarrollo con Business Central. Dentro se encuentran los símbolos del sistema y el código fuente del estándar de Business Central para visualizarlo.

- .vscode: en la carpeta encontramos los ficheros de configuración del proyecto. Se destaca entre ellos el launch.json, en el cuál se encuentra la información de conexión con el contenedor de Business Central que hemos instalado en nuestro equipo. Además debemos indicar, dentro de este fichero, la dirección del servidor, instancia y la autenticación entre otros.

- test: contiene el código desarrollado para los tests de la aplicación.

- app.json: en este fichero encontraremos la configuración principal del proyecto. En él se encuentra la información de la aplicación a desarrollar, es decir, su nombre, versión, identificador, dependencias que posee y quién lo ha publicado.

En la figura 4.1 podemos observar la distribución del módulo de testing dentro de la aplicación del SII.

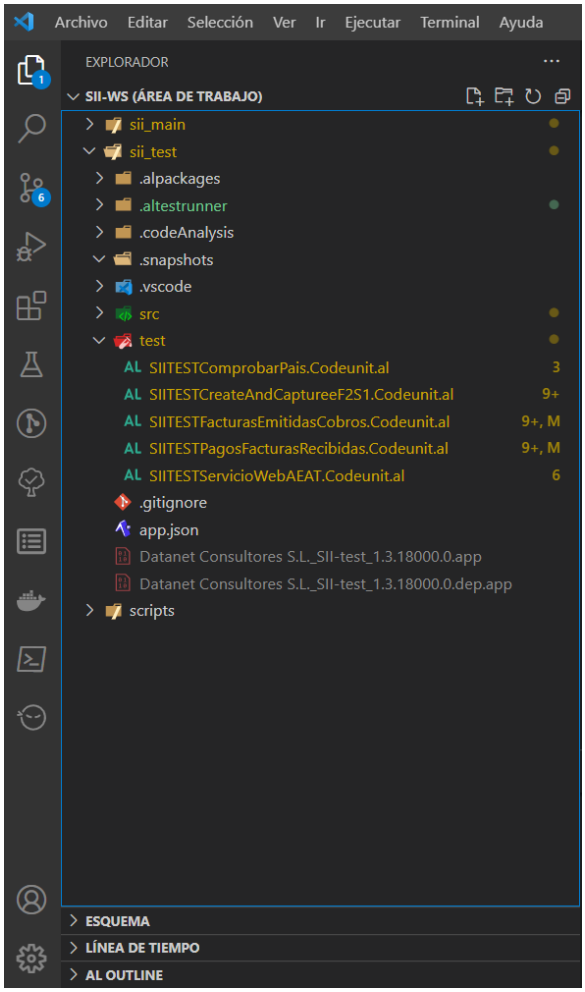


Figura 4.1: Estructura carpetas módulo de test del SII.

4.3. Descripción técnica de la implementación

En esta sección vamos a proceder a describir detalladamente la implementación realizada a lo largo del desarrollo del proyecto. En nuestro caso procederemos a dividirlo en apartados, comenzaremos con un apartado común para el desarrollo de todas las codeunits y luego un apartado por cada desarrollo de codeunit realizado independientemente de las demás.

4.3.1. Desarrollo común a todas las codeunits

Para desarrollar estos tests necesitamos utilizar las tablas del estándar de Business Central que se inicializan vacías, es por ello que tenemos que crear todos los datos de una forma aleatoria. Estos datos son introducidos aleatoriamente porque para realizar las validaciones no nos interesa el valor de los datos sino que nosotros validamos los formatos de los datos.

Teniendo en cuenta todo lo expuesto anteriormente, procedemos a explicar la realización de nuestro desarrollo. El procedimiento general para el desarrollo de las distintas codeunits es el siguiente:

En primer lugar debemos configurar la aplicación del SII, esto viene determinado por el estándar de Business Central con la configuración de: “entorno de desarrollo para pruebas”, como se muestra en el listing 4.1.

```
SIILibrary.CreateConfiguracionSIICert(SIISIISetup,
       SIISIISetup."Inform Date of Issue"::"Posting Date",
       SIISIISetup."Certificate Location"::"Local Computer",
       false, false, false, false, SIISIISetup."Check FE Key 02"::" ",
       SIISIISetup."Check FR Key 13"::" ",
       SIISIISetup."Check Keys F5 and LC"::" ",
       SIISIISetup."Check SII Classification S1"::" ",
       SIISIISetup."Check SII Classification S2"::" ",
       IsolatedCertificate.Code, true);
```

Listing 4.1: Configuración SII.

Visto desde la interfaz de Business Central se muestra figura 4.2.

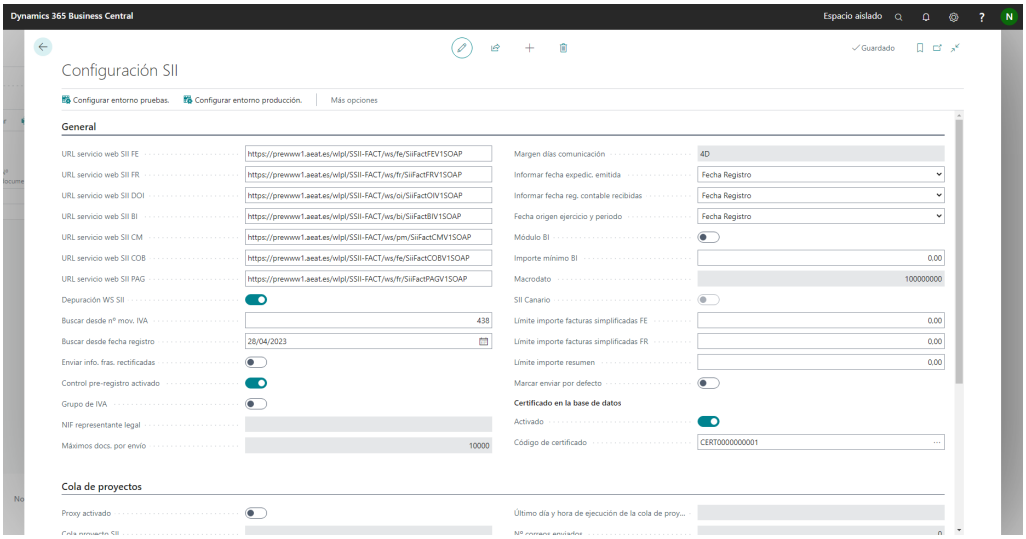


Figura 4.2: Configuración del SII.

A continuación debemos replicar una factura de venta, como la haría cualquier usuario, desde la interfaz de Business Central, en cambio nosotros la realizaremos por código. Vamos a proceder a explicar los pasos realizados.

Con las siguientes líneas de código creamos un cliente de forma aleatoria con las funciones procedentes del estándar de Business Central. En el listing 4.2 se muestra el código para crear el cliente.

```
LibrarySales.CreateCustomerWithVATRegNo(Customer);
VATOperational := CopyStr(Customer."VAT Registration No.", 1, 9);
Customer.Validate(Customer."VAT Registration No.", VATOperational);
Customer.Modify();
SIILibrary.SetUpCustomerSII(Customer,
    Customer."SII Resid. Country Tax ID Code"::"01-NIF (SPAIN)", false);
```

Listing 4.2: Creación cliente.

La interfaz de Business Central para crear un cliente corresponde a la figura 4.3.

