

Master profesional en Programación Avanzada en Python para BigData, Hacking y Machine Learning

Trabajo Fin de Master

Modelo Predictivo: Estimación de condiciones

climáticas que se darán en una región en los próximos meses

Pedro Jesús Navarro Blázquez

Abril, 2024

|  |  |
| --- | --- |
|  | ▷ Másteres con garantía de empleo | EIPosgrados.com |

Trabajo Fin de Master

Master profesional en Programación Avanzada en Python para BigData, Hacking y Machine Learning

Modelo Predictivo: Estimación de condiciones

climáticas que se darán en una región en los próximos meses.

|  |  |
| --- | --- |
| **Autor:** | Pedro Jesús Navarro Blázquez |
| **Tutor:** | Pablo Bermejo López |
| **Cotutor:** | Miguel Ángel Sánchez Cifo |

Abril, 2024

Declaración de Autoría

Yo, Pedro Jesús Navarro Blázquez con DNI 47449567-S, declaro que soy el único autor del trabajo fin de master titulado “Modelo Predictivo: Estimación de condiciones climáticas que se darán en una región en los próximos meses.” y que el citado trabajo no infringe las leyes en vigor sobre propiedad intelectual y que todo el material no original contenido en dicho trabajo está apropiadamente atribuido a sus legítimos autores.

Yellowknife, a 10 de abril de 2024

Fdo: ......

Resumen

El objetivo de este documento es explicar el desarrollo realizado para crear la aplicación de escritorio “FPA TOOL PN”, la cual está accesible en la tienda de Microsoft. Esta aplicación se utiliza para aplicar FPA y así medir el tamaño de un software. Al inicio de la memoria se ha dedicado una sección a explicar la estimación del tamaño funcional y en especial FPA. Seguidamente se ha explicado la metodología ágil Scrum, se ha comentado por qué se ha utilizado y como se ha aplicado. Así mismo, se han ido mostrando diferentes artefactos que se han ido obteniendo durante el desarrollo. Después se explican las distintas iteraciones realizadas, las cuales aportaban valor al cliente frecuentemente, hasta llegar la última iteración con la cual se ha conseguido el producto final. Para comprobar el funcionamiento correcto de la aplicación se ha realizado una estimación con la herramienta de un caso real. Para finalizar se ha realizado la conclusión en la cual se ha matizado la importancia que este proyecto ha aportado al desarrollador.

Agradecimientos

Quiero transmitir mi más sincero agradecimiento a todo el que me ha ayudado a lo largo de esta etapa.

En primer lugar, al tutor y cotutor, por su gran ayuda y confianza a lo largo de este largo y gratificante proyecto

En segundo lugar, a mi mujer Kassidy, mis padres y mi hermana, que me han apoyado incondicionalmente en todo momento y desde cualquier lugar. Este largo camino no lo hubiera logrado sin vosotros.

También quiero dar las gracias a todos los profesores y compañeros que he tenido durante este largo viaje.

“La vida es como una caja de bombones, nunca sabes lo que te va a tocar”-Forrest Gump.

Y a mí me habéis tocado vosotros, mil gracias.

Índice general

[Capítulo 1 Introducción 1](#_Toc77110868)

[1.1 Introducción 1](#_Toc77110869)

[Capítulo 2 Estado del Arte 3](#_Toc77110870)

[2.1 Introducción 3](#_Toc77110871)

[2.2 Estimación del tamaño del software 3](#_Toc77110872)

[Longitud 5](#_Toc77110873)

[Complejidad Ciclomática (CC) 8](#_Toc77110874)

[Tamaño Funcional 9](#_Toc77110875)

[2.3 Métodos de métricas orientados a los puntos función 12](#_Toc77110876)

[IFPUG 13](#_Toc77110877)

[MKII 18](#_Toc77110878)

[NESMA 19](#_Toc77110879)

[FISMA 19](#_Toc77110880)

[COSMIC-FFP 19](#_Toc77110881)

[Capítulo 3 Metodología 21](#_Toc77110882)

[3.1 Introducción 21](#_Toc77110883)

[3.2 Scrum 21](#_Toc77110884)

[Roles en Scrum 23](#_Toc77110885)

[Proceso de Scrum 23](#_Toc77110886)

[Artefactos de Scrum 24](#_Toc77110887)

[3.3 ¿Por qué Scrum? 25](#_Toc77110888)

[3.4 Adaptación de Scrum 26](#_Toc77110889)

[Inicialización del Product Backlog 28](#_Toc77110890)

[Transparencia del proceso con Jira 31](#_Toc77110891)

[Documentación del proceso 33](#_Toc77110892)

[Stack tecnológico 34](#_Toc77110893)

[Temporización del desarrollo 36](#_Toc77110894)

[Capítulo 4 Historial Sprints 39](#_Toc77110895)

[4.1 Introducción 39](#_Toc77110896)

[4.2 Artefactos 39](#_Toc77110897)

[4.3 Memoria de los sprints 42](#_Toc77110898)

[Sprint-1 (3-Semanas) 42](#_Toc77110899)

[Sprint-2 (2.5 Semanas) 43](#_Toc77110900)

[Sprint-3 (6 Semanas) 44](#_Toc77110901)

[Sprint-4 (2 semanas) 45](#_Toc77110902)

[Sprint-5 (1 semana) 46](#_Toc77110903)

[Sprint-6 (2 semanas) 47](#_Toc77110904)

[Sprint-7 (2 semanas) 48](#_Toc77110905)

[Sprint-8 (1 Semana) 49](#_Toc77110906)

[Sprint-9 (2 Semanas) 50](#_Toc77110907)

[Sprint-10 (1 Semana) 50](#_Toc77110908)

[Sprint-11 (1 Semana) 51](#_Toc77110909)

[Sprint-12 (1 Semana) 52](#_Toc77110910)

[Sprint-13 (1 Semana) 53](#_Toc77110911)

[Sprint-14 (1 Semana) 53](#_Toc77110912)

[Sprint-15 (1 Semana) 54](#_Toc77110913)

[Sprint-16 (1 Semana) 55](#_Toc77110914)

[Sprint-17 (1 Semana) 56](#_Toc77110915)

[Sprint-18 (2 Semanas) 56](#_Toc77110916)

[Sprint-19 (1 Semana) 57](#_Toc77110917)

[Capítulo 5 Herramienta 59](#_Toc77110918)

[5.1 Introducción 59](#_Toc77110919)

[5.2 Descripción 59](#_Toc77110920)

[5.3 Pantallas de la aplicación 60](#_Toc77110921)

[Capítulo 6 Pruebas 75](#_Toc77110922)

[6.1 Introducción 75](#_Toc77110923)

[6.2 Ejercicio práctico. 76](#_Toc77110924)

[Capítulo 7 Conclusiones 85](#_Toc77110925)

[Bibliografía 87](#_Toc77110926)

[Anexo I. Historias de Usuario. 93](#_Toc77110927)

[Anexo II. Despliegue en tienda Microsoft 99](#_Toc77110928)

[I. Registrarse como desarrollador en Microsoft, que conlleva un pago de 17 euros. 99](#_Toc77110929)

[II. Seleccionar .NET native tool chain en los ajustes del proyecto en Visual Studio 99](#_Toc77110930)

[III. Reservar un nombre de la aplicación. 100](#_Toc77110931)

[IV. Se genera un paquete desde la solución del proyecto. 100](#_Toc77110932)

[V. Comprobamos con el kit de certificación. 101](#_Toc77110933)

[VI. Realizamos la sumisión 102](#_Toc77110934)

[VII. Esperar a que los desarrolladores de Windows certifiquen la aplicación 102](#_Toc77110935)

[VIII. Error surgido durante el despliegue 103](#_Toc77110936)

Índice de figuras

[**Figura 2.1.** Secuencia de Estimaciones 4](#_Toc77110937)

[**Figura 2.2**. Triángulo del Hierro 4](#_Toc77110938)

[**Figura 2.3.** Notación métrica Halstead 6](#_Toc77110939)

[**Figura 2.4.** Cobertura de caminos (NAVARRO MARTÍNEZ, 2019) 8](#_Toc77110940)

[**Figura 2.5.** Técnicas creadas en base a la idea de Albrecht (Buglione, 2019) 10](#_Toc77110941)

[**Figura 2.6**. Secuencia Fibonacci (Pierce, 2020) 12](#_Toc77110942)

[**Figura 3.1.** Proceso de Scrum (Eudaimonia, 2020) 24](#_Toc77110943)

[**Figura 3.2.** Herramienta de comunicación Slack 28](#_Toc77110944)

[**Figura 3.3.** User Story Mapping 30](#_Toc77110945)

[**Figura 3.4.** Tablero / Sprint Backlog 32](#_Toc77110946)

[**Figura 3.5.** GitLab 34](#_Toc77110947)

[**Figura 3.6.** Aplicación Escritorio en UWP con C# y XAML 35](#_Toc77110948)

[**Figura 3.7.** Balsamiq Mockups 36](#_Toc77110949)

[**Figura 3.8:** Cronograma de Hitos (octubre-enero) 36](#_Toc77110950)

[**Figura 3.9:** Cronograma de Hitos (febrero-abril) 37](#_Toc77110951)

[**Figura 3.10:** Cronograma de Hitos (mayo-julio) 37](#_Toc77110952)

[**Figura 4.1.** Project Burn-Up chart 41](#_Toc77110953)

[**Figura 4.2.** Hola Mundo en WPF 43](#_Toc77110954)

[**Figura 4.3.** Repositorio del proyecto en GitLab 43](#_Toc77110955)

[**Figura 4.4.** Prototipado inicial con Word 45](#_Toc77110956)

[**Figura 4.5.** Pantalla sin ajustarse al prototipo 46](#_Toc77110957)

[**Figura 4.6**. Popups 47](#_Toc77110958)

[**Figura 4.7.** Archivo XML inicial 49](#_Toc77110959)

[**Figura 4.8.** Canvas con aplicaciones arrastrables 51](#_Toc77110960)

[**Figura 4.9**. Product Backlog Final 58](#_Toc77110961)

[**Figura 5.1.** Pantalla Principal 62](#_Toc77110962)

[**Figura 5.2.** Crear FUR 63](#_Toc77110963)

[**Figura 5.3.** Mostrar descripción larga de FUR 63](#_Toc77110964)

[**Figura 5.4.** Crear aplicación principal 64](#_Toc77110965)

[**Figura 5.5**. Mensaje aplicación principal ya existe. 65](#_Toc77110966)

[**Figura 5.6.** Grid con dos aplicaciones 65](#_Toc77110967)

[**Figura 5.7**. Crear función de datos 67](#_Toc77110968)

[**Figura 5.8.** Función de datos añadida a la aplicación 67](#_Toc77110969)

[**Figura 5.9.** Edición o eliminación de función de datos seleccionada 68](#_Toc77110970)

[**Figura 5.10.** Verificar eliminación de función de datos 69](#_Toc77110971)

[**Figura 5.11:** Listado de Funciones de datos de FUR actual 69](#_Toc77110972)

[**Figura 5.12**. Crear Función Transaccional 71](#_Toc77110973)

[**Figura 5.13.** Extra Det 72](#_Toc77110974)

[**Figura 5.14:** Editar o Borrar Función Transaccional 72](#_Toc77110975)

[**Figura 5.15.** Borrar Función Transaccional 73](#_Toc77110976)

[**Figura 5.16.** Editar Función Transaccional 73](#_Toc77110977)

[**Figura 5.17.** Lista Funciones Transaccionales 74](#_Toc77110978)

[**Figura 6.1.** Ejercicio 12 manualmente 77](#_Toc77110979)

[**Figura 6.2.** Ejercicio 12 elementos creados en la herramienta 79](#_Toc77110980)

[**Figura 6.3.** Ejercicio 12 Listado de Funciones de Datos 79](#_Toc77110981)

[**Figura 6.4.** Ejercicio 12 Listado de Funciones Transaccionales 80](#_Toc77110982)

[**Figura 7.1.** Registrarse como desarrollador 99](#_Toc77110983)

[**Figura 7.2.** Seleccionar .NET native tool chain 100](#_Toc77110984)

[**Figura 7.3.** Reservar nombre 100](#_Toc77110985)

[**Figura 7.4.** Generar paquete .msix 101](#_Toc77110986)

[**Figura 7.5**. Revisión del paquete con kit Certification 101](#_Toc77110987)

[**Figura 7.6.** Sumisión en Microsoft Partner Center 102](#_Toc77110988)

[**Figura 7.7**. Desarrolladores de Microsoft deben verificar la aplicación 103](#_Toc77110989)

[**Figura 7.8**. Error librería .msi 103](#_Toc77110990)

Índice de tablas

[**Tabla 3.1.** Herramientas y frameworks usados. 38](#_Toc77110991)

[**Tabla 4.1.** Product Backlog inicial 40](#_Toc77110992)

[**Tabla 6.1.** Matriz de Complejidad para ILF y EIF 81](#_Toc77110993)

[**Tabla 6.2**. Traducción de complejidad del ILF 81](#_Toc77110994)

[**Tabla 6.3**. Traducción de complejidad del EIF 82](#_Toc77110995)

[**Tabla 6.4.** Matriz de Complejidad para EI 82](#_Toc77110996)

[**Tabla 6.5.** Matriz de Complejidad para EO y EQ 82](#_Toc77110997)

[**Tabla 6.6**. Traducción de complejidad del EI y EQ 82](#_Toc77110998)

[**Tabla 6.7**. Traducción de complejidad del EO 82](#_Toc77110999)

[**Tabla 7.1.** Historias de Usuario 93](#_Toc77111000)

# Introducción

## Introducción

Este documento tiene como objetivo principal explicar el desarrollo de la herramienta “FPA TOOL PN” que se encuentra alojada en la tienda de Windows.

En el primer capítulo, que es en el que nos encontramos en estos momentos, se realiza una introducción al documento indicando las diferentes partes de este.

En el segundo capítulo se profundizará en los conceptos más relevantes respecto a la estimación del tamaño del software, ya que esa es la función de la herramienta desarrollada. Se describirán los distintos tipos de medición que se han ido utilizado a lo largo de los años y que muchos de ellos aún siguen utilizándose. También se estudiarán las distintas métricas más relevantes que hay en la actualidad. En la última parte del capítulo se explorarán los diferentes métodos de medición y se hará especial énfasis en la metodología FPA, sobre la que está basada el proyecto que se ha desarrollado y por consecuente el producto obtenido.