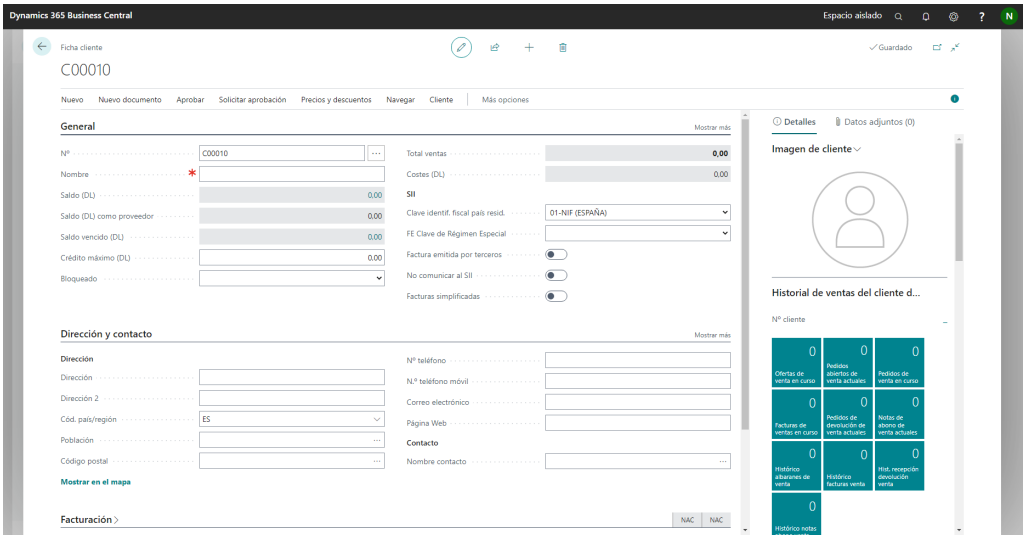


Figura 4.3: Interfaz creación cliente.

Se crea la factura y completamos los campos que se encuentran en la cabecera pertenecientes al cliente, creado anteriormente. Esto se muestra en el listing 4.3.

```
1 LibrarySales.CreateSalesHeader(SalesHeader,
2     "Sales Document Type".FromInteger
3         (SalesHeader."Document Type"::Invoice.AsInteger()), Customer."No.");
4 codeGLAccount := LibraryERM.CreateGLAccountWithSalesSetup();
```

Listing 4.3: Creación cabecera factura.

En Business Central, la interfaz que genera la creación de la factura viene representada en la figura 4.4.

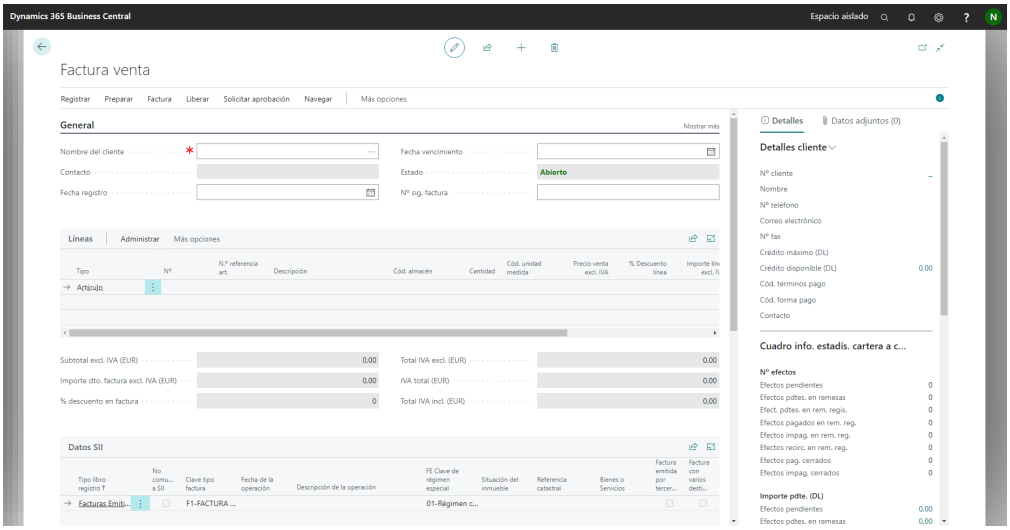


Figura 4.4: Interfaz creación factura.

Una vez completados los campos del cliente en la factura, debemos crear y configurar nuestro grupo registro de IVA, que hacen referencia a los campos que se comunicaran al SII, con esto lo que conseguimos es poner un IVA del 21 % a nuestra factura de venta.
El campo *"SII FE Special Regime Key"* tiene el valor "01" que hace referencia, en este caso, al régimen común, este campo más adelante será modificado. El código de la creación del grupo de IVA se muestra en el listing 4.4.

```
SIILibrary.CreateVATBusinessPostingGroup('VATF2S1', VATBusinessPostingGroup,
    VATBusinessPostingGroup."SII FE Special Regime Key"::"01",
    VATBusinessPostingGroup."SII FR Special Regime Key"::" ");
SIILibrary.CreateVATProductPostingGroup('VATTEST3', VATProductPostingGroup);
SIILibrary.CreateVATPostingSetUp(VATPostingSetup, 'VATF2S1', 'VATTEST3',
    VATPostingSetup."VAT Calculation Type"::"Normal VAT",
    VATPostingSetup."Unrealized VAT Type"::" ", '47700001', '',
    '47200001', '', '', '', 'VATTEST3', 21,
    VATPostingSetup."SII Classification"::"S1-Subject Not Exempt",
    VATPostingSetup."SII Intra-communit. Op. Type"::" ");
```

Listing 4.4: Configuración grupo registro IVA.

La interfaz desde la que completamos los campos para la creación del grupo del IVA se muestra en la figura 4.5.

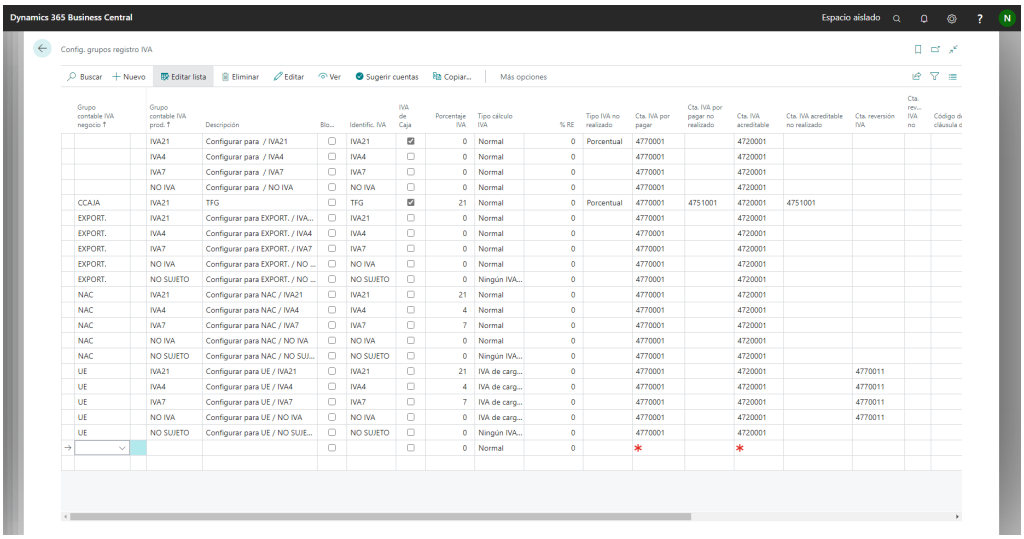


Figura 4.5: Interfaz configuración grupo registro de IVA.

Continuamos actualizando la factura con los nuevos datos adicionales del SII para que se nos indique el tipo de factura que vamos a realizar.
El codigo añadido se muestra en el listing 4.5.

```
SIIAdditionalData.Get(SIIAdditionalData."Posting Book Type"::"Issued Invoices",
    SIIAdditionalData."Document Type"::Invoice, SalesHeader."No.");
SIILibrary.ModifyDatosAdicionalesSIIServiceSales(SIIAdditionalData,
    SIIAdditionalData."Invoice Type Key"::"F2",
    SIIAdditionalData."FE Special Regime Key"::"01",
    true, false, SIIAdditionalData."Operation Date",
```

```
    SIIAdditionalData."FRECT Rectified Base",
    SIIAdditionalData."FRECT Rectified Fee");
```

Listing 4.5: Configuración datos adicionales del SII.

La figura 4.6 hace referencia a cómo se ve desde la interfaz de Business Central esta configuración adicional de datos del SII.

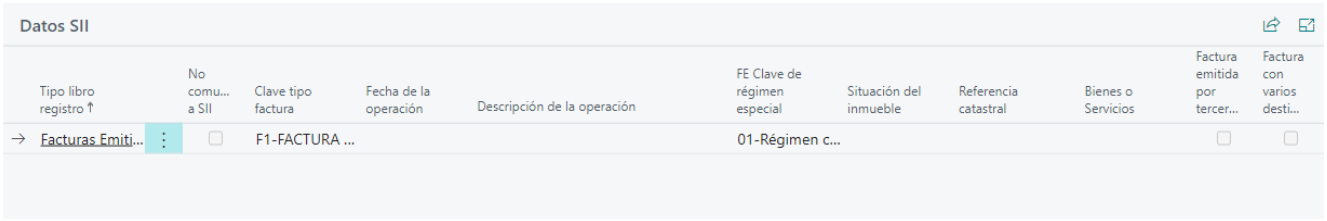


Figura 4.6: Interfaz datos adicionales del SII.

Realizamos la creación de las líneas de venta que indican los artículos que vamos a vender y la cantidad, todo esto se crea de forma aleatoria haciendo uso de la librería "Library - Random" del estándar de Business Central.
Una vez realizado, procedemos a registrar la factura. Esto se consigue con una función llamada "PostSalesDocument", que le pasamos como parámetros la cabecera de la factura y le indicamos que nos genere una nueva factura con los booleanos. Esto simula como si hiciésemos clic al botón de "registrar" desde la interfaz que generaría el albarán de venta.
Se muestra el codigo en el listing 4.6.

```
LibrarySales.CreateSalesLine(SalesLine, SalesHeader,
    "Sales Line Type".FromInteger(SalesLine.Type::"G/L Account".AsInteger()),
    codeGLAccount, 1);
SalesLine.Validate("Unit Price", LibraryRandom.RandDecInRange(0, 1000, 0));
SalesLine.Validate("VAT Prod. Posting Group", VATProductPostingGroup.Code);
SalesLine.Validate("VAT Bus. Posting Group", VATBusinessPostingGroup.Code);
SalesLine.Modify(true);
codeDocumentNo := LibrarySales.PostSalesDocument(SalesHeader, true, true);
```

Listing 4.6: Creación líneas de venta.

En la figura 4.7 se muestra cómo se ve la interfaz de Business Central de la creación de líneas de venta.

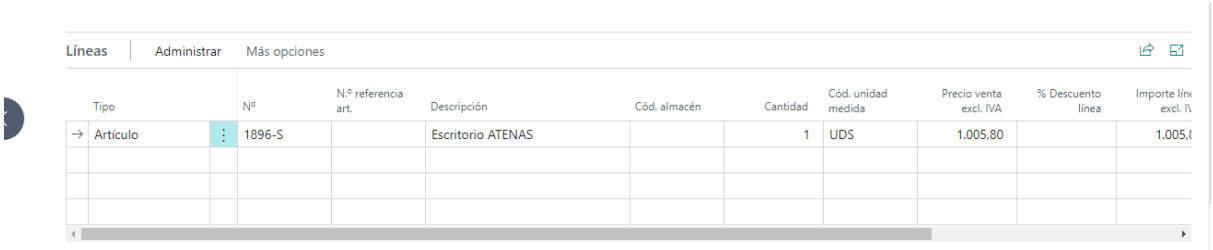


Figura 4.7: Interfaz creación líneas de venta.

Una vez realizada la factura y registrada debemos dirigirnos a la página de “Comunicación SII” donde, con los comandos que tenemos a continuación, se realizará un evento llamando a la acción “CaptureData” para capturar el SII. Esto capturará el movimiento de IVA de la factura para su futura comunicación a la agencia tributaria.
El código de la captura del movimiento de IVA se muestra en el listing 4.7.

```
TestPageSIICommunication.OpenView();
TestPageSIICommunication."CaptureData".Invoke();
```

Listing 4.7: Captura de facturas para el SII.

En la figura 4.8 se muestra cómo sería la captura del movimiento de IVA de una factura desde la interfaz de Business Central.

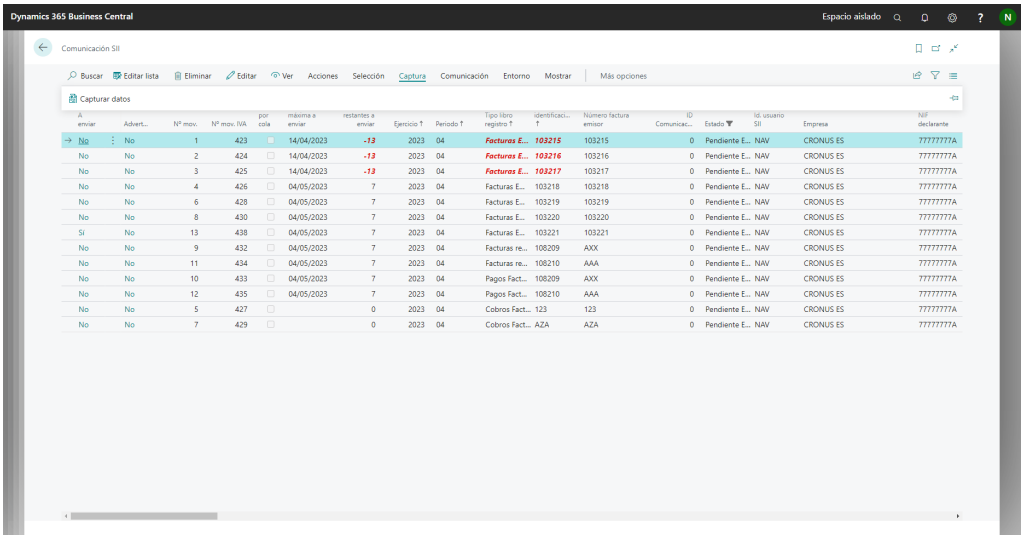


Figura 4.8: Interfaz capturar datos del SII.

Una vez capturada hay que buscar la factura. Esto se logrará filtrando por el número de factura que nos hemos guardado en una variable anteriormente con la función “SetRange” y, seguidamente, con la función “FindFirst” que coloca el puntero en la primera factura encontrada con el filtro dado.
En el listing 4.8 se muestra el código para buscar la factura.

```
SIICommunication.Reset();
SIICommunication.SetRange("Issuer Invoice Number", codeDocumentNo);
If SIICommunication.FindFirst() then begin
    //Validaciones
end;
```

Listing 4.8: Filtrar facturas.

Todo lo anteriormente realizado es común en todas las facturas que, a su vez, generamos en las codeunits, ya que es la forma de realizar una factura con movimiento de IVA. A continuación, se procede a explicar las diferentes validaciones realizadas en cada test.

4.3.2. SIITESTServicioWebAEAT

Para la actual codeunit existe una modificación respecto a las facturas generales, en este caso, como lo que nos piden es comprobar que el servicio web de la AEAT no está caído, debemos intentar comunicar las facturas con la agencia tributaria.
Esto se realiza mediante el fragmento de código que aparece a continuación, en el que primero marcamos la factura que hemos realizado y, seguidamente, la comunicamos con el SII.
El código se muestra en el listing 4.9.

```
1 TestPageSIICommunication.MarkUnmark.Invoke();
2 TestPageSIICommunication.SendFromServer.Invoke();
```

Listing 4.9: Comunicar al SII.

En la figura 4.9 se muestra como sería la comunicación con el SII de una factura. Esto enviaría directamente a la AEAT los movimientos de IVA de las facturas marcadas.