En el tercer capítulo nos vamos a centrar en la metodología que se ha utilizado para desarrollar el proyecto. El marco de trabajo escogido es Scrum, inicialmente lo describimos para posteriormente poderlo aplicar. En nuestra metodología hay que destacar los roles de Scrum y el proceso de Scrum, por lo que se exploraran en profundidad. Una vez se ha definido la metodología, se explica el motivo porque ha sido escogida. Finalmente se explica cómo se ha aplicado esta metodología en el proyecto y cuales han sido las herramientas utilizadas para ello.

En el cuarto capítulo nos centramos en el historial de Sprints que se han ido realizando a lo largo del desarrollo del proyecto. Al principio del capítulo se analiza el product backlog con el que se va a iniciar las iteraciones. Seguidamente se analiza el project burn-up chart que muestra la relación entre la cantidad de elementos que han sido finalizados correctamente del product backlog y los que no se han completado. El resto del capítulo explica cada uno de los 19 Sprints realizados durante el desarrollo del proyecto. Cada sprint es explicado en 5 apartados; objetivo, planificación, desarrollo, revisión y retrospectiva.

En el quinto capítulo se explica la herramienta desarrollada durante el proyecto. Inicialmente se realiza una pequeña descripción de la herramienta y su funcionalidad. El resto del capítulo está centrado en explicar detalladamente las interfaces que nos ofrece la herramienta, es decir, enumerar todos los componentes que hay en cada pantalla y explicar su funcionalidad.

En el sexto capítulo nos centramos en las pruebas realizadas al producto obtenido, es decir, a la herramienta. Para ello se va a hacer uso de un caso práctico cogido de la asignatura Gestión de Proyectos Software. Se mostrará el enunciado del caso propuesto y su correspondiente resolución tanto manualmente como con la herramienta, demostrándose así la funcionalidad de esta.

El último capítulo, que es el séptimo, aloja todas las referencias bibliográficas utilizadas para el desarrollo de este documento.

Para finalizar, el documento tiene dos anexos con información adicional para los lectores más curiosos. En el primer anexo se aloja una tabla que contiene todas las historias de usuario creadas durante el desarrollo del proyecto, con su descripción, tamaño en puntos función y los criterios de aceptación. El segundo anexo explica los pasos que hay que realizar para poder desplegar una aplicación de escritorio en la tienda de Windows.

# Estado del Arte

## Introducción

En este capítulo se va a realizar un análisis de tipo documental respecto a la estimación del tamaño del software. Se explican las 3 categorías en las que se agrupa la medición del tamaño del software junto con las diferentes métricas más relevantes hasta la actualidad. Posteriormente, se explorarán diferentes métodos de medición y se hará especial énfasis en la metodología FPA, sobre la que está basada el proyecto que se ha desarrollado y el producto obtenido.

## Estimación del tamaño del software

La estimación del tamaño del software es una actividad de ingeniería del software que nos permite estimar el tamaño de un componente, para así poder implementar otras actividades de gestión del proyecto (Borade, 2018). La estimación del tamaño es una parte esencial de la gestión de proyectos software. Esta permite a los encargados del proyecto predecir mejor el tiempo que será necesario, tanto para la creación del proyecto, como para los tiempos de las distintas entregas (rawat & Rai, 2019).

En la actualidad el fracaso de los proyectos software es un principal reto al que nos enfrentamos, y una de las principales razones es la ineptitud de estimar un proyecto software de manera precisa. El momento ideal para realizar la estimación del tamaño software es durante el periodo de recogida de requisitos, ya que es muy importante realizar una buena estimación al inicio del desarrollo del proyecto. Cuando se estima el tamaño del software, indistintamente del método que se utilice se puede estimar el esfuerzo en personal humano. Si tenemos el esfuerzo en personal humano se puede estimar el tiempo necesario para el desarrollo del proyecto (Borade, 2018). Una vez tenemos el personal humano y el tiempo se puede estimar el coste del proyecto. En definitiva, para realizar una buena estimación del software debemos de seguir la secuencia que se muestra en la figura: **Figura 2.1**.

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente

**Figura 2.1.** Secuencia de Estimaciones

Durante la estimación de los proyectos software, se debe de encontrar un equilibrio entre el tiempo, alcance y coste para garantizar una calidad determinada. Estas tres variables conforman los tres pilares esenciales para determinar los objetivos y retos del proyecto, son conocidos como el triángulo del hierro y su representación la podemos ver en la figura: **Figura 2.2**. El alcance son las tareas o los requisitos que se tienen que realizar, el tiempo es la duración del proyecto y el coste la cantidad de recursos que se dedicaran al proyecto.

En un proyecto la calidad debe estar garantizada, cualquier modificación de una de estas variables implica la modificación de las otras dos para poder seguir garantizando esa calidad (Albaladejo, 2021).

Diagrama

Descripción generada automáticamente

**Figura 2.2**. Triángulo del Hierro

¿Por qué es necesaria la estimación del software?

El tamaño del proyecto influye sobre la estimación del software, la precisión cuando se estiman proyectos pequeños no es muy importante, mientras que en los proyectos grandes sí lo es. A razón del crecimiento del proyecto, la estimación del software debe de ser más precisa ya que el impacto será mayor.

En el pasado se realizaron diferentes investigaciones respecto a la estimación del coste de modelación y todas ellas llegaron a la misma conclusión: “Era muy difícil predecir el coste preciso de un software desarrollado cuando el software crecía en tamaño e importancia, ya que también crecía en complejidad” (P.K.Suri & Ranjan, 2012).

La medición del tamaño de software se agrupa en 3 categorías, dependiendo del atributo de software que se mide. Estas tres categorías son; longitud, complejidad y funcionalidad, que serán explicadas más detalladamente en los siguientes apartados (Bermejo, 2020).

### Longitud

La longitud se basa en la medición del código fuente, es decir, en el tamaño físico del software escrito (Bermejo, 2020). Hay diferentes métricas para medir el tamaño mediante longitud, a continuación se va a explicar las más relevantes.

#### Líneas de código (LOC)

Las líneas de código fueron la primera unidad del tamaño del software utilizada. Es una medida de estimación del tamaño bastante simple que depende del lenguaje de programación utilizado. Debido a que utiliza las líneas de código, no es útil en etapas tempranas del proyecto donde hay poco código escrito.

La idea principal de esta métrica es: “cuantas más líneas de código, más trabajo que tengo que hacer” (Weiss, 2002). Aun así hay que tener en cuenta que no todas las líneas de código tienen la misma lógica, es decir, “una línea de código requiere mayor esfuerzo que otra” (Weiss, 2002).

Uno de los problemas principales de esta métrica es no tener un acuerdo preciso sobre que líneas de código se cuentan y cuáles no. Para poder diferenciar entre las líneas de código que se tienen en cuenta y la complejidad de estás se sigue una serie de decisiones que cada organización decidirá como hacer uso de ellas. Las decisiones que deben de tomar las organizaciones son:

– Decidir si se tiene en cuenta los comentarios de código, ya que dependiendo del programador será mayor o menor la cantidad de estos.

– Decidir si se tiene en cuenta el código que no se ejecuta como son los encabezamientos, variables, etc.

– Decidir si el código que el cliente no recibirá, como es el caso de las pruebas, se deben de tener en cuenta o no.

– Decidir sobre las declaraciones de condiciones, como es el caso de la cláusula if-else, hay que decidir si se cuenta como una sola unidad o como dos.

Una vez hemos profundizado algunos aspectos de esta métrica podríamos definir una línea de código como; “Una línea de código es cualquier línea de un programa de texto que no es un comentario o una línea en blanco, a pesar del número de declaraciones o fragmentos de declaraciones en una línea. Esto específicamente incluye todas las líneas que contienen encabezados en el programa, declaraciones, y declaraciones ejecutables y no ejecutables” (Weiss, 2002).

#### Métricas de Halstead

Una forma más precisa de medir el tamaño software fue propuesta por Halstead. Las métricas de Halstead son un conjunto de métricas obtenidas a razón de varias medidas. Estas medidas son aplicadas sobre el código de los programas.

(Sandeep, 2020) (Sáez Vacas, 2008) Para estudiar más en profundidad esta métrica, debemos de fijarnos en la notación mostrada en la figura: **Figura 2.3**.

Texto

Descripción generada automáticamente

**Figura 2.3.** Notación métrica Halstead

Las medidas utilizadas en la métrica son:

– Longitud: Número total de ocurrencias del operando y número total de ocurrencias del operador, o lo que es lo mismo: .

– Vocabulario: Número total de operadores únicos y ocurrencias únicas de operandos únicos, o lo que es lo mismo: .

– Volumen: Medida que representa el tamaño en bits, es una medida más precisa ya que tiene en cuenta tanto la longitud como el vocabulario, se representa como .

– Nivel: Representa el nivel de detalle con que ha sido codificado el programa. Se supone que cuanto más código se utiliza para representar una función, de más bajo nivel es. Por lo tanto, cuanto más alto nivel es un lenguaje, menos esfuerzo será necesario para desarrollar un programa con ese lenguaje. Se representa como.

– Esfuerzo: mediante el volumen y la longitud podemos obtener el esfuerzo necesario para una porción de código. Se representa como: . (Sandeep, 2020) (Sáez Vacas, 2008)

#### Estimación previa

Si hacemos caso a la secuencia **Figura 2.1**, para poder llegar a calcular el coste primero debemos saber el tamaño. Así mismo, para poder realizar una estimación debemos de tener el código ya escrito, por lo que la gran cuestión es ¿Cómo podemos planificar el esfuerzo, tiempo y coste en etapas iniciales de un proyecto software?

Se puede realizar de dos formas principalmente:

1. Analogía.

Está basada en la información respecto a proyectos anteriores y ofrece tres maneras diferentes de predecir la longitud del software:

1. Ecuación de regresión: función multivariada que trata de ajustar los parámetros dependiendo de las variables utilizadas.
2. Mediante CBR (Case-Based-Reasoning): Se comparan atributos de proyectos anteriores que se parecen. Estos atributos nos permiten hacer una predicción con cada proyecto dependiendo de la importancia de las similitudes encontradas.
3. Aprendizaje automático: Técnicas automáticas que usan como referencia un conjunto de proyectos y se hace una predicción automatizada con los atributos del nuevo proyecto (Bermejo, 2020).
4. Método Delphi

Este método lo ponen en práctica un grupo de personas experimentadas que realizan una estimación anónima. Una vez están terminadas las estimaciones de cada experto, estas se comparten y en caso de no haberse llegado a un consenso se repite el mismo proceso hasta que se llega a un consenso (Estimation Techniques - Wideband Delphi, 2021).

### Complejidad Ciclomática (CC)

La complejidad Ciclomática es una métrica relacionada con la calidad, se utiliza para medir la complejidad del software. Está métrica fue desarrollada por Thomas J. McCabe y se caracteriza por indicar el número de caminos independientes que puede tomar un programa durante su ejecución (Gutierrez Ramirez, 2011).

La complejidad Ciclomática es un tipo de pruebas de caja blanca o lo que es lo mismo pruebas que tienen en cuenta el mecanismo interno de un sistema o un componente. Está basada en una representación de flujo de control de programa, que consiste en representar un programa como gráfico con sus respectivos nodos y bordes como podemos apreciar en la figura: **Figura 2.4**. Esta figura muestra con números las declaraciones, con letras las condiciones y con colores las regiones.

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza baja

**Figura 2.4.** Cobertura de caminos (NAVARRO MARTÍNEZ, 2019)

Cuando se calcula la complejidad Ciclomática se deben de seguir una serie de pasos:

– Construir el grafo con sus respectivos nodos y flechas a partir del código.

– Identificar los caminos independientes que nos ofrece el grafo.

– Cálculo de la complejidad Ciclomática a partir de los datos previos.

– Diseñar los casos de prueba para los cálculos previos.

La complejidad Ciclomática se puede calcular a partir de tres formulas diferentes, si las tres nos ofrecen el mismo resultado tendremos gran probabilidad de haber realizado el cálculo correctamente. Estas tres formulas son:

– , donde E = flechas y N = nodos.

–

–

Para finalizar esta métrica, debemos de matizar que cuanto mayor número de caminos, mayor será el número de pruebas de caja blanca para poder garantizar una calidad determinada (NAVARRO MARTÍNEZ, 2019).

### Tamaño Funcional

El tamaño funcional representa los requisitos funcionales que son ofrecidos al usuario. Es ajeno a cualquier aspecto técnico o decisión en la implementación, ya que se centra en los requisitos funcionales. Al focalizarse en los requisitos funcionales nos permite realizar un cálculo en las fases iniciales del proyecto, aun así no tiene en cuenta los requisitos no funcionales, es decir, la calidad de nuestro proyecto (Bermejo, 2020).

En los siguientes subapartados se va a explorar los diferentes métodos para estimar el tamaño funcional de un software.

#### Puntos Función (FP)

Son una unidad de medida utilizada para el cálculo del tamaño funcional. Para poder hacer uso de esta medida se necesita una descripción lo más detallada posible de los requisitos funcionales.

Los puntos función se calculan mediante las entradas externas, las salidas y peticiones externas, y por los archivos lógicos (Bermejo, 2020). Hay varias metodologías para realizar el cálculo de puntos función, aunque las más importantes son MKII, FPA, COSMIC, NESMA, FISMA e IFPUG, en la figura: **Figura 2.5** podemos apreciar cómo se han desarrollado las diferentes variantes a partir de la idea de Albrecht que principalmente se basaba en que su método fuera independiente de la tecnología utilizada (Buglione, 2019). Estas metodologías serán exploradas más detalladamente a lo largo de este documento.

Diagrama

Descripción generada automáticamente

**Figura 2.5.** Técnicas creadas en base a la idea de Albrecht (Buglione, 2019)

#### Puntos Función completo (FFP)

Son una medida del tamaño funcional que surgió en 1997, al igual que los FP se trata desde una perspectiva del usuario y obviando las características técnicas del software. Esta medida se diseñó especialmente para los softwares de tiempo real y los empotrados. FFP se centró en añadir la forma de contar procesos y datos que cambian en tiempo real (Abran, St-Pierre, maya, & Desharnais, 1998).