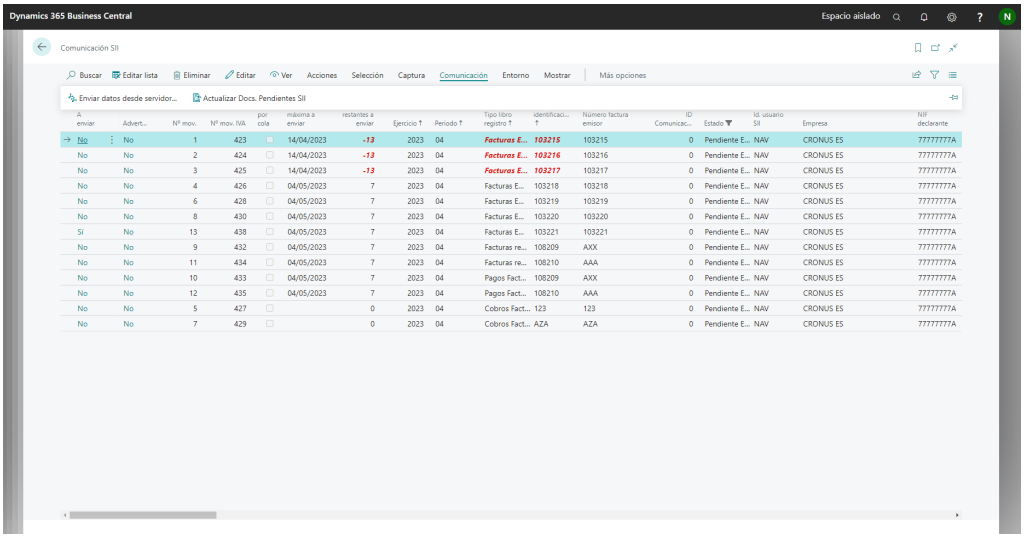


Figura 4.9: Interfaz comunicación SII.

Una vez realizado el envío, en caso de que el servidor funcione, los datos de dicha factura se habrán actualizado y estas son las validaciones que realizaremos para comprobar si el servicio web de la AEAT funciona correctamente.

Los campos que se modifican al comunicar con la agencia tributaria son el campo “Comunication ID” que, en estado inicial, se encuentra a “0” y en caso de comunicación se modificará a “1”, y el campo “Status” el cual su estado inicial al capturar una factura aparecerá como “Pending Send”, pero una vez comunicada aparecera como “Error”. Esta variable que aparece como “Error” es debida a que nos encontramos en un entorno de pruebas y no vamos a poder comunicar nada real a la agencia tributaria.

Se muestran las respectivas validaciones en el listing 4.10.

```
If SIICommunication.FindFirst() then begin

    Assert.IsTrue(SIICommunication."Communication ID" = 1,
        StrSubstNo(ServiceErr, Format(SIICommunication."Communication ID" = 1),
            Format(SIICommunication."Communication ID")));
    Assert.IsTrue(SIICommunication."Status" = SIICommunication.Status::Error,
        StrSubstNo(ServiceErr,
        Format(SIICommunication."Communication ID" =
                SIICommunication.Status::Error),
            Format(SIICommunication."Status")));
end
else
    Assert.IsTrue(false, CapturedErr);
```

Listing 4.10: Validaciones SIITESTServicioWebAEAT.

A continuación se explican dos tipos de eventos necesarios para la correcta funcionalidad de esta codeunit.

El primero de estos eventos lo que realiza es que Business Central antes de comunicar con el SII desde la interfaz nos aparece un evento tipo burbuja en el que nos pregunta si deseamos comunicar a la Agencia Tributaria las facturas con IVA y tenemos las opciones de sí y no. Desde el entorno de programación esto se representa con el fragmento de código mostrado en el listing 4.11.

```
[ConfirmHandler]
procedure ConfirmHandler(Question: Text; var Reply: Boolean)
Var
    Frist: Boolean;
    QuestionText: Label 'Desea enviar al entorno de pruebas del SII?';
begin
    If Question = StrSubstNo(QuestionText) then
        Reply := true

    else
        Reply := false;
end;
```

Listing 4.11: Tipo de evento 1.

El segundo de ellos realiza que, cuando nosotros marcamos una factura para comunicar con el SII dentro de Business Central, nos aparece un mensaje diciendo que hemos marcado “X” facturas y, simplemente, nos aparece el botón de aceptar y, en este caso, el usuario debe hacer click en este botón. Esto, desde el entorno de programación se realiza como aparece en el código del listing 4.12.

```
[MessageHandler]
Procedure MessageHandler(Msg: Text[1024])
begin
end;
```

Listing 4.12: Tipo de evento 2.

Esta es la modificación que habría que realizar al inicio de la codeunit, respecto al esquema general, para que los dos eventos explicados anteriormente se ejecuten de manera correcta. Se muestra esta modificación del código en el listing 4.13.

```
[Test]
[HandlerFunctions('MessageHandler,ConfirmHandler')]
procedure ServicioWebAEATactivo()
var
begin

end;
```

Listing 4.13: Modificación estructura test.

4.3.3. SIITESTCreateAndCaptureF2F1

En esta codeunit se observan como ejemplo un par de validaciones que hemos realizado, ya que, en general, se suele seguir la misma dinámica, es decir, comprobar que el campo no esté vacío para que al comunicar con el SII no provoque error por falta de datos. En este caso, como ejemplo, hemos escogido la validación del tipo de factura *F2* y que el NIF esté compuesto por 9 caracteres.

La sintaxis del código a seguir es siempre la misma, una validación comienza por la palabra Assert y, a continuación, podemos escoger entre IsTrue o IsFalse, en nuestro caso siempre vamos a comprobar que la condición sea cierta.

Podemos observar, como se muestra a continuación, que lo primero que se realiza es la comprobación y, en caso de que devuelva True continuaremos con la siguiente comprobación. En caso contrario entraría en el StrSubstNo donde, como primer parámetro, tenemos el nombre que se le da a la variable que nos devolverá el error, el siguiente campo sería la condición que debe cumplirse y en el último parámetro tenemos el campo del cual debe hacerse dicha validación. El código se muestra en el listing 4.14.

```
//TipoFactura
Assert.IsTrue(SIICommunication."Invoice Type Key" =
    SIICommunication."Invoice Type Key"::"F2",
    StrSubstNo(InvoiceTypeKeyErr,
    Format(SIICommunication."Invoice Type Key"::"F2"),
    Format(SIICommunication."Invoice Type Key")));

//NIF(Titular) --Para decir que tiene que ser igual que 9 caracteres
Assert.IsTrue(StrLen(SIICommunication."Declarant VAT") = 9,
    StrSubstNo(DNIErr,
    Format(MaxStrLen(SIICommunication."Declarant VAT") = 9),
    Format(SIICommunication."Declarant VAT")));
```

Listing 4.14: Validaciones.

En caso de que la comprobación anterior nos hubiera dado Falso, el primer campo del StrSubstNo habría cogido la variable InvoiceTypeKeyErr y nos habría devuelto el texto que aparece. Lo mismo ocurre en el caso del NIF. El código de las excepciones se representa en el listing 4.15

```
1 InvoiceTypeKeyErr: Label 'La Clave tipo factura debe ser %1 y es %2.';
2 DniErr: Label 'El DNI no cumple con la condicion de 9 caracteres';
```

Listing 4.15: Excepciones de error.

4.3.4. SIITESTComprobarPaís

En esta codeunit hemos seguido los pasos generales para generar la factura y capturarla con el SII, como se ha explicado anteriormente hasta la fase de validación.

Se nos pedía que el código introducido por el cliente fuera válido dentro de una lista de países disponibles siguiendo la normativa de la AEAT.

En este caso lo que se realiza es: primero creamos una variable booleana para saber si encontramos el país, seguidamente realizamos una comprobación para asegurarnos de que el país introducido no está vacío, en caso de estar vacío nos devuelve el primer mensaje de error correspondiente a "CodPaisErr", en caso contrario continua y recoge en una variable el país y vuelve a filtrarse en la tabla del estándar "Country/Region" que nosotros la llamamos "Pais" y que hace referencia a la tabla con todos los códigos de países que tiene Business Central.

Una vez realizado creamos un bucle utilizando el repeat junto con until pais.next() = 0 que quiere decir que, hasta que no acabe de recorrer toda la tabla de países no parará el bucle. Continúa entrando en una condición en la que compara si el código del país que se introduce es el mismo que el que hay en la tabla, en caso de no ser así, se vuelve a repetir el bucle con el siguiente país de la tabla hasta encontrarlo.

Si la condición anterior se cumple se comprueba si ese país está dentro de la Unión Europea con un método. Esto nos servirá para ver si el país se está dentro de la lista de países válidos que se encuentra en el RQ02. En caso afirmativo la variable creada al inicio cambia a True y sale del bucle y el test es válido, en caso de no pertenecer a la Unión Europea nos salta el mensaje de error correspondiente a "NoUEErr", en caso contrario se vuelve a repetir el bucle con el siguiente país de la tabla hasta encontrarlo.