#### Puntos Caso de Uso

Son una medida del tamaño del software que fue desarrollada en 1993 por Gustav Kamer, bajo la supervisión de Ivar Jacobson. Es una metodología parecida a los puntos función, se trata de “una cuenta no ajustada de Puntos de Caso de Uso (UAUCP) asignando una complejidad a los actores y a los casos de uso” (Gómez, 2013).

Esta metodología encaja muy rápido en las empresas que ya usan la técnica de casos de uso, por lo que incorporarla sería muy sencillo. Tiene una serie de pasos que se pueden seguir con mucha facilidad para poder realizar el cálculo.

#### Puntos Objeto

Medida basada en las diferentes pantallas e informes que tiene el software, también se le conoce como puntos de aplicación. El hecho de usar las pantallas e informes como referencia para los cálculos no nos permite realizar un cálculo en etapas tempranas del proyecto (Medición del software, 2011).

#### Puntos Característica

Medida que tiene como base principal los puntos función, ya que son una extensión de estos. Hacen uso de los mismos elementos que en FP, pero con el añadido de algoritmos que realizan operaciones complejas (Bermejo, 2020).

#### Puntos de Historia

Los puntos de historia son un esfuerzo subjetivo que estiman los desarrolladores, les permite ponerse de acuerdo respecto a la complejidad, esfuerzo y riesgo que tiene una tarea. Los puntos de historia permiten acordar “como de grande es una tarea” (Casillas, 2019). Este es el motivo principal por lo que esta estimación se utiliza en metodología ágil. En la metodología ágil el equipo selecciona una serie de requisitos y se toma como referencia un requisito que todo el equipo tenga claro su tamaño y se le asigna una complejidad X. Cada iteración o sprint el equipo selecciona una serie de historias de usuario a implementar y estima el tamaño en puntos de historia, tomando como referencia X (Casillas, 2019).

No se trata de una medida útil para planificar el tamaño del desarrollo del software ni para la planificación del cronograma del proyecto software. Sino que, es útil para que el equipo pueda tener una gestión autónoma durante las iteraciones (Bermejo, 2020).

La referencia X la estipula el equipo de trabajo, por lo tanto no es comparable entre distintos equipos de trabajo, ya que la estimación es subjetiva. Para la estimación de las tareas, muchos equipos de trabajo hacen uso de la secuencia Fibonacci, esta puede estar representada como una espiral, véase en la figura: **Figura 2.6**.

Imagen que contiene Diagrama

Descripción generada automáticamente

**Figura 2.6**. Secuencia Fibonacci (Pierce, 2020)

Uno de los principales problemas que tienen los equipos de desarrollo nuevos, es la diferencia entre puntos de historia y tiempo. En internet hay muchos ejemplos que explican este problema, en nuestro caso se va a explorar uno que ha llamado la atención al equipo, se trata de la metáfora del viaje. En esta una familia va desde un punto a otro en coche para disfrutar las vacaciones y los hijos hacen la típica pregunta “¿Mamá, Papá, hemos llegado ya?” (Lundholm & Brodd, 2014). El tiempo entre dos distancias que se recorren con un coche nunca va a ser el mismo, ya que pueden pasar muchas cosas. En cambio sí que puedes comparar si esta distancia es mayor o menor que la distancia entre otras dos ciudades. Por ejemplo, si se conoce la distancia entre Londres y Roma, puedes estimar aproximadamente la distancia a Paris. Los humanos somos capaces de estimar bastante bien en términos relativos, pero no somos buenos para la estimación absoluta.

Es por eso por lo que utilizamos historias de usuario de referencia cuando hacemos estimaciones.

## Métodos de métricas orientados a los puntos función

A finales de la década de los 70´s Allan J. Albrecht propuso una medida que se basaba en medir el software a partir de funcionalidades entregadas al usuario en vez de a partir de aspectos técnicos. Pretendía realizar valoraciones independientemente de la tecnología utilizada.

En octubre de 1979, en Monterey (California), Albrecht propuso oficialmente el concepto puntos función como una nueva métrica de las aplicaciones del software. Esta teoría estaba basada en estudiar 5 componentes o características principales del software: las entradas, las salidas, las consultas, los ficheros lógicos internos y los ficheros lógicos externos (Anzola, 2011).

Hay varias metodologías para realizar el cálculo de puntos función, aunque las más importantes son las mostradas en la: **Figura 2.5**. Todas las metodologías parten de la idea de Albrecht.

### IFPUG

En 1986 se creó IFPUG (Agrupación Internacional de Usuarios de Puntos Función), que es una organización no lucrativa que se centró en dar soporte a la nueva técnica creada por Albrecht y con ello poder promocionar su uso.

IFPUG decidió llamar a la metodología FPA (Function Point Analysis). La aplicación que se va a desarrollar a lo largo de este documento trata sobre cálculos realizados con esta metodología para la obtención de los puntos función de un software.

Para poder realizar los cálculos en base a unas reglas y criterios determinados, en 1990 IFPUG creó la normativa CPM (Function Point Counting Practices Manual), que básicamente es un manual con las diferentes prácticas a seguir para realizar la cuenta en base al análisis de los puntos función (Lorenzo, 2021).

A lo largo de los años, se han ido creando diferentes versiones del manual CPM como podemos apreciar en la figura: **Figura 2.5**, aun así desde el año 2010 no se ha realizado ninguna actualización de la que es la última versión la 4.3.1. Esta versión cumple con el estándar ISO/IEC 14143-1:2007 para la medición del tamaño funcional (International Function Point Users Group, 2009).

CPM nos ofrece 6 fases para calcular el tamaño funcional del software (The International Function Point Users Group, 2005). Estas fases son:

1. **Reunir toda la documentación posible.**

Se debe reunir toda la documentación que describe los requisitos funcionales del usuario. Según IFPUG un usuario se define como: “cualquier persona o cosa que se comunique o interactúe con el software” (Bermejo, 2021).

IFPUG divide los requisitos en tres categorías diferentes según el nivel de detalle de estos:

1. Requisitos de usuario iniciales: requisitos previos a la sesión entre el usuario y los desarrolladores software.
2. Requisitos técnicos: requisitos que los desarrolladores software crean tras hablar con el cliente. Estos requisitos son definidos con terminología técnica que el usuario normalmente no conoce.
3. Requisitos funcionales finales: requisitos que son el resultado de la unión de los dos anteriores y que debe garantizar diferentes características; es necesaria una terminología que se entienda tanto por los desarrolladores como por el cliente, proporciona descripción integra de todos los requisitos del usuario y algunas otras más características.
4. **Determinar el alcance y límites de la cuenta, e identificar los requisitos funcionales del usuario.**

El alcance define la funcionalidad que será incluida en una cuenta particular de puntos función, es decir, el conjunto de requisitos funcionales que serán incluidos en la cuenta de FPs. Los requisitos se seleccionan según el tipo de cuenta:

1. Cuenta de puntos función del desarrollo: incluye todas las funciones afectadas por las actividades del proyecto.
2. Cuenta de puntos función de mejora: incluye todas las funciones añadidas, modificadas y borradas.
3. Cuenta de puntos función de la aplicación: incluye todas las funciones entregadas o las funciones usadas por el usuario.

El límite indica la frontera entre el software y el usuario. La frontera de la aplicación:

– Define qué es externo a la aplicación.

– Actúa como una membrana en la que los datos cruzan mediante transacciones hacia dentro y fuera.

– Los datos en el interior de la aplicación son mantenidos por el sistema, mientras que los del exterior solamente están referenciados.

– Es independiente de consideraciones técnicas, depende de la vista del usuario.

1. **Medir las funciones de datos.**

Las funciones de datos representan la funcionalidad proporcionada al usuario para satisfacer los requisitos de almacenamiento interno o externo. Hay dos tipos te funciones de datos:

1. Archivo Lógico Interno (ILF): “es un grupo identificable por el usuario de datos relacionados lógicamente o información de control mantenida dentro de los límites de la aplicación” (The International Function Point Users Group, 2005). El objetivo de un ILF es mantener los datos mantenidos a través de uno o más procesos elementales de la aplicación que se está contando.
2. Archivo de Interfaz Externo (EIF): “es un grupo identificable por el usuario de datos relacionados lógicamente o información de control referenciada por la aplicación, pero mantenida dentro de los límites de otra aplicación” (The International Function Point Users Group, 2005). El objetivo de un EIF es contener datos referenciados a través de uno o más procesos elementales dentro del límite de la aplicación contada. Un EIF que está contado para la aplicación principal debe estar en un ILF de otra aplicación.

Para el cálculo de funciones de datos no es suficiente con saber la cantidad de funciones de datos que tenemos, sino que hay que saber la complejidad de estas. La complejidad de un ILF o EIF se mide en relación con el número de DETs y RETs.

Un **DET (Data Element Type)** es un “campo único y reconocible por el usuario” (The International Function Point Users Group, 2005). Cuando se cuentan los DET hay una serie de reglas:

1. Contamos un DET para cada campo no repetido, que sea mantenido o recuperado desde un ILF o un EIF a través de la ejecución de un proceso.
2. Cuando dos aplicaciones mantienen o referencian el mismo ILF/EIF, pero cada uno mantiene o referencia DET separados, se cuenta solo los DET usados por cada aplicación.
3. Contamos un DET por cada dato requerido por el usuario para establecer una relación con otro ILF o EIF.

Un **RET (Record Element Type)** es un “grupo de DET reconocible por el usuario y dentro de un ILF o un EIF” (The International Function Point Users Group, 2005). Cuando contamos RET hay una serie de reglas:

1. Contamos un RET para cada subgrupo opcional u obligatorio del ILF o EIF.
2. Si no hay subgrupos contamos la función de datos como un RET.

Cuando sabemos el número de funciones de datos, y los DET y RET de cada función de datos utilizamos las tablas de complejidad y conversión para averiguar cuántos FPs corresponden a la función. Estas tablas se explicarán a lo largo del documento.

1. **Medir las funciones transaccionales.**

Una función transaccional representa la funcionalidad proporcionada al usuario para el procesamiento de datos por una aplicación. Hay tres tipos de funciones de datos:

1. EI (External Input): “Es un proceso elemental que procesa datos o controla información que proviene de fuera del límite de la aplicación” (The International Function Point Users Group, 2005). La intención principal es mantener uno o más ILF o alterar el comportamiento del sistema.
2. EO (External Output): “Es un proceso elemental que envía datos o controla información fuera de los límites de la aplicación” (The International Function Point Users Group, 2005). La intención principal es presentar información a través de la lógica de procesamiento, además de la recuperación de datos o información de control.
3. EQ (External Inquiry): “Es un proceso elemental que envía datos controla información fuera de los límites de la aplicación” (The International Function Point Users Group, 2005). La intención principal es presentar información al usuario mediante la recuperación de datos o control de información desde un ILF o un EIF.

Al igual que las funciones de datos, las funciones transaccionales tienen complejidad. Esta se mide a partir del número de DET y FTR de cada función transaccional.

Un **FTR (File Type Referenced)** es una función de datos leída o mantenida por una función transaccional. Los FTR se calculan a razón de; un ILF leído o mantenido cuenta como 1 FTR, un EIF leído cuenta como 1 FTR.

Para realizar los cálculos de DET tenemos en cuenta: los botones de acción que inician la validación de datos o almacenamiento cuentan como un DET, si dos botones realizan la misma función se cuenta como un solo DET y los mensajes de error cuentan como un solo DET independientemente de la cantidad de mensajes.

Cuando sabemos el número de funciones transaccionales, y los DET y FTR de cada función transaccional nos fijamos en las tablas de complejidad y conversión para averiguar la cantidad de FP de cada función transaccional.

1. **Calcular el tamaño funcional.**

Para calcular el tamaño funcional, es decir, la complejidad de las funciones de datos y de las funciones transaccionales, dependerá del tipo de fórmula que se use:

– Fórmula para el tamaño funcional de un nuevo desarrollo: Tamaño funcional del proyecto de desarrollo (DFP) = tamaño de las funciones entregadas al usuario con este nuevo desarrollo (ADD) + tamaño funcional de las funciones de conversión (CFP), es decir, .

– Fórmula para el tamaño funcional de una aplicación: Tamaño funcional de la aplicación (AFP) = tamaño de las funciones que proporciona la aplicación (ADD), es decir, .

– Fórmula para el tamaño funcional de un proyecto de mejora: Tamaño funcional del proyecto de mejora (EFP) = tamaño funcional que se añade (ADD) + tamaño de las funciones cambiadas (CHGA) + tamaño de las funciones de conversión (CFP) + tamaño de las funciones eliminadas (DEL), es decir,

.

1. **Documentar e informar**

Realizar toda la documentación respecto al cálculo del tamaño funcional del software.

Como se ha mencionado en apartados anteriores, los puntos función están basados solamente en los requisitos funcionales, por ello IFPUG decidió crear un proceso para la evaluación de los requisitos no funcionales del software que fue denominado SNAP. Al igual que él método FPA tiene un manual, también se ha creado otro para el proceso SNAP, denominado APM (Assesment Practices Manual) (International Function Point Users Group, 2021).

IFPUG es el órgano del gobierno para diferentes certificados que puede adquirir cualquier interesado, estos certificados son; especialista Punto de función certificados (CFPA), el practicante Punto de función certificados (CFPP), y Profesional Certificado SNAP (CSP) certificaciones (Lorenzo, 2021).

Así mismo el propio software que admite recuento de puntos función puede obtener un certificado, dependiendo de las funciones que este realice. Hay tres tipos de certificados:

– Tipo 1: Software que proporciona la funcionalidad del cálculo de los puntos función, este cálculo es realizado por el usuario de forma manual y el software se encarga del almacenaje de los datos.

– Tipo 2: Software que proporciona la funcionalidad de cálculo de los puntos función, este cálculo es realizado mediante la interacción entre el usuario y el sistema. El usuario básicamente responde a las peticiones del sistema y este realiza el cálculo adecuadamente.

– Tipo 3: Software realiza un recuento automático de los puntos función de una aplicación haciendo uso de diferentes fuentes de información. (International Function Point Users Group, 2021).

La aplicación que se ha desarrollado a lo largo de este documento está basada en el tipo 2 de certificación, ya que se basa en la interacción entre el usuario y el sistema.

### MKII

MKII es una abreviación de Mark II análisis de punto función. En acuerdo con ISO/IEC 20968:2002, se define como: “método para el análisis cuantitativo y la medición de aplicaciones de procesamientos de información. Cuantifica los requisitos de procesamiento de información especificados por el usuario para proporcionar una cifra que exprese el tamaño del producto software” (ISO, 2002).

Es un método que permite medir la eficiencia del proceso y administrar costos durante el desarrollo de la aplicación, utilizado principalmente en etapas tempranas del proceso de desarrollo del software. Tiene un componente lógico y transacciones lógicas a razón de los requisitos funcionales. (Gómez, 2013) (Cuesta Meza, López Trujillo, & Joyanes Aguilar, 2019).

### NESMA

Método que se creó a razón del método IFPUG. La cualidad que hace este método diferente es que se ajusta mejor la medición de proyectos en los que se modifica su funcionalidad. Es un método adaptado para los proyectos de mejora, donde solo se tiene en cuenta los cambios realizados en las funciones transaccionales y en las funciones de datos (Gómez, Métodos de Medicion en Puntos Función (II): Nesma, 2013). La definición ofrecida por la página oficial de Nesma es la siguiente: “método para medir el tamaño funcional de un sistema de información. FPA mide el tamaño funcional mirando el (funcional) transacciones y (lógico) archivos de datos que son relevantes para el usuario en el negocio” (Nesma, 2014).

### FISMA

Método definido para medir cualquier software, se creó a partir del método FSM método de experiencia 2.0 FPA que fue usado principalmente en Finlandia. Este método no se centra en el proceso como los anteriores, está orientado al servicio, es decir, el método se basa en encontrar los servicios que proporciona un software (Gómez, Métodos de Medición en Puntos Función (y V): FiSMA, 2013).

### COSMIC-FFP

Es un método que “define una serie de principios, reglas y un proceso que permite medir el tamaño funcional estándar de una pieza de software” (Cosmic, 2020). Respecto al tamaño funcional se refiere a la cantidad de funcionalidad que esta expresada para que el usuario sea capaz de comprenderla. Normalmente el tamaño se obtiene a partir de los requisitos, pero también se puede utilizar otro tipo de artefactos del software como diseños, sistemas instalados, etc. Esta medida está enfocada a software que tienen funciones que ingresan, almacenan y recuperan datos. Distintos estudios destacan que también se puede aplicar con éxito en software con gran manipulación de datos, como es el caso de software científico (Cosmic, 2020) (Gómez, Métodos de Medición en Puntos Función (IV): COSMIC FFP, 2013).

En mayo de 2020, el parlamento europeo recomendó la metodología IFPUG para el cálculo del coste del software. La decisión del parlamento europeo fue la siguiente: “acoge con satisfacción la nueva estructura organizativa establecida por la Agencia para reforzar las capacidades necesarias para la planificación operativa y las adquisiciones asociadas, … la Agencia puede considerar utilizar la metodología del Grupo Internacional de Usuarios de Puntos de Función (IFPUG), Una metodología estándar para determinar el precio de las actividades de desarrollo” (Administracion). La información completa la podemos encontrar [aquí](https://www.europarl.europa.eu/doceo/document/TA-9-2020-0102_EN.html).

# Metodología

## Introducción

En este capítulo se va a desarrollar la metodología usada durante el proyecto software. Se explicará inicialmente la metodología escogida y el motivo por lo que ha sido escogida. Después se mostrará cómo se ha adaptado, como se ha puesto en práctica y las herramientas utilizadas para ello.

## Scrum

Scrum es un marco de trabajo ágil para el desarrollo de proyectos, sus inicios se remontan a mediados de los 80, donde dos japoneses, Hirotake Takeuchi e Ikujijo Nonaja realizaban estudios al respecto. Los japoneses compararon la forma de trabajo en equipo con el avance en formación de una melé, de ahí el nombre de Scrum. La melé es tan complicada que si un solo miembro del equipo se viene abajo, es muy probable que se caiga toda la melé. Para evitar esto los jugadores deben coordinarse y apoyarse para poder avanzar.

En 1993 Jeff Sutherland y Ken Schwaber tomaron como base el estudio de Takeuchi y Nonaka para crear el primer proyecto desarrollado con Scrum. Fue en 1995 cuando Sutherland y Schwaber presentaron Scrum como un proceso formal en la conferencia OOPSLA 96 en Austin, Texas (Lynch, 2019).

En 2001, Sutherland y Schwaber, junto con otros pioneros de la ideología ágil, crearon el manifestó ágil. El manifestó por el desarrollo ágil de software decía; “Estamos descubriendo formas mejores de desarrollar software tanto por nuestra propia experiencia como ayudando a terceros. A través de este trabajo hemos aprendido a valorar:

– **Individuos e interacciones** sobre procesos y herramientas

– **Software funcionando** sobre documentación extensiva

– **Colaboración con el cliente** sobre negociación contractual

– **Respuesta ante el cambio** sobre seguir un plan

Esto es, aunque valoramos los elementos de la derecha, valoramos más los de la izquierda” (Highsmith, 2001).

No solo crearon el manifiesto ágil, sino que, decidieron unos principios para el desarrollo ágil, en concreto fueron 12 principios que podemos ver [aquí](https://agilemanifesto.org/iso/es/principles.html).

En 2010 Sutherland y Schwaber publicaron por primera vez la guía de Scrum, posteriormente han salido diferentes versiones hasta la última publicada en noviembre de 2020. En esta guía podemos encontrar una definición más formal de Scrum; “Scrum es un marco ligero que ayuda a las personas, equipos y organizaciones a generar valor a través de soluciones adaptables para problemas complejos” (Schwaber & Sutherland, 2020).

Scrum está basado en el pensamiento Lean que se basa en centrarse en lo esencial y también está basado en el empirismo que se refiere a la experiencia y toma de decisiones en base a lo que se observa. Scrum tiene tres pilares empíricos esenciales:

– Transparencia: El proceso debe ser visible para los que realizan y reciben el trabajo.

– Inspección: Los artefactos de scrum y el progreso deben de ser inspeccionados frecuentemente.

– Adaptación: Si el producto resultante es inaceptable o si algún proceso se desvía fuera de los límites aceptables, estos deben ajustarse lo antes posible para que cause el menor impacto posible.

Scrum tiene 5 valores para conseguir el éxito; compromiso, enfoque, apertura, respeto y coraje (Schwaber & Sutherland, 2020).

### Roles en Scrum

En Scrum hay diferentes roles según las funciones que se realizan durante el proceso:

– **Product Owner (Propietario):** Es el encargado de representar a todos los interesados en el producto. El Product Owner es responsable de la gestión del Product Backlog (pila de producto).

– **Scrum Master (Líder):** Es el encargado de establecer Scrum como se define en la Guía de Scrum. El Scrum Master debe conseguir que todos los integrantes del equipo entiendan la teoría y la práctica de Scrum.

– **Developers (Desarrolladores):** pequeño equipo de personas que tienen en común el objetivo del producto. Los desarrolladores se auto organizan y, comparten la responsabilidad del trabajo que realizan en cada iteración y en el proyecto. (González López, 2018).

### Proceso de Scrum

En Scrum un proyecto se ejecuta en ciclos de tiempos cortos y con una duración fija que suele ser entre 2 y 4 semanas para completar una cantidad de trabajo establecida, a esto se le denomina Sprint (Albaladejo, 2008).

Scrum tiene una serie de eventos que se realizan durante un sprint:

– **Planificación del Sprint**: reunión que define qué se ha de entregar en el sprint próximo y cómo se va a conseguir ese trabajo. En esta reunión el Product Owner y los desarrolladores deciden que items del Product Backlog (PBI) deberán formar parte del sprint backlog, es decir, los PBI que se añadirán al sprint.

– **Daily Scrum (Scrum diario):** reunión diaria de una duración máxima de 15 minutos. En esta reunión los desarrolladores informan de su trabajo sin entrar en discusiones unos con otros, solo informan y escuchan.

– **Sprint Review (Revisión del sprint):** reunión en la que los desarrolladores muestran al Product Owner y a otros interesados, el desarrollo realizado durante el sprint. También se discute el progreso hacia el objetivo del producto y se determina el siguiente objetivo.

– **Sprint Retrospective (Retrospectiva del sprint):** reunión de todo el Equipo Scrum, donde se busca la forma de aumentar la calidad y la eficacia. El equipo Scrum analiza qué fue bien durante el Sprint, los problemas que surgieron y si se estos fueron solucionados o no, y qué se puede hacer para mejorar.

Nos podemos hacer una idea gráfica del proceso de Scrum fijándonos en la figura: **Figura 3.1**.

Diagrama

Descripción generada automáticamente

**Figura 3.1.** Proceso de Scrum (Eudaimonia, 2020)

### Artefactos de Scrum

La guía de Scrum indica: “Los artefactos de Scrum representan trabajo o valor. Están diseñados para maximizar la transparencia de la información clave”. Los artefactos principales de Scrum son:

– **Product Backlog (Pila del Producto):** Lista ordenada por prioridad en la que se encuentra el trabajo pendiente del producto.

– **Burndown chart (gráfica de progreso):** Gráfico que muestra la relación entre los días del sprint y la cantidad de trabajo pendiente.

– **Sprint Backlog (Pila del Sprint):** Conjunto de elementos que han sido seleccionados para este sprint a partir de la lista del product backlog. Los desarrolladores deben convertir estos elementos en funcionalidad.

– **Increment (Incremento):**  Es un paso sólido hacia el objetivo del producto. Cada incremento es aditivo y debe de asegurarse que todos los incrementos funcionen correctamente juntos (Schwaber & Sutherland, 2020).

## ¿Por qué Scrum?

Antes de escoger la metodología a utilizar en el proyecto se pensó en dos preguntas que pueden ser decisivas a la hora de escoger metodología, estas son:

– ¿Sois un grupo grande o pequeño?

– ¿El cliente quiere una sola entrega final o quiere participar en el desarrollo?

La primera pregunta fue fácil de responder, ya que los desarrolladores cuentan con un solo integrante que es el alumno. Respecto a la segunda pregunta esta fue respondida en la primera reunión entre los desarrolladores (alumno) y el product owner (tutor), este indicó que quería realizar un seguimiento del proyecto y que por lo tanto se involucraría en él.

Una vez sabemos que el equipo es pequeño y que se puede contar con que el Product Owner esté disponible e involucrado en el desarrollo del proyecto se pensó directamente en metodologías ágiles y por consecuente en Scrum, ya que es una de las más utilizadas en la actualidad.

No solo se ha escogido en base a los desarrolladores y a la idea del PO de estar involucrado en el desarrollo. Hay diferentes beneficios que scrum nos puede proporcionar a la hora de desarrollar un software. Scrum es muy fácil de aprender, ya que los roles, hitos y herramientas son bastante sencillos y con un objetivo claro. Debido a su facilidad, lo podemos relacionar con nuestro día a día en el trabajo ya que se asemeja mucho. Una de las grandes ventajas de cualquier metodología ágil y en especial Scrum es la rapidez con la que el cliente puede comenzar a usar el producto. Una vez el cliente puede empezar a usar el producto puede darnos feedback al respecto rápidamente y ayudarnos a mejorar el producto a razón de lo que él quiere.

El proceso se agiliza mucho respecto a metodologías en cascada, ya que se entrega valor en cada sprint que se realiza, es decir, cada 2 o 3 semanas.

En conclusión, Scrum es muy interesante para proyectos con un objetivo claro de entrega de valor continua al cliente, como es el caso de este proyecto. Esto permite al cliente empezar a ver resultados lo más pronto posible.

Esta metodología permite gilizar los procesos, ofrecer transparencia, inspección y adaptación, y también es motivadora para el equipo a través de la independencia para auto organizarse y auto gestionarse (Abellán, 2020).

## Adaptación de Scrum

Como en todo proyecto, lo primero que se hizo fue escuchar al cliente para ver qué es lo que quiere, para cuando lo quiere y como de involucrado quería estar. Una vez dijo que quería estar involucrado durante todo el desarrollo del proyecto se decidió utilizar la metodología Scrum.

Seguidamente se asignaron los roles:

– **Product Owner:** tutor y co-tutor.

– **Scrum Master:** tutor y co-tutor

– **Developers:** alumno.

– **Cliente del producto:** profesor de Gestión de Proyectos Software.

– **Cliente de la memoria:** tribunal.

Se tendrá interacción constante entre los desarrolladores y el PO, este último es el responsable de maximizar el valor entregado al tribunal.

Normalmente el método Scrum está basado en una reunión inicial entre el cliente y el Product Owner para que el cliente explique la idea que tiene en su cabeza. En este caso, en la primera reunión estaban el cliente, el product owner y los desarrolladores. En esta reunión se pretendía sentar las bases del proyecto y entender la idea del cliente lo mejor posible.

Al inicio de la reunión el Product Owner explicó al cliente que se iba a utilizar una metodología ágil llamada Scrum y que esta principalmente se basaba en reuniones cada 2 o 3 semanas donde el equipo le entregaba un incremento funcional del producto, la cual el cliente acogió gratamente.

Seguidamente el cliente explica al PO la idea que tiene. “Se trata de una herramienta visual en la que los elementos estarán prediseñados y que permitirán drag and drop. Una definición gráfica de la aplicación sería una barra de elementos en la parte superior. En la parte derecha tendremos un cuadrante donde nos mostraría los diferentes cálculos junto con un desplegable con las distintas funciones que tenemos. El cálculo total se realizaría con la suma de cada ILF (que no sea repetido), más los EIF que tengamos y las distintas transacciones (EI, EO, EQ)”.

Así mismo, el cliente informa que la aplicación que quiere está relacionada con la asignatura Gestión de proyectos software, concretamente el tema 3 (Bermejo, 2021), por lo que los desarrolladores encontraron familiar los conceptos expuestos.

Una vez el Product Owner y los desarrolladores entienden las peticiones del cliente, se crea un artefacto que es esencial para el desarrollo del producto; la VISIÓN. La VISIÓN se define como una breve declaración del estado futuro deseado, que se lograría mediante el desarrollo y la implementación de un producto (Visual Paradigm, 2021). En otras palabras, lo que el Product Owner tiene en mente de lo que el producto tiene que ser. Se definió la VISIÓN como:

**VISIÓN**: “El Product Owner, que en este caso es el profesor de gestión de Proyectos Software, quiere una aplicación de escritorio de tipo 2 respecto a la certificación de IFPUG para aplicar FPA en clase. Así mismo, el Product Owner quiere que la herramienta sea aprobada por IFPUG y que salga publicada en la revista Metric Views”

Una vez se tiene la VISIÓN y se entiende que es lo que el cliente quiere, se elige la herramienta de comunicación para que los involucrados en el proyecto puedan comunicarse entre sí.