El código descrito viene representado en el listing 4.16.

```
SIIComunication.Reset();
SIIComunication.SetRange("Issuer Invoice Number", codeDocumentNo);
If SIIComunication.FindFirst() then begin
    encontrado := false;
    SIIComunication.Validate(SIIComunication."Country Code", 'ES');

    if (SIIComunication."Country Code" <> '') then begin
        codePais := SIIComunication."Country Code";
        Pais.SetRange(Code, codePais);
        repeat
            if (SIIComunication."Country Code" = Pais.Code) then begin
                if (Pais.IsEUCountry(Pais.Code)) then begin
                    encontrado := true;
                    break;

                end;
                Assert.IsTrue(encontrado = true,
                    StrSubstNo(NoUEErr, Format(encontrado = true), Format(
    encontrado)));
            end;
        until Pais.Next() = 0;

        Assert.IsTrue(encontrado = true,
        StrSubstNo(CodPaisErr, Format(encontrado = true), Format(encontrado)));
    end
end
else
    Assert.IsTrue(false, CapturedErr);
```

Listing 4.16: Validacion SIITESTComprobarPais.

4.3.5. SIITESTFacturasEmitidasCobros

Respecto a esta codeunit los pasos a seguir respecto a la creación y registro de factura son los mismos que los descritos en el caso general pero, antes de capturarla hay que realizar otra serie de pasos ya que se trata de una factura especial conocida como RECC (régimen especial criterio de caja) *"este tipo de facturas se les realiza a empresas y autónomos que facturen menos de 2 millones de euros al año y, por lo tanto, no tienen que adelantar a Hacienda el impuesto de las facturas hasta que no sean cobradas"*[8] (RECC).

El paso intermedio entre la facturación y la captura es el cobro de la misma. Ya que el impuesto sobre el IVA en este tipo de facturas no se refleja en Business Central hasta que no se cobra la factura, hay que realizar el pago mediante la interfaz "Diario Cartera" dentro de Business Central. El pago de la factura genera en Business Central el impuesto de IVA de la factura con la fecha del cobro.

En la figura 4.10 se muestra la interfaz de “Diario Cartera” que será la que habrá que completar con el código que aparece más adelante para realizar el pago de dicha factura.

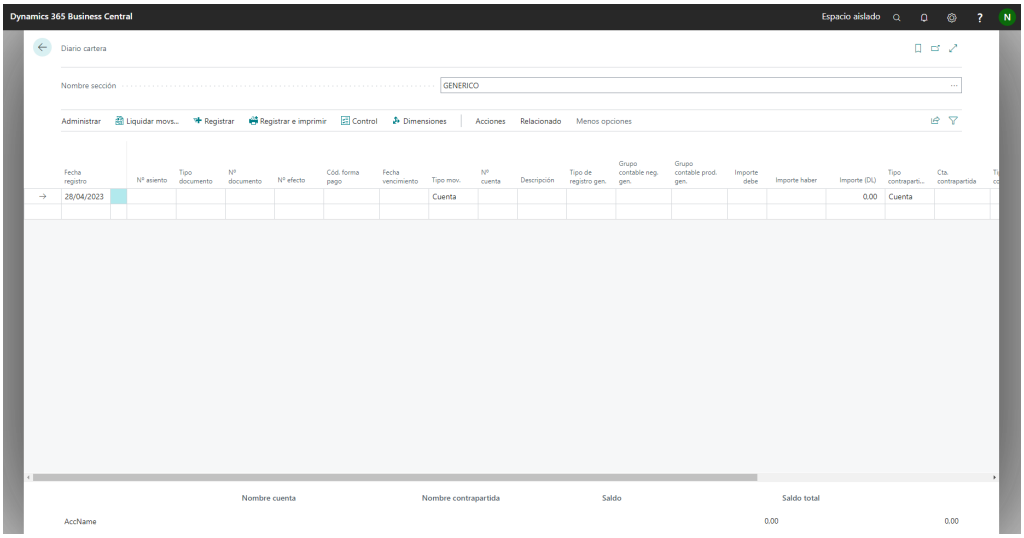


Figura 4.10: Interfaz creación diario cartera.

Para la creación de factura original hay que modificar, como mencionamos al principio del apartado, las siguientes líneas de código, ya que se trata de facturas especiales de régimen especial criterio de caja que se corresponde al valor “07” en el campo *”SII FE Special Regime Key”*[2]. Esto implica que se tenga que añadir más números de cuenta en la creación del grupo registro del IVA.
Se muestra el código modificado en el listing 4.17.

```
SIILibrary.CreateVATBusinessPostingGroup('CAJAR1S1', VATBusinessPostingGroup,
    VATBusinessPostingGroup."SII FE Special Regime Key"::"07",
    VATBusinessPostingGroup."SII FR Special Regime Key"::"07");
SIILibrary.CreateVATProductPostingGroup('VATTEST6', VATProductPostingGroup);
SIILibrary.CreateVATPostingSetUp(VATPostingSetup, 'CAJAR1S1', 'VATTEST6',
    VATPostingSetup."VAT Calculation Type"::"Normal VAT",
    VATPostingSetup."Unrealized VAT Type"::Percentage, '4770001', '4751001',
    '4720001', '4751001', '', '', 'VATTEST6', 21,
    VATPostingSetup."SII Classification"::"S1-Subject Not Exempt",
    VATPostingSetup."SII Intra-communit. Op. Type"::" ");
```

Listing 4.17: Modificación respecto a la factura original.

Una vez explicado este tipo de factura, los pasos a seguir son los siguientes: primero abrimos la página “Diario Cartera” con la función OpenView() y completamos los campos obligatorios como nos indica el estándar de la agencia tributaria. Continuamos añadiendo la fecha de hoy usando la función WorkDate(), el tipo de cuenta con toda la información correspondiente al cliente y, finalmente, se genera el documento con las líneas de pago y el precio.

Una vez lo tenemos se registra y es cuando podemos volver a la página de “Comunicación SII” y proceder a capturar la factura. Esta será la que usaremos para hacer las validaciones pertinentes.

Se muestra en el listing 4.18 la configuración del diario.

```
TestPageCarteraJournal.OpenView();
GenJournalLine.Init();
GenJournalLine.Validate("Journal Template Name", 'CARTERA');
GenJournalLine.Validate("Journal Batch Name", 'GENERICO');

GenJournalLine.Insert(true);
GenJournalLine.Validate(GenJournalLine."Posting Date", WorkDate());
GenJournalLine.Validate(GenJournalLine."Account Type",
    GenJournalLine."Account Type"::Customer);
GenJournalLine.Validate(GenJournalLine."Account No.", Customer."No.");
GenJournalLine.Validate(GenJournalLine."Document Type",
    GenJournalLine."Document Type"::Payment);

GenDocNo := LibraryUtility.GenerateRandomCode(
    GenJournalLine.FieldNo("Document No."), DATABASE::"Gen. Journal Line");
GenJournalLine.Validate(GenJournalLine."Document No.", GenDocNo);
GenJournalLine.Validate(GenJournalLine."Applies-to Doc. Type",
    GenJournalLine."Applies-to Doc. Type"::Invoice);

GenJournalLine.Validate(GenJournalLine."Applies-to Doc. No.", codeDocumentNo);
GenJournalLine.Modify(true);
GenJournalLine.Validate(GenJournalLine.Amount,
    LibraryRandom.RandDecInRange(-1200, -1, 0));
GenJournalLine.Validate(GenJournalLine."Bal. Account Type",
    GenJournalLine."Bal. Account Type"::"Bank Account");
GenJournalLine.Validate(GenJournalLine."Bal. Account No.", 'AHORROS');
GenJournalLine.Modify(true);

TestPageCarteraJournal.Post.Invoke();
```

Listing 4.18: Configuración diario cartera.