La herramienta de comunicación escogida es **Slack**: “es una aplicación de mensajería para empresas o particulares que conecta a las personas con la información que necesitan. Al reunir a las personas para que trabajen como si fueran parte del mismo equipo, Slack transforma la manera en la que se comunican las organizaciones” (Slack Technologies, 2021). Se ha escogido esta herramienta debido a que combina tanto mensajes en grupos como mensajes entre personas. A pesar de que hay otras herramientas que ofrecen la mayoría de estas características no todas pueden ofrecen la misma funcionalidad. Slack te permite trabajar de forma conectada, flexible e inclusiva (Slack Technologies, 2021). En la figura: **Figura 3.2**, podemos ver la interfaz que nos ofrece esta herramienta.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

**Figura 3.2.** Herramienta de comunicación Slack

Una vez tenemos la herramienta de comunicación instalada, procedemos a crear un canal de comunicación en el que estaremos en contacto los integrantes del proyecto.

Posteriormente el equipo scrum crea otro artefacto que permite crear el Product Backlog, para ello el Product Owner a través de Slack convoca una reunión con el Scrum Master, los desarrolladores y el cliente.

### Inicialización del Product Backlog

Para crear el Product Backlog el equipo, respaldado por el Scrum Master ha decidido usar el taller de User Story Mapping.

**User Story Mapping** “es una idea completamente simple. Habla sobre el recorrido del usuario a través de su producto mediante la creación de un modelo simple que cuenta tu historia de usuario igual que lo haces tú” (Patton, 2014). Esta técnica permite crear requisitos de forma cooperativa entre el Product Owner, los usuarios y los desarrolladores. Es una experiencia de usuario increíble, ya que permite realizar una tarea bastante compleja de una forma relativamente sencilla.

El objetivo de esta técnica es definir el MVP (Minimum Viable Product), o lo que es lo mismo, una sucesión de funcionalidades imprescindibles que aseguran que el producto cumple sus funciones vitales.

Antes de poner en práctica esta técnica nos debemos asegurar que tenemos una pizarra, post-its y rotuladores que van a ser utilizados para componer muchas historias de usuario.

En el inicio del taller el Scrum Master toma las riendas y explica las pautas que se van a seguir. Expone que se van a definir tres capas diferentes, para la primera capa se deben de identificar los distintos roles o personas, la segunda capa esta compuestas por acciones o épicas que cada rol va a realizar y la última capa contendrá las diferentes historias de usuario de cada rol.

Antes de empezar a crear las capas el Scrum Master hace hincapié en recordar los conceptos épica e historia de usuario. Podemos decir que una épica es una gran historia de usuario que no se puede entregar como se define en una sola iteración o es lo suficientemente grande como para dividirse en historias de usuario más pequeñas. En cambio una historia de usuario es una descripción corta y simple de los requisitos de un cliente.

Una vez se han aclarado las pautas a seguir y recordar conceptos importantes, se empieza a crear las diferentes capas. Lo primero que se hace es identificar las personas o roles que van a interactuar con el producto software. Se identificaron tres roles:

– **Medidor Principal:** Usuario que realiza las mediciones con la herramienta.

– **Colaborador:** usuario que puede añadir comentarios

– **Automatización:**  Diferentes cálculos que el sistema realiza.

En la figura: **Figura 3.3** la capa de roles o personajes corresponden con los post-its de color naranja.

Después de obtener los roles, se empieza a identificar las distintas épicas, para ello inicialmente se generarán diferentes épicas y seguidamente se van añadiendo en la pizarra debajo del rol al que más se ajusta. Una vez tengamos las épicas debajo de cada rol estas se reordenan en orden de mayor a menor prioridad. En la figura: **Figura 3.3** podemos ver las épicas de color verde situadas debajo de cada rol y ordenadas por prioridad.

Para la tercera y última capa se crean todas las historias de usuario que cubren con el objetivo del cliente respecto a la aplicación. Para ello se utiliza una plantilla que es muy común a la hora de redactar las historias de usuario. La plantilla tiene tres partes inicialmente para crear la descripción de la historia de usuario y una cuarta parte que aloja los criterios de aceptación de esa descripción. Las partes de la plantilla son:

– **Como** <Rol de Usuario>

– **Quiero** <Objetivo>

– **Para que** <Beneficio>

– **Criterio de aceptación** <Resultado>

En la figura **Figura 3.3** se pueden ver las distintas historias de usuario que se han añadido inicialmente, estas estan representadas en pos-its de color amarillo y siguen las pautas de la plantilla definida previamente. Si ampliamos en la imagen solo podemos ver la descripción de la historia de usuario, ya que los criterios de aceptación estan escritos en la parte trasera del post-it.

Un pizarrón blanco

Descripción generada automáticamente con confianza media

**Figura 3.3.** User Story Mapping

Una vez se ha realizado el taller del Story Mapping el equipo empieza a estimar las historias de usuario mediante puntos de historia. Para ello se apoyan en la escala fibonacci y han empezado a valorarlas haciendo uso de la técnica de las camisetas pero sin seguirla a raja tabla.

Para la estimación se coge inicialmente dos historias de usuario y se comparan, colocando a la izquierda la que creemos que es mas pequeña y a la derecha la mas grande. Despues hemos cogido otra historia de usuario y la hemos colocado en la posición que creiamos a razón de si es mayor o menor que las dos escogidas anteriormente, es decir, si es menor que la de la derecha y mayor que la de la izquierda pues esta se colocara en medio. Sucesivamente se han ido cogiendo una a una las historias de usuario y se ha ido realizando el mismo mecanismo creando así una lista de historias de usario de menor a mayor.

Para poder estimar correctamente debemos de tener en cuenta que si alguna historia de usuario no se puede realizar en una iteración, estas deben de ser desglosadas en historias de usuario mas pequeñas. En nuestro caso no se ha encontrado ninguna que exceda los límites de una iteración. Seguidamente se cogen tres historias de usuario como referentes, estas suelen ser la menor y mayor en la lista y luego otra a mitad de la lista que el equipo tenga bastante clara. Al tener las referencias elegidas se puede proceder a darle valor a las demás historias de usuario en función de esas tres.

### Transparencia del proceso con Jira

Una vez hemos realizado el taller de Story Mapping podemos generar el Product Backlog. Para ello, nos hemos apoyado en la herramienta Jira. El Scrum Master recomienda la herramienta Jira debido a que es la herramienta más utilizada por equipos ágiles.

Jira se define como “una familia de productos creada para ayudar a todos los tipos de equipos con la gestión de su trabajo. Jira ofrece varias opciones de implementación y productos diseñadas específicamente para el software, las TI, la empresa, los equipos de operaciones y mucho más” (Atlassian, 2018).

Jira es una plataforma que contiene tres productos; Jira Software, Jira Service Management y Jira Work Management. En nuestro caso hemos cogido el producto Jira software para que todos los miembros del equipo scrum puedan planificar, supervisar y publicar software de calidad.

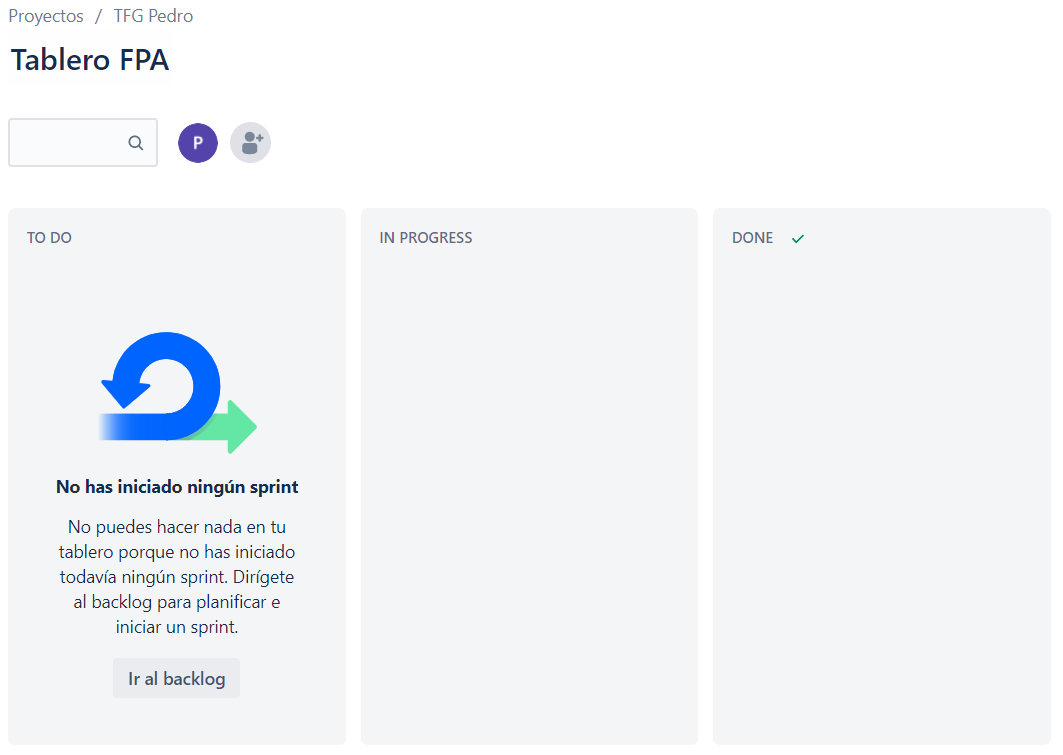
Para poder utilizar Jira Software, el equipo scrum ha tenido que registrarse y crear una cuenta. Una vez la cuenta esta creada, se crea un proyecto, que en este caso se ha denominado TFG Pedro. Después de generar el proyecto procedemos a generar el product backlog, en el que comenzamos añadiendo todas las historias de usuario obtenidas en el user story mapping. En el capítulo siguiente explicaremos más detalladamente el Product Backlog y los items de este.

Posteriormente el Product Owner y los desarrolladores reordenan estas historias de usuario a razón de su prioridad. Para ello primero se ha elegido el nivel de prioridad en el campo especifico para ello, este tiene 5 niveles desde menor a mayor prioridad. Seguidamente se han reordenado las historias ayudandose del campo prioridad.

Ahora ya se puede inicializar las iteraciones en el proceso de Scrum, ya que el artefacto inicial necesario para ello es el Product Backlog. El siguiente paso que se realiza es generar un tablero o sprint backlog en Jira Software, para ello esta herramienta nos ofrece diferentes tableros por defecto y también nos permite crear uno desde cero.

Se ha decidido crear un tablero nuevo, este estaria compuesto por tres fases por las que pasarian las historias de usuario seleccionadas para esa iteración. Las tres fases del tablero son; To Do, In Progress y Done.

En la figura: **Figura 3.4** podemos ver el tablero generado en Jira Software. La primera fase,To Do, aloja las historias de usuario que aun no han sido iniciadas y que están pendientes durante el sprint. La segunda fase In Progress, aloja las historias de usuario que estan siendo realizadas, es decir, en las que los desarrolladores estan trabajando. La tercera fase Done, aloja las historias de usuario que han sido completadas.



**Figura 3.4.** Tablero / Sprint Backlog

Antes de inicializar el primer sprint, se añade los puntos de historia y el criterio de aceptación a las historias de usuario , para los puntos de historia Jira Software ya esta preparado y en los propios campos de la historia de usuario puedes añadirlo. En cambio para los criterios de aceptación hay que añadir otro campo desde la configuración. El campo se añade al tipo de incidencia historia de usuario y a partir de ahí aparece en todas las historias de usuario. Seguidamente se añade a cada historia de usuario sus criterios de aceptación.

Una vez tenemos las historias de usuario con todos sus campos completos podemos inicializar el primer sprint y añadir las historias de usuario que queremos que se realicen durante esa iteración. Para la selección de las historias de usuario el Product Owner es el que indica que historias son prioritarias y cuales se deben de hacer durante la iteración, teniendo en cuenta la carga de trabajo de los desarrolladores.

### Documentación del proceso

Para poder documentar las iteraciones que se van realizando, se crea una plantilla. Esta plantilla permitira reflejar todo lo que afecta a una iteración desde sus inicios hasta su finalización. La plantilla creada esta compuesta por 5 apartados, que son realizados en el orden siguiente:

– **Objetivo del Sprint:** El product owner indica la meta que se quiere conseguir al terminar la iteración.

– **Planificación del Sprint:** El Product Owner, junto con los desarrolladores indican las historias de usuario que pueden realizar durante la iteración.

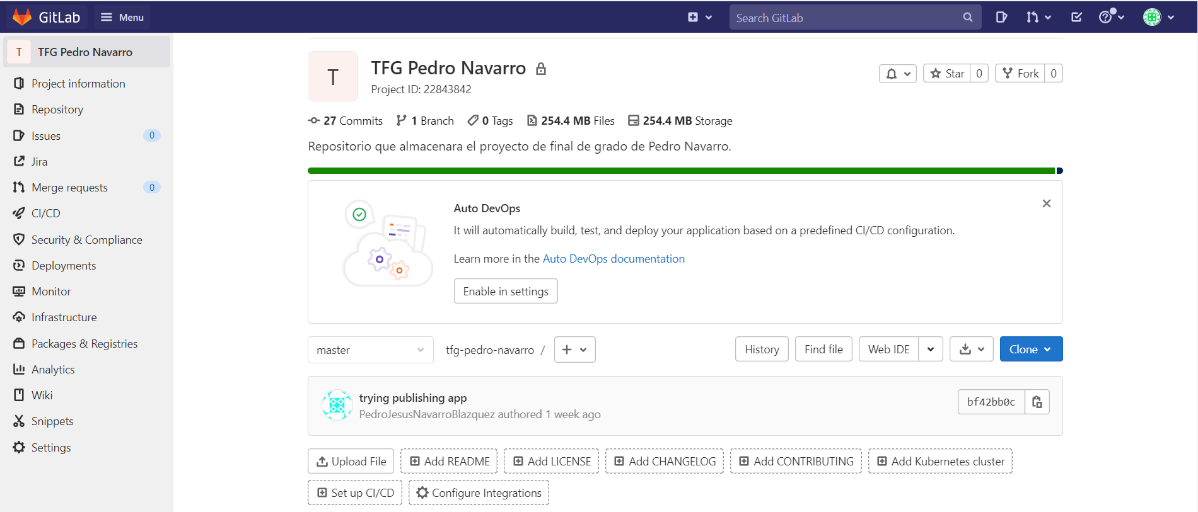
– **Desarrollo del Sprint:** Los desarrolladores indican el incremento realizado.