Se muestran un par de ejemplos de validaciones respecto a este tipo de libro registro. En este caso tenemos el campo "IDVersiónSII" y "NombreRazón", en ellos se comprueba haciendo uso del operador '<>' (distinto de), que significa que el campo es distinto de blanco, es decir, se encuentra relleno los valores que, como ya hemos mencionado anteriormente, se generan aleatoriamente.
Se muestran ejemplos de validaciones en el codigo del listing 4.19.

```
//IDVersionSII
Assert.IsTrue(SIISIISetup."API SII Version" <> '',
    StrSubstNo(SetupErr, SIISIISetup.FieldCaption("API SII Version"),
        Format(SIISIISetup."API SII Version" = ''), Format(SIISIISetup."API SII
    Version")));
//NombreRazon
Assert.IsTrue(SIICommunication."Dec. L.Name-F.Name Bus. Name" <> '',
    StrSubstNo(CampoVacioErr, SIICommunication.FieldCaption("Dec. L.Name-F.Name
    Bus. Name"),
        Format(SIICommunication."Dec. L.Name-F.Name Bus. Name" <> ''),
        Format(SIICommunication."Dec. L.Name-F.Name Bus. Name")));
```

Listing 4.19: Validaciones para SIITESTFacturasEmitidasCobros.

4.3.6. SIITESTPagosFacturasRecibidas

Para la codeunit final hacemos uso de todo lo aprendido anteriormente pero, en este caso, si antes realizamos facturas de venta, ahora lo que realizamos es una factura de compra. Los pasos a seguir son los mismos solo que, en este caso, las variables cambian, ya que en vez de tratar con un "Customer" lo hacemos con un "Vendor" en lo que, únicamente cambia es el coste del objeto (ya que antes era un valor positivo porque lo vendíamos y ahora es un valor negativo ya que estamos comprando el objeto).

Este **"Tipo de libro registro"** es como el correspondiente a la anterior codeunit, también se trata de un tipo de factura de régimen especial criterio de caja y genera el resultado contrario al anterior, ya que, en este caso, nuestra empresa es la que está adquiriendo un producto.

En cuanto al código se trata de una codeunit muy parecida a la anterior pero cambiando algunos valores, se muestran las líneas de código modificadas respecto el caso general de facturas de compra junto al código modificado de la codeunit anterior en el listing 4.20.

```
1  LibraryPurchase.CreateVendorWithVATRegNo(Vendor);
2  LibrarySIITestLibrary.SetUpVendorSII(Vendor,
3      Vendor."SII Resid. Country Tax ID Code"::"01-NIF (SPAIN)", false);
4  LibraryPurchase.CreatePurchaseLine(PurchaseLine, PurchaseHeader,
5      "Purchase Line Type".FromInteger(PurchaseLine.Type::"G/L Account".AsInteger
       ()),
6          codeGLAccount, 1);
7  PurchaseLine.Validate(PurchaseLine."Direct Unit Cost", LibraryRandom.
       RandDecInRange(0, 1000, 0));
8  codeDocumentNo := LibraryPurchase.PostPurchaseDocument(PurchaseHeader, true,
       true);
9
10 //Pagina Diario Cartera
11 GenJournalLine.Validate(GenJournalLine."Account Type",
12     GenJournalLine."Account Type"::Vendor);
13 GenJournalLine.Validate(GenJournalLine."Account No.", Vendor."No.");
14 GenJournalLine.Validate(GenJournalLine.Amount,
15     LibraryRandom.RandDecInRange(0, 1210, 0));
```

Listing 4.20: Modificaciones para SIITESTPagosFacturasRecibidas.

Finalmente tenemos otro par de ejemplos de validaciones que hemos realizado para este tipo de facturas. En este caso validamos los campos "NumSerieFacturaEmisor" y "FechaExpediciónFacturaEmisor", el primero de ellos sigue el mismo patrón que en la anterior codeunit, el segundo, al tratarse de una fecha, se valida usando el valor "0D" que representa una fecha indefinida o en blanco.

Se muestra en el listing 4.21 el código de las validaciones.

```
//NumSerieFacturaEmisor
Assert.IsTrue(SIICommunication."Issuer Invoice Number" <> '',
    StrSubstNo(CampoVacioErr, SIICommunication.FieldCaption("Issuer Invoice
    Number"),
    Format(SIICommunication."Issuer Invoice Number" <> ''),
    Format(SIICommunication."Issuer Invoice Number")));
//FechaExpedicionFacturaEmisor
Assert.IsTrue(SIICommunication."Invoice Issue Date" <> 0D,
    StrSubstNo(InvoiceIssueDateErr, SIICommunication."Invoice Issue Date" <> 0D
    ,
    Format(SIISIISetup."Inform Date of Issue".Names)));
```

Listing 4.21: Validaciones para SIITESTPagosFacturasRecibidas.

4.4. Verificación y validación

Respecto a la verificación y validación de las codeunits se han ido verificando de dos formas.

La primera de ellas ejecutando los tests automáticos una vez cargadas las codeunits en la página "Test Tool" dentro de Business Central y leyendo el mensaje de error que nos podía devolver cada test.

La figura 4.11 nos muestra un ejemplo de la validación por mensaje de error en test automático.

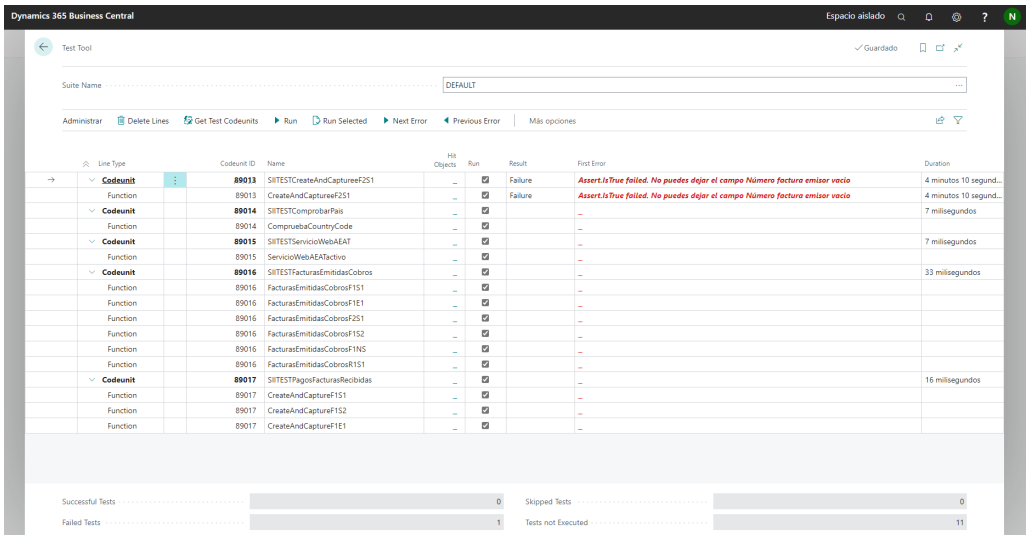


Figura 4.11: Mensaje de error test automático.

El segundo de ellos, el más importante, ha sido la ejecución del debugger una herramienta para seguir, paso a paso la ejecución de las líneas de código que va realizando la codeunit y siguiendo simultáneamente cómo van cambiando las variables al pasar por cada línea de código.

Este segundo método ha sido el más utilizado, ya que han habido muchos momentos a lo largo de la estancia que surgían errores y el mensaje de error era muy escaso. Es por ello que había que profundizar dentro de cada línea de ejecución para encontrar el fallo poniendo puntos de interrupción en las líneas de código donde se pensaba que podia estar el problema.

En la figura 4.12 vemos un ejemplo de la ejecución del debugger y a la izquierda se muestra un desplegable con todas las variables utilizadas en ese instante dado de la ejecución.

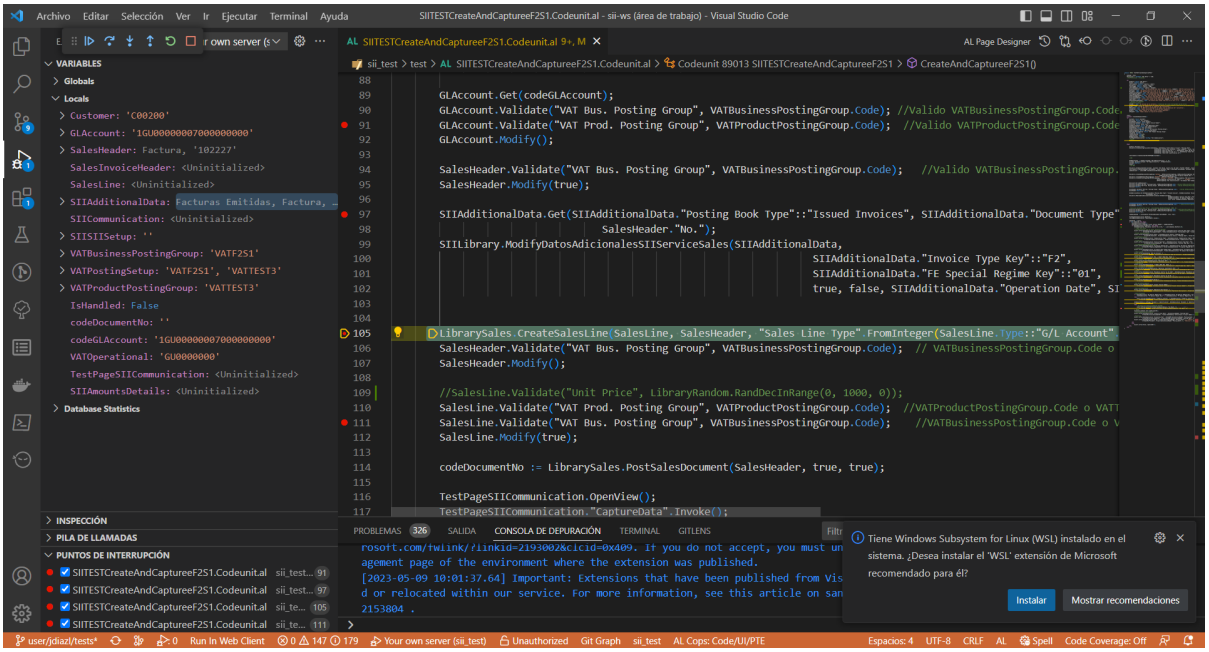


Figura 4.12: Interfaz Visual Studio Debugger.

Tras la explicación de los dos métodos usados a lo largo del proyecto, tenemos que, para la validación, se hacen pruebas usando datos que son consistentes y otras pruebas en las que hay datos inconsistentes y así los tests nos demuestran que sí que detectan esas inconsistencias.

Procedemos a realizar las respectivas validaciones para diferentes tipos de datos. Hemos de validar si los 4 tests funcionan correctamente, para ello procedemos a realizar las respectivas validaciones para diferentes tipos de datos, hemos testeado con datos correctos y con datos incorrectos. Se prueba a testear una factura del tipo *F2S1*, por ejemplo, pasándole un código incorrecto distinto como en el listing 4.22.

```
codeunit 89013 "SIITESTCreateAndCaptureeF2S1"
{
    Subtype = Test;
    Permissions = tabledata "VAT Entry" = rimd;

    var
        InvoiceTypeKeyErr: label 'La Clave tipo factura debe ser %1 y es %2.';

    [Test]
    procedure CreateAndCaptureeF2S1()
    var
```

```
12
13      begin
14          .
15          .
16          .
17      SIILibrary.ModifyDatosAdicionalesSIIServiceSales(SIIAdditionalData,
18          SIIAdditionalData."Invoice Type Key"::"F6",
19          SIIAdditionalData."FE Special Regime Key"::"01",
20          true, false, SIIAdditionalData."Operation Date",
21          SIIAdditionalData."FRECT Rectified Base",
22          SIIAdditionalData."FRECT Rectified Fee");
23          .
24          .
25          .
26      //TipoFactura
27      Assert.IsTrue(SIICommunication."Invoice Type Key" =
        SIICommunication."Invoice Type Key"::"F2",
28          StrSubstNo(InvoiceTypeKeyErr,
29          Format(SIICommunication."Invoice Type Key"::"F2"),
30          Format(SIICommunication."Invoice Type Key")));
31      end
32  }
```

Listing 4.22: Código para la validación de test.

En la figura 4.13 se observa como al realizar la validación aparece el mensaje de error.

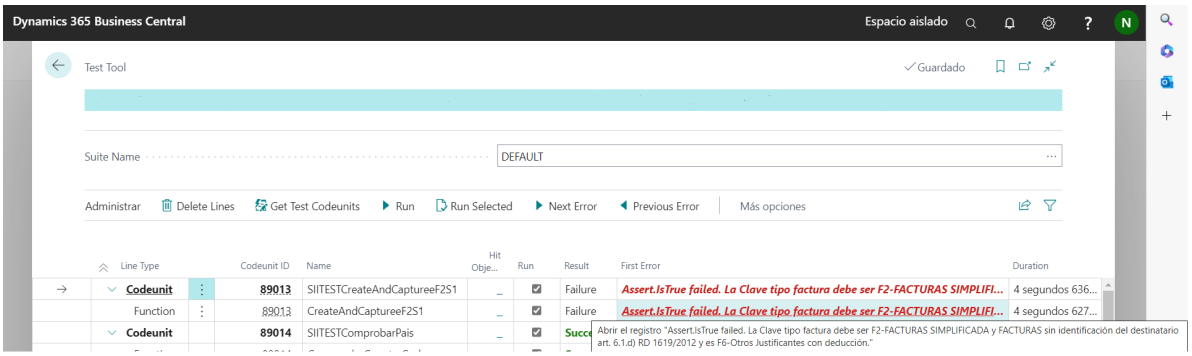


Figura 4.13: Test incorrecto.

Por el contrario cuando se testea un código correcto se muestra como en el listing 4.23.

```
1   //Cambios respecto al listing 4.18
2   SIILibrary.ModifyDatosAdicionalesSIIServiceSales(SIIAdditionalData,
3           SIIAdditionalData."Invoice Type Key"::"F2",
4           SIIAdditionalData."FE Special Regime Key"::"01",
5           true, false, SIIAdditionalData."Operation Date",
6           SIIAdditionalData."FRECT Rectified Base",
7           SIIAdditionalData."FRECT Rectified Fee");
8           .
9           .
10          .
11      //TipoFactura
12      Assert.IsTrue(SIICommunication."Invoice Type Key" =
        SIICommunication."Invoice Type Key"::"F2",
13          StrSubstNo(InvoiceTypeKeyErr,
```

```
14          Format(SIICommunication."Invoice Type Key"::"F2"),
15          Format(SIICommunication."Invoice Type Key")));
16      end
17  }
```

Listing 4.23: Modificación del código para la validación de test.

En la figura 4.14 aparece el mensaje de "Success" que indica que el test ha superado todas las comprobaciones.

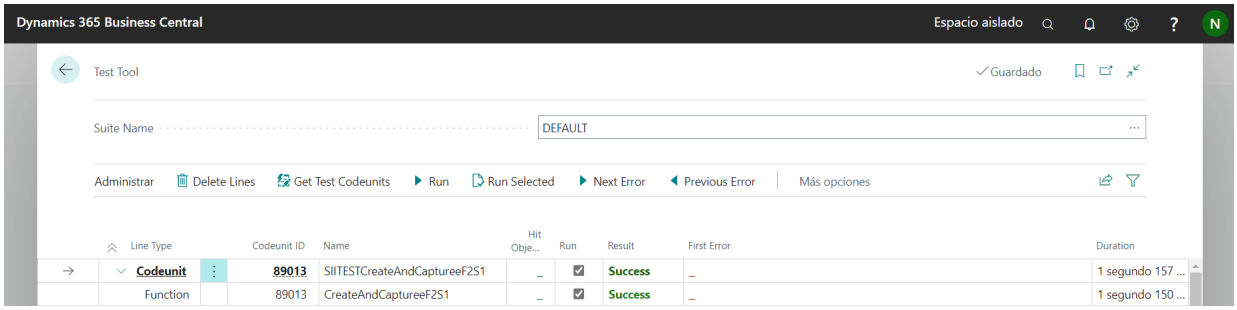


Figura 4.14: Test correcto.

Una vez ejecutados los tests se produce un rollback dentro de la base de datos, para que ningún dato de los que hemos creado aleatoriamente no interfiera con ningún cliente.