– **Revisión del Sprint:** Se presenta al Product Owner y al cliente las nuevas funcionalidades implementadas durante la iteración.

– **Retrospectiva del Sprint:** Los integrantes del equipo Scrum discuten que ha ido bien y que hay que cambiar para mejorar.

### Stack tecnológico

Durante las iteraciones del proyecto se incrementa la funcionalidad del software. Para poder controlar las diferentes funcionalidades creadas se utiliza GitLab. GitLab es un servicio web de control de versiones y desarrollo de software colaborativo, o lo que es lo mismo es un servicio web que nos permite alojar el repositorio de nuestro proyecto de software (Romero, 2021). Hay muchos servicios que ofrecen las mismas funcionalidades o muy parecidas, pero decidimos escoger este debido a que es el más utilizado en la actualidad y el equipo aun no lo había utilizado en ningún otro proyecto. En la figura: **Figura 3.5**, podemos ver la interfaz de GitLab con el repositorio TFG Pedro Navarro que se ha desarrollado durante este proyecto. Para tener una mayor integración entre la documentación y el repositorio del software, se vincula el servicio GitLab dentro de la herramienta Jira Software.



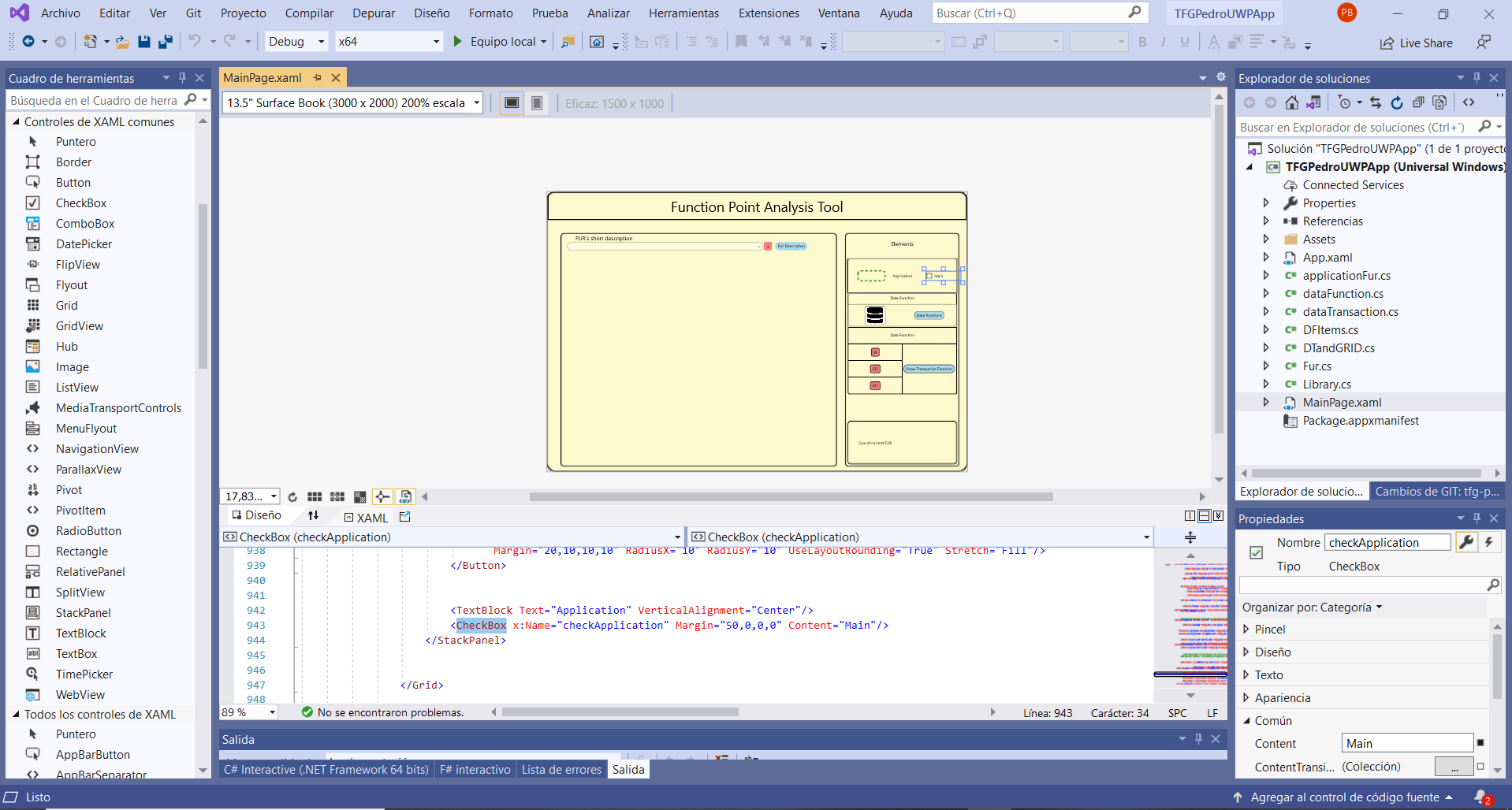
**Figura 3.5.** GitLab

Para poder desarrollar la aplicación software que el cliente ha pedido, el equipo scrum hace uso de de la herramienta Visual Studio Enterprise. Esta herramienta permite desarrollar cualquier tipo de aplicación trabajando en un entorno de desarrollo integrado (IDE). Esta herramienta no solo permite la edición de código, también reúne diseñadores gráficos, compiladores, herramientas de finalización de código, control de código fuente y muchas otras características (Microsoft, 2021).

Visual Studio Enterprise tiene diferentes areas de desarrollo, nos hemos enfocado en .NET, que nos permite desarrollar aplicaciones para escritorio que es el objetivo de nuestro cliente. Hay diferentes tecnicas para crear una aplicación de escritorio en .NET, como es WPF, Winforms, UWP, etc. El equipo utiliza UWP (Universal Windows Platform), que permite crear aplicaciones que se ejecutan en dispositivos de windows 10.

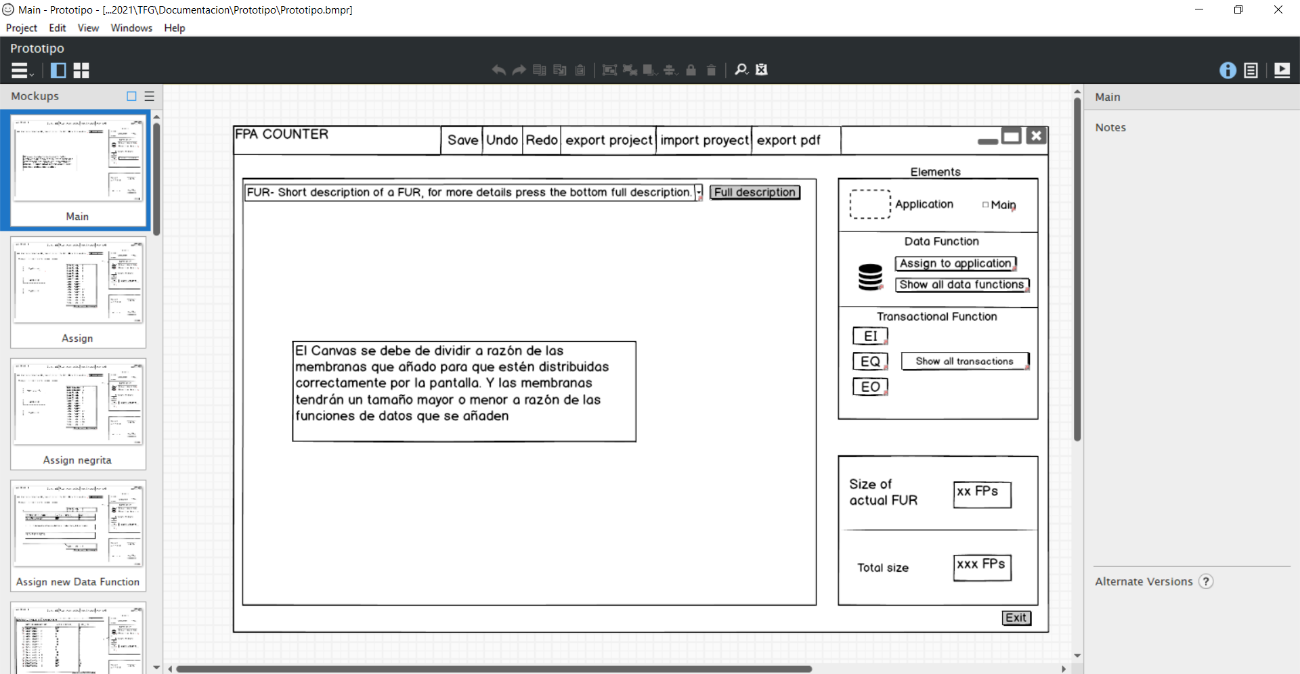
UWP permite crear aplicaciones seguras con funcionalidades específicas del dispositivo; la interfaz se adapta al tamaño de la pantalla del dispositivo, esta disponible desde Microsoft Store y permite elegir diferentes lenguajes para su programación. Se ha programado con lenguaje C# para el código del proyecto y XAML para la interfaz de usuario.

En la figura: **Figura 3.6** podemos ver Visual Studio inicializado con la técnica UWP, con C# como lenguaje y XAML como interfaz.



**Figura 3.6.** Aplicación Escritorio en UWP con C# y XAML

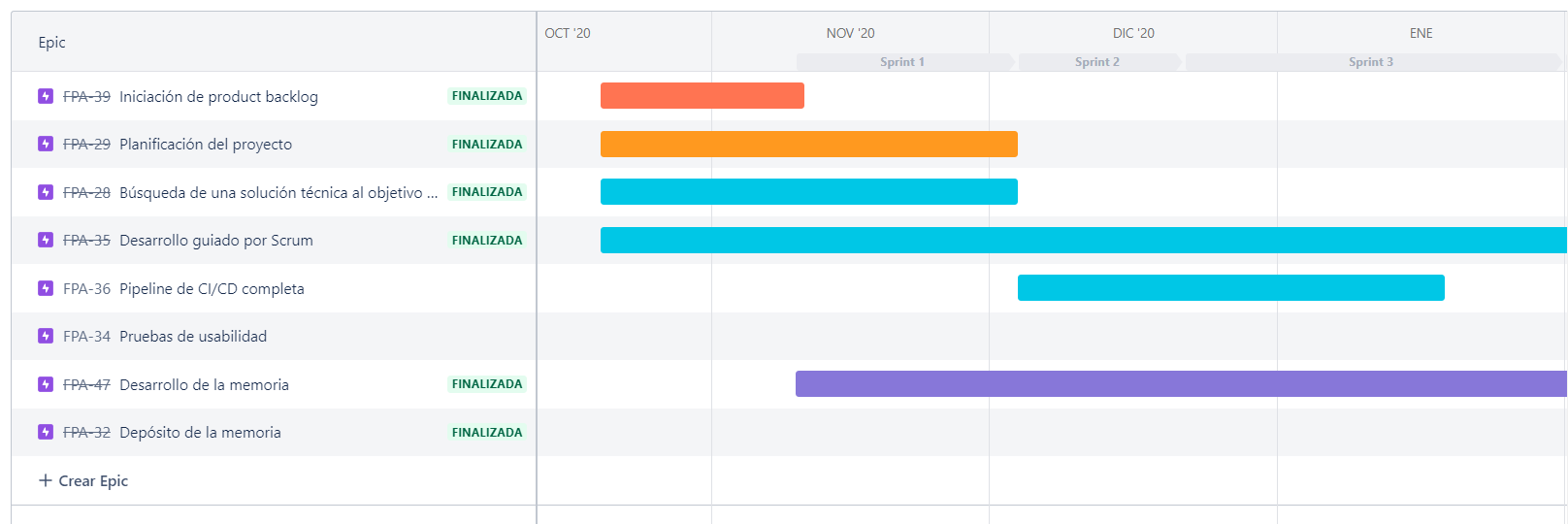
Durante las primeras iteraciones del proyecto, se realiza un prototipo y para ello se usa la herramienta Balsamiq Mockups para poder realizar un prototipado de la aplicación. Balsamiq Mockups “es una herramienta rápida de estructura de líneas de interfaz de usuario de baja fidelidad que reprodue la experiencia de dibujar en un bloc de notas o pizarra, pero usando una computadora” (Balsamiq, 2008). Esta herramienta permite crear un prototipo funcional por el que puedes navegar, esto fue un factor primordial a la hora de escoger esta herramienta, ya que se le mostraria al cliente una visión gráfica de como quedaria la aplicación. Otro factor importante fue que el equipo ya habia utilizado la herramienta con anteriorirdad y se realizaria el prototipo relativamente rápido. La figura: **Figura 3.7** representa la interfaz de Balsamiq Mockups con el prototipo del proyecto ya creado.