Finalmente se instaló la nueva versión de Business Central 21 y se comprobó que funcionaba todo de manera correcta y como se esperaba. Se muestra en la figura 4.15 la ejecución correcta de todas las codeunits.

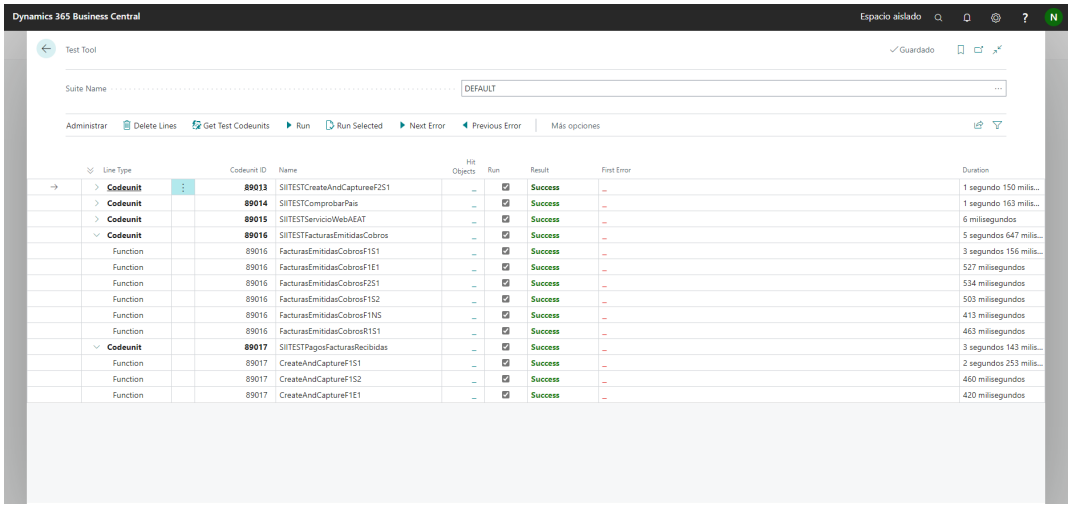


Figura 4.15: Validación de codeunits final.

Capítulo 5

Conclusiones

El testing hasta ahora, es un área dentro del desarrollo de proyectos en Business Central poco desarrollada, pero cada vez más empresas lo solicitan para futuras migraciones entre versiones de sus proyectos.

El proyecto realizado debería continuar haciendo todas las validaciones para todos los tipos de datos posibles, debido al escaso tiempo que teníamos en la estancia de prácticas y al gran desarrollo que estas codeunits conllevan, esto no entraba en la propuesta ni el alcance del proyecto. Nosotros nos hemos centrado en los tipos de factura que más se utilizan dentro de la empresa.

El TFG me ha ofrecido el aprendizaje en diferentes ámbitos de la informática y esto ha sido todo un reto para mí, me ha supuesto bastante complejo a nivel de concepto ya que se trataba de un lenguaje nuevo y un pensamiento diferente de programar al que veníamos aprendiendo a lo largo de los años.

La realización y descripción de este proyecto hace pensar en lo importante que es el tema de la fiabilidad en las empresas no solo en la protección de datos sino también en la verificación y validación de estos.

Aunque los ERP parece que sea un producto que sea para personas que se dedican al área de "Sistemas de la Información" tiene una parte tecnológica donde se necesitan personas que puedan dar soporte a los desarrolladores y las empresas.

Finalmente, decir que ha sido una experiencia para conocer el entorno profesional de primera mano, con un ambiente de trabajo insuperable para poder adquirir diferentes competencias y habilidades que seguro me servirán para un futuro. Y, por todo ello, quisiera dar las gracias a la empresa por la confianza depositada en mí para el desarrollo de este proyecto y a todo el personal de BC Puertos en Castellón por su ayuda en todo momento.

Bibliografía

[1] Agencia Tributaria. Agencia estatal de la agencia tributaria. https://sede.agenciatributaria.gob.es/. [Consulta: 02 de Mayo de 2023].

[2] Agencia Tributaria. Cuestiones especificas-régimen especial del criterio de caja (RECC). https://sede.agenciatributaria.gob.es/Sede/impuestos-tasas/iva/iva-libros-registro-iva-traves-aeat/preguntas-frecuentes/8-cuestiones-especificas-regimen-especial-caja.html. [Consulta: 02 de Mayo de 2023].

[3] Agencia Tributaria. Suministro inmediato de información (S.I.I.). https://www.agenciatributaria.es/static_files/AEAT/Contenidos_Comunes/La_Agencia_Tributaria/Modelos_y_formularios/Suministro_inmediato_informacion/FicherosSuministros/V_05/FAQs_v05.pdf. [Consulta: 12 de Abril de 2023].

[4] Ambit. Concepto y términos de IT. https://www.ambit-bst.com/blog/concepto-y-t%C3%A9rminos-de-it-para-otros-departamentos#:~:text=Qu%C3%A9%20es%20IT,la%20inform%C3%A1tica%20y%20la%20tecnolog%C3%ADa. [Consulta: 09 de Abril de 2023].

[5] Atlassian. Jira. https://www.atlassian.com/software/jira. [Consulta: 18 de Abril de 2023].

[6] Azure. Azure devops. https://azure.microsoft.com/es-es/products/devops/. [Consulta: 18 de Abril de 2023].

[7] Diccionario panhispánico del español jurídico. Definición autoridad portuaria. https://dpej.rae.es/lema/autoridad-portuaria. [Consulta: 18 de Abril de 2023].

[8] Infoautónomos. IVA con criterio de caja. https://www.infoautonomos.com/fiscalidad/el-iva-decaja/#:~:text=Con%20el%20r%C3%A9gimen%20del%20IVA%20con,%C3%A9ste%20r%C3%A9gimen.%20Te%20explicamos%20los%20detalles.&text=Con%20el%20r%C3%A9gimen%20del,Te%20explicamos%20los%20detalles.&text=r%C3%A9gimen%20del%20IVA%20con,%C3%A9ste%20r%C3%A9gimen.%20Te%20explicamos. [Consulta: 22 de Abril de 2023].

[9] Julia Martins. Diagrama de gantt: qué es y cómo crear uno con ejemplos. https://asana.com/es/resources/gantt-chart-basics. [Consulta: 20 de Abril de 2023].

[10] Lãberit Sistemas S.L. Conoce lãberit sistemas S.L. https://www.laberit.com/conocenos. [Consulta: 09 de Abril de 2023].

[11] Microsoft. Dynamics 365 business central. https://learn.microsoft.com/es-mx/dynamics365/business-central/. [Consulta: 12 de Abril de 2023].

[12] Microsoft. Microsoft project. https://www.microsoft.com/es-es/microsoft-365/project/project-management-software. [Consulta: 18 de Abril de 2023].

[13] Microsoft. Microsoft teams. https://www.microsoft.com/en-us/microsoft-teams/group-chat-software. [Consulta: 18 de Abril de 2023].

[14] Microsoft Dynamics. Definición de un ERP. https://dynamics.microsoft.com/es-mx/erp/define-erp/. [Consulta: 12 de Abril de 2023].

[15] Roger S.Pressman. *Modelos de proceso prescriptivo*, chapter 2.3, pages 33–43. Mccgraw-Hill, 2010.

[16] Visual Studio. Visual studio code. https://code.visualstudio.com/. [Consulta: 18 de Abril de 2023].

[17] Visual Studio Marketplace. Git graph. https://marketplace.visualstudio.com/items?itemName=mhutchie.git-graph. [Consulta: 18 de Abril de 2023].

[18] Visual Studio Marketplace. AL extension pack. https://marketplace.visualstudio.com/items?itemName=EdySpider.alextensionpack. [Consulta: 18 de Abril de 2023].

[19] Visual Studio Marketplace. AL language tools. https://marketplace.visualstudio.com/items?itemName=ClipDynamics.al-language-tools. [Consulta: 18 de Abril de 2023].

[20] Visual Studio Marketplace. AL object designer. https://marketplace.visualstudio.com/items?itemName=martonsagi.al-object-designer. [Consulta: 18 de Abril de 2023].