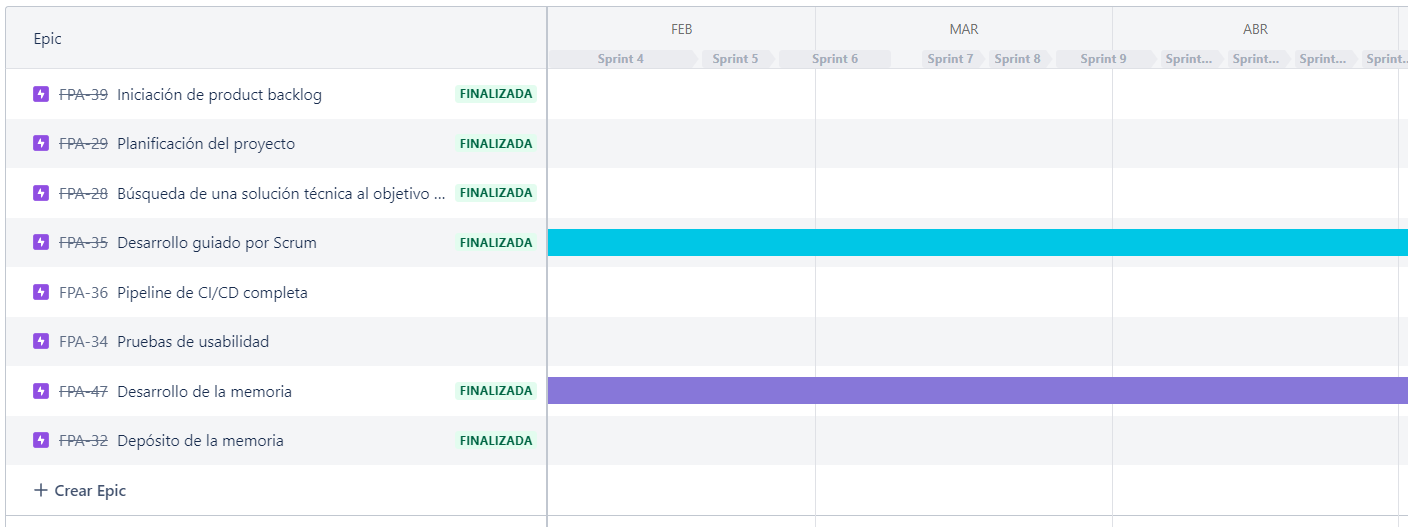
**Figura 3.7.** Balsamiq Mockups

Al principio del proceso Scrum se crea un artefacto denominado cronograma de hitos, su objetivo es mostrar un plan de hitos desde el principio del desarrollo del proyecto. Este artefacto se ha creado en Jira Software en el apartado denominado hoja de ruta. La figuras: **Figura 3.8**, **Figura 3.9** y **Figura 3.10**, muestran el cronograma de hitos dividido por trimestres.

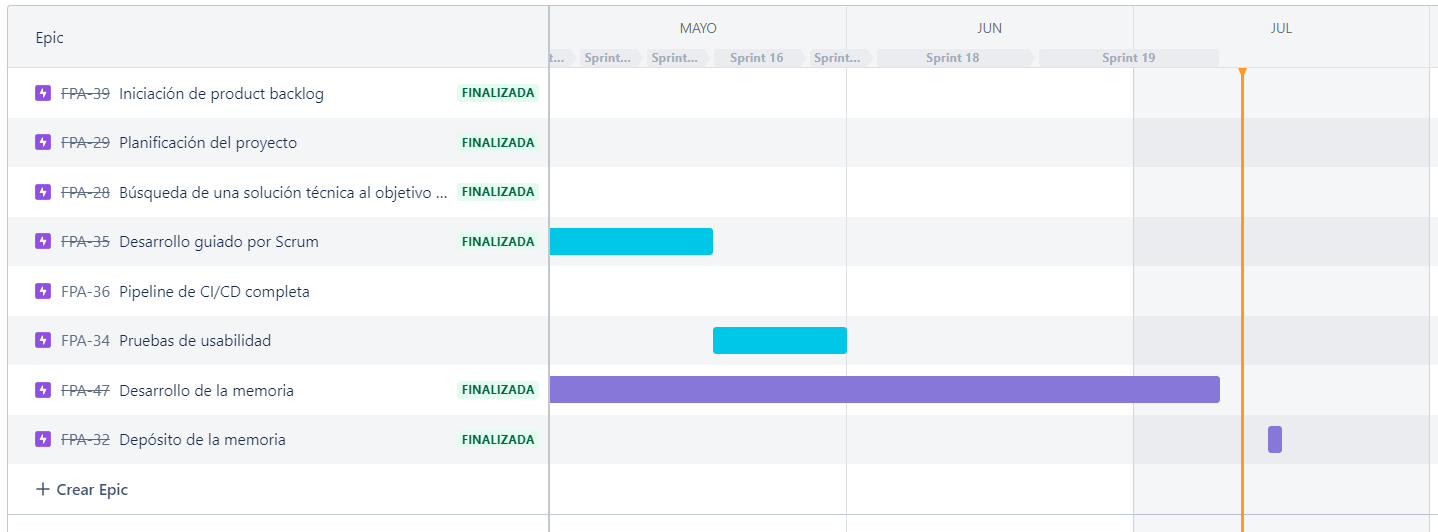
### Temporización del desarrollo



**Figura 3.8:** Cronograma de Hitos (octubre-enero)



**Figura 3.9:** Cronograma de Hitos (febrero-abril)



**Figura 3.10:** Cronograma de Hitos (mayo-julio)

Este artefacto permite supervisar y controlar el desarrollo de todas las actividades del proyecto, es decir, realizar un seguimiento del progreso del proyecto.

El alcance del proyecto no ha sido alcanzado completamente, debido a qe hay algunas tareas que no se han completado. Tanto la iniciación del product backlog como la planificación del proyecto cumplieron perfectamente con el alcance del proyecto. La búsqueda de una solución técnica al objetivo del proyecto no se ha completado por muy poco, debido a que solo ha faltado el calculo total de un conjunto de FURs y el poder guardar y cargar un proyecto. Respecto a la integración continua y el despliegue continuo no han sido completados con éxito, ya que el despliegue se ha realizado al final del desarrollo del proyecto y la integración continua solo ha sido completada con la compilación del proyecto, pero no con las pruebas. El desarrollo de la memoria y el depósito si han sido completadas con éxito y a razón del cronograma de hitos. En definitiva, si que ha habido desviación en el alcance del proyecto. Esto se debe principalmente a que es el primer proyecto de esta magnitud que realiza el equipo de desarrollo y se ha encontrado con problemas inesperados, los cuales tendra en cuentan para futuros proyectos.

Si aplicamos correctamente el proceso Scrum tendremos muchas posibilidades de entregar un producto con calidad y acorde a lo que el cliente pide. Es muy importante elegir los framework y herramientas adecuados, estos nos ayudaran a realizar el proceso de una manera más sencilla y organizada. En la tabla **Tabla 3.1** podemos ver los que se han utilizado durante este proyecto.

**Tabla 3.1.** Herramientas y frameworks usados.

|  |  |  |
| --- | --- | --- |
| Nombre | Logo | Descripción |
| Slack |  | Aplicación de mensajería para empresas o particulares que conecta a las personas con la información que necesitan. |
| Balsamiq Mockups |  | Herramienta rápida de estructura de líneas de interfaz de usuario de baja fidelidad que reproduce dibujar en pizarra o en bloc de notas, pero usando una computadora. |
| GitLab |  | Servicio web de control de versiones y desarrollo de software colaborativo |
| Visual Studio |  | Herramienta que permite desarrollar cualquier tipo de aplicación trabajando en un entorno de desarrollo integrado (IDE). |
| Jira Software |  | Producto diseñado para que todos los miembros de tu equipo de software puedan planificar, supervisar y publicas software de calidad. |
| Windows Store | Imagen que contiene Gráfico  Descripción generada automáticamente | Plataforma de distribución de software desarrollada por Microsoft. |

# Historial Sprints

## Introducción

En este capítulo nos centraremos en los distintos artefactos generados para el seguimiento de los Sprints los cuales se explicarán de forma detallada. Inicialmente se explorarán 2 artefactos que están conectados directamente con el historial de Sprints. Seguidamente se verán todos los sprint realizados, que siguen una estructura definida previamente.

## Artefactos

Antes de detallar las iteraciones realizadas durante el desarrollo del proyecto se va a explorar dos artefactos que han sido creados en Jira Software y que están vinculados a los Sprints.

El primer artefacto que vamos a analizar es el Product Backlog, para ello nos vamos a ayudar de la tabla: **Tabla 4.1**, que representa el listado de historias de usuario iniciales que fueron obtenidas mediante la técnica User Story Mapping. Este artefacto representa un listado de todas las historias que se pretenden hacer durante el desarrollo del proyecto, no es una lista definitiva ya que es dinámica. El Product Backlog ha permitido a los integrantes del equipo tener una visión transparente de todo lo que se espera realizar.

**Tabla 4.1.** Product Backlog inicial

|  |  |
| --- | --- |
| ID | Descripción |
| FPA-1 | Como Tutor quiero que se inicialice el Product Backlog antes de comenzar el desarrollo Scrum, para así tener un conjunto priorizado de primeros PBIs a elegir para el Sprint 1. |
| FPA-23 | Como Tribunal quiero un cronograma para que el alumno demuestre que tenía un plan a nivel de hitos desde el principio. |
| FPA-2 | Como medidor principal quiero crear un proyecto nuevo para poder empezar a aplicar FPA (Functional Point Analysis). |
| FPA-3 | Como medidor principal quiero abrir un proyecto ya creado para continuar con la cuenta. |
| FPA-4 | Como medidor principal quiero deshacer y rehacer cambios para corregir errores. |
| FPA-5 | Como medidor principal quiero insertar una descripción textual de un FUR para poder insertar las funciones de datos. |
| FPA-6 | Como medidor principal quiero crear funciones de datos (EIF, ILF) para indicar la funcionalidad de almacenamiento. |
| FPA-7 | Como medidor principal quiero editar funciones de datos (EIF, ILF) para modificar la funcionalidad de almacenamiento. |
| FPA-8 | Como medidor principal quiero borrar funciones de datos (EIF, ILF) para no tener funciones que no se utilizan o que son erróneas. |
| FPA-9 | Como medidor principal quiero crear aplicaciones para alojar las funciones de datos. |
| FPA-10 | Como medidor principal quiero editar aplicaciones para modificar las características de la aplicación. |
| FPA-11 | Como medidor principal quiero borrar aplicaciones para poder así quitar las que no son correctas o que no sean necesarias. |
| FPA-12 | Como medidor principal quiero crear funciones transaccionales en un FUR para poder asignarlas a elementos. |
| FPA-13 | Como medidor principal quiero modificar funciones transaccionales en un FUR para editarlas a nuestra conveniencia. |
| FPA-14 | Como medidor principal quiero borrar funciones transaccionales en un FUR para así quitarla de donde no corresponde |
| FPA-15 | Como medidor principal quiero asignar funciones de datos a FUR y aplicaciones para que se sumen a la cuenta. |
| FPA-16 | Como medidor principal quiero conectar funciones y membranas o aplicaciones para así poder realizar los cálculos |
| FPA-17 | Como medidor principal quiero exportar un informe para mostrar el tamaño funcional y los distintos elementos añadidos |
| FPA-18 | Como medidor principal quiero guardar el proyecto para continuar su edición más adelante. |
| FPA-20 | Como medidor principal quiero ver las tablas de complejidad. |
| FPA-21 | Como colaborador quiero comentar FURs, DETs, RETs, FTRs para opinar en relación con las funciones creadas. |
| FPA-22 | Como automatización quiero calcular la complejidad de los FUR y la aplicación para obtener el tamaño funcional del programa. |
| FPA-24 | Como tutor queremos que recopiles la información necesaria para el Estado del Arte para así poder seguir con el desarrollo de la memoria. |
| FPA-25 | Como tutor queremos que instancies la plantilla de TFG. |

El segundo artefacto que vamos a explorar es el Project Burn-up Chart, este es un gráfico basado en la cantidad de incidencias y el tiempo. Si nos fijamos en la figura: **Figura 4.1**, esta representa el burn up chart del proyecto que se ha realizado. En la gráfica el color lila representa las historias a realizar y el color verde las ya completadas. Si nos fijamos en los inicios del desarrollo la velocidad del equipo en completar tareas era bastante lento, ya que se tuvieron que documentar bastante antes de poder realizar muchas de las tareas. La imagen muestra como entre febrero y marzo hubo una subida bastante notable. En cambio, desde marzo hasta mediados de abril hubo un crecimiento de la curva bastante lento debido a que hubo problemas para desarrollar algunas tareas como es el caso de creación de elementos arrastrables. Una vez se solucionaron esas tareas el equipo empezó a completar tareas a un ritmo bastante alto durante el resto del desarrollo del proyecto. Al final del desarrollo del proyecto, podemos apreciar, como no se han completado todas las tareas que están pendientes de hacer y que podrían realizarse en futuras versiones de la aplicación. En Scrum, gracias a la acción del Product Owner, podemos asegurar que las tareas no realizadas son las que aportan menos valor al cliente.

Gráfico, Histograma

Descripción generada automáticamente

**Figura 4.1.** Project Burn-Up chart

## Memoria de los sprints

En este apartado se van a desarrollar los Sprints que se han realizado a lo largo del desarrollo del proyecto. Cada sprint tiene 5 elementos.

### Sprint-1 (3-Semanas)

**Objetivo del sprint**: Primeros pasos en los contenidos de la memoria y en la selección de la herramienta de la programación.

**Planificación del sprint:** Para el primer sprint se ha decidido añadir al tablero las siguientes historias de usuario:

FPA-1: Como tutor quiero que se inicialice el product backlog antes de comenzar el desarrollo Scrum, para así tener un conjunto priorizado de primeros PBIs a elegir para el Sprint 1.

FPA-23: Como tribunal quiero un cronograma para que el alumno demuestre que tenía un plan a nivel de hitos desde el principio.

También se ha acordado añadir al product backlog 3 historias de usuario nuevas, que serían:

FPA-24: Como tutor queremos que recopiles la información para el Estado del Arte.

FPA-25: Como tutor queremos que instancies la plantilla de TFG.

FPA-26: Como tutor queremos que se inicialice el repositorio que almacena el código fuente

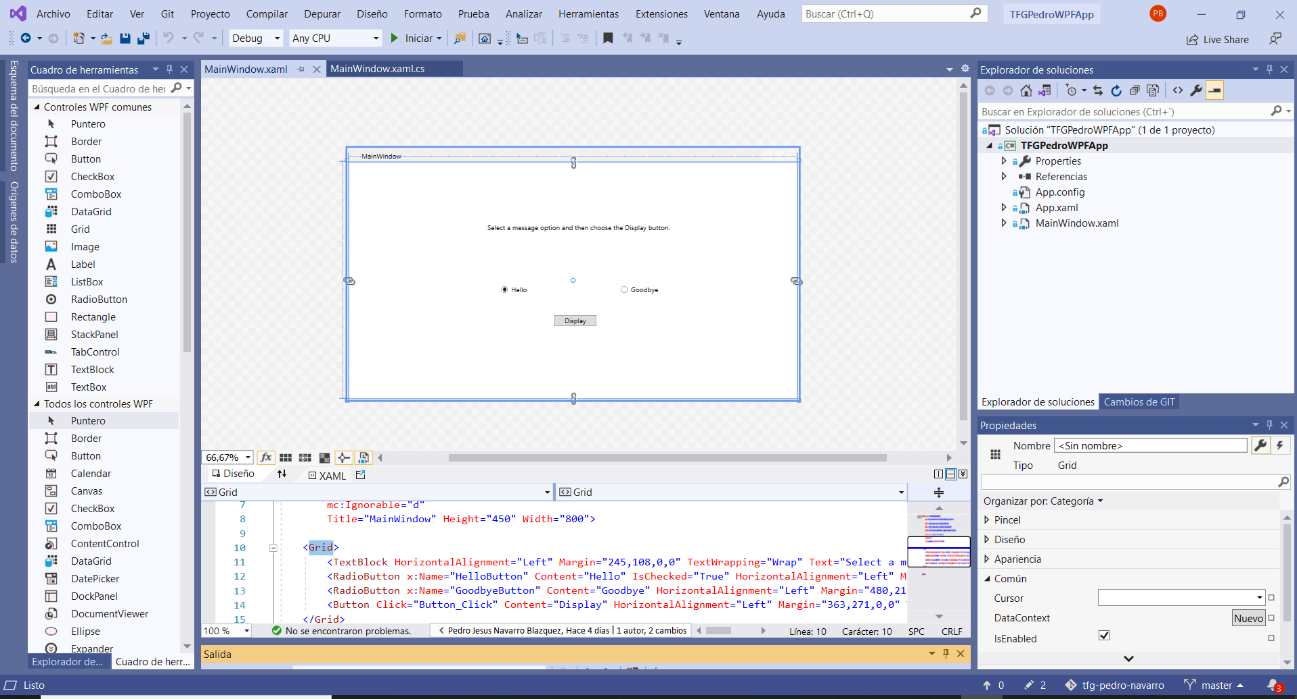
Tanto FPA-25 como FPA-26 han sido añadidas al sprint backlog o tablero de este sprint.

**Desarrollo del sprint:** Durante este sprint se ha inicializado el product backlog y se han ordenado las historias de usuario a razón de su prioridad. Se ha utilizado la herramienta Jira tanto para añadir el product backlog como para crear el cronograma de hitos. Por otra parte se ha inicializado el repositorio en GitLab como se puede apreciar en la figura: **Figura 4.3** y se ha creado un proyecto con la tecnología WPF de Microsoft.

**Revisión del sprint:** La revisión se ha iniciado con el análisis del proyecto creado de Hola Mundo en WPF, podemos apreciar la interfaz de este en la figura: **Figura 4.2**. Los desarrolladores han comentado algunas dudas respecto al futuro de la aplicación. Por otro lado se ha revisado el cronograma de hitos y se han realizado pequeñas modificaciones que el scrum master creía necesarias. Finalmente se ha verificado la inicialización de la plantilla del TFG. Todas las historias de usuario del sprint han sido completadas.

**Retrospectiva del sprint:**

Por la facilidad de desplegar en tienda y por el aspecto estético junto con que Windows va a dejar de darle soporte se ha decido pivotar de WPF a UWP.



**Figura 4.2.** Hola Mundo en WPF

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

**Figura 4.3.** Repositorio del proyecto en GitLab

### Sprint-2 (2.5 Semanas)

**Objetivo del sprint**: Desarrollo del estado del arte en la memoria y creación de un Hola Mundo arrastrable en UWP.

**Planificación del sprint:** Inicialización del estado del arte de la memoria. Los desarrolladores deben documentarse respecto a UWP. Para ello se añaden los PBIs siguientes al tablero:

FPA-24: Como tutor queremos que recopiles la información para el Estado del Arte.

Se ha añadido un nuevo PBI al product backlog y posteriormente se ha añadido al tablero de este sprint, este PBI es:

FPA-43: Como tutor quiero que crees un Hola Mundo arrastrable en UWP para poder así empezar con el desarrollo del código.

**Desarrollo del sprint:** Durante este sprint se ha creado un nuevo proyecto con la tecnología UWP como se acordó en el sprint anterior y se ha añadido al repositorio de GitLab. Por otra parte, se ha empezado a recopilar información respecto al estado del arte. También se ha vinculado la herramienta Jira con el servicio de control de versiones GitLab para que este todo integrado, para ello se ha utilizado el siguiente [video](https://www.youtube.com/watch?v=fWvwkx5_00E).

Al final del desarrollo se ha realizado una pequeña búsqueda de información respecto al estado del arte.

**Revisión del sprint:** En la revisión se ha comenzado analizando el proyecto de Hola Mundo creado en UWP. Seguidamente se ha revisado la información adquirida para el estado del arte. El PBI respecto a la memoria no se ha completado, ya que se seguirá recopilando información, mientras que el PBI respecto a UWP sí que se ha completado.

**Retrospectiva del sprint:**

En este sprint el equipo scrum ha hecho hincapié en que hay que acelerar la búsqueda y escritura de información.

### Sprint-3 (6 Semanas)

**Objetivo del sprint**: Crear un prototipo de la interfaz principal y desarrollar el apartado de estado del arte de la memoria.

**Planificación del sprint:** Realizar un prototipo de la aplicación y continuar con el estado del arte de la memoria. Por lo tanto se añaden al tablero los PBIs:

FPA-24: Como tutor queremos que recopiles la información para el Estado del Arte.

Se han creado 2 PBIs nuevos que han sido añadidos al product backlog y seguidamente añadidos al tablero de este sprint. Estos nuevos PBIs son:

FPA-45: Como tutor quiero que crees un prototipo para hacernos una idea de cómo se vera la interfaz principal.

FPA-46: Como tutor queremos que escribas la información necesaria para el Estado del arte para así poder seguir con el desarrollo de la memoria.

**Desarrollo del sprint:** Durante este sprint se ha recopilado información y se ha escrito el estado del arte de la memoria. A su vez se ha creado un prototipo de la interfaz inicial de la aplicación con la herramienta Word, se puede ver gráficamente en la figura: **Figura 4.4**.

**Revisión del sprint:** Al principio de la revisión se ha inspeccionado la interfaz principal del prototipo. Después se ha revisado las partes principales del estado del arte de la memoria, por lo que se han completado todas las historias correctamente.

**Retrospectiva del sprint:** Por la dificultad de creación de un prototipo con la herramienta Word y debido a que no permite navegabilidad se ha decidido pivotar a otra herramienta que sí que nos permita navegar, esta es Balsamiq Mockups. También se ha decidido que el idioma de la aplicación sea inglés.

**Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente**

**Figura 4.4.** Prototipado inicial con Word

### Sprint-4 (2 semanas)

**Objetivo del sprint**: Crear prototipado que sea navegable y añadir funcionalidad al proyecto.

**Planificación del sprint:** Crear el prototipo con la herramienta Balsamiq Mockups y dar funcionalidad al proyecto de UWP.

Las historias de usuario añadidas al tablero de este sprint son:

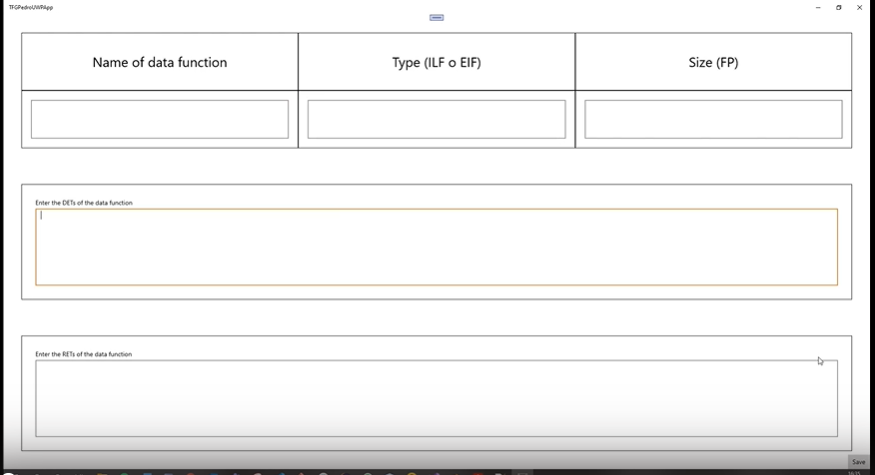
FPA-48: Como tutor quiero que crees un prototipo detallado de cómo se vera la interfaz para tener una idea más clara de la aplicación.

FPA-49: Como tutor quiero que añadas funcionalidad al proyecto para ver como interactúa la interfaz.

**Desarrollo del sprint:** Al principio del desarrollo los desarrolladores se han instalado la herramienta mockups para el prototipado. Seguidamente se han empezado a crear las pantallas necesarias para que el prototipado sea navegable. Una vez se tenía el prototipo creado se ha procedido a añadir funcionalidad al proyecto.

**Revisión del sprint:** Se ha revisado el prototipo navegable para darle una idea al cliente de como quedaría la aplicación y en paralelo se ha revisado el proyecto creado. En el proyecto se ha detectado un error en las pantallas, ya no se adaptaban bien a la pantalla, como se puede apreciar en la figura: **Figura 4.5**. Se ha concretado que se debe de solucionar las pantallas del proyecto, para ello los desarrolladores han sugerido el uso de pantallas emergentes o popups.

**Retrospectiva del sprint:** El equipo scrum ha resaltado el gran potencial de la herramienta Balsamiq Mockups para realizar prototipos navegables.



**Figura 4.5.** Pantalla sin ajustarse al prototipo

### Sprint-5 (1 semana)

**Objetivo del sprint**: Mejorar el sistema de ventanas y los campos editables de la interfaz.

**Planificación del sprint:** Se debe arreglar el sistema de ventanas y los campos editables del proyecto. Para ello se han añadido al tablero los siguientes PBIs:

FPA-5: Como medidor principal quiero insertar una descripción textual de un FUR para poder insertar las funciones de datos.

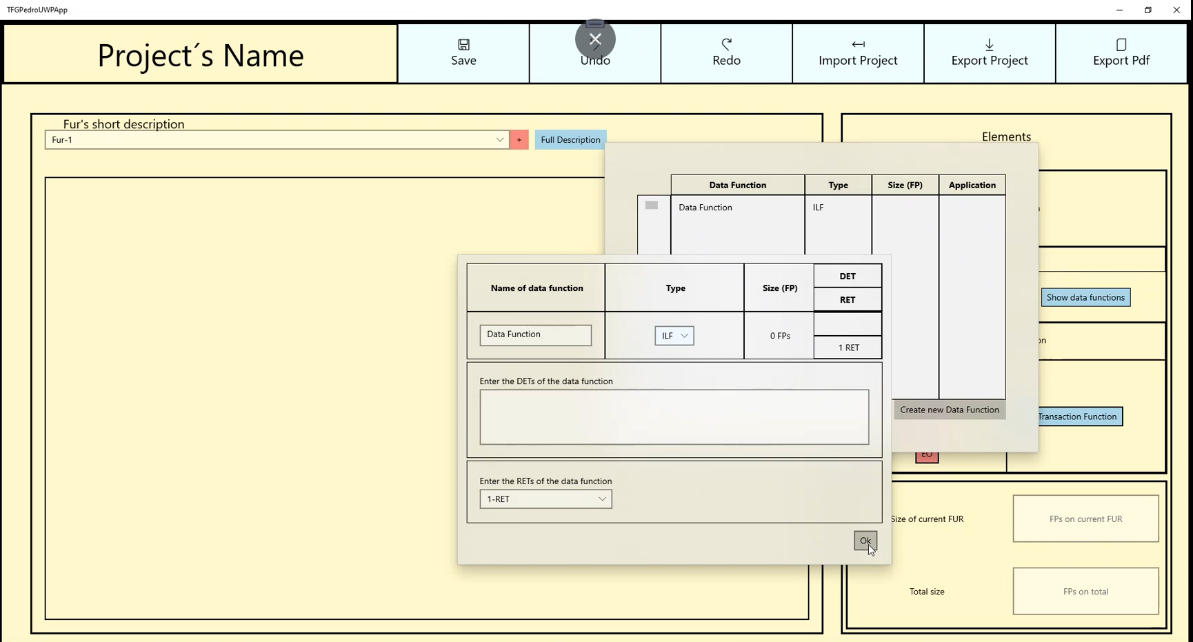
FPA-50: Como tutor quiero que mejores el sistema de ventanas y los campos editables de la interfaz para obtener una mejor navegación a través de la aplicación.

**Desarrollo del sprint:** Al inicio del sprint los desarrolladores se han documentado respecto a las pantallas emergentes o popups. Para ello se ha decidido realizar un curso denominado; “The complete Universal Windows Platform course” en la plataforma de aprendizaje Udemy, este curso esta accesible [aquí](https://www.udemy.com/course/complete-uwp-course/) y no es gratuito.

**Revisión del sprint:** Inicialmente se han revisado los popups creados en el proyecto para la navegabilidad entre pantallas, un ejemplo de estos se muestra en la figura: **Figura 4.6**. Seguidamente se han revisado los campos editables modificados. Todos los PBIs han sido completados con éxito.

**Retrospectiva del sprint:**

Se ha creado la DoD (definition of done) con un primer requisito; todos los elementos del proyecto se deben de poder abrir y guardar.



**Figura 4.6**. Popups

### Sprint-6 (2 semanas)

**Objetivo del sprint**: Mejorar la interfaz de usuario y darles mayor funcionalidad a los campos editables de la interfaz.

**Planificación del sprint:** Solucionar los errores encontrados en la revisión del sprint anterior y documentarse sobre drag and drop. Se añaden al tablero el PBI:

FPA-51: Como tutor quiero que añadas funcionalidad arrastrable a los FUR y mejora de la interfaz para obtener una mejor navegación a través de la aplicación.

**Desarrollo del sprint:** Al principio del desarrollo los desarrolladores han solucionado los distintos errores respecto a la funcionalidad de la interfaz. Después se han documentado respecto a drag and drop en UWP pero no se ha conseguido implementar en el proyecto.

**Revisión del sprint:** Inicialmente se han revisado todas las modificaciones que se mencionaron en el sprint anterior. Al no guardarse y abrir el proyecto no se ha satisfecho la DoD. Por último, se ha discutido la dificultad de crear elementos arrastrables, ya que no se han podido implementar y se ha analizado la documentación al respecto.

**Retrospectiva del sprint:**

Se ha decidido que los desarrolladores tienen que documentarse más detalladamente respecto a drag and drop o buscar alguna alternativa al respecto.

### Sprint-7 (2 semanas)

**Objetivo del sprint**: Creación de archivo xml para guardar y cargar datos.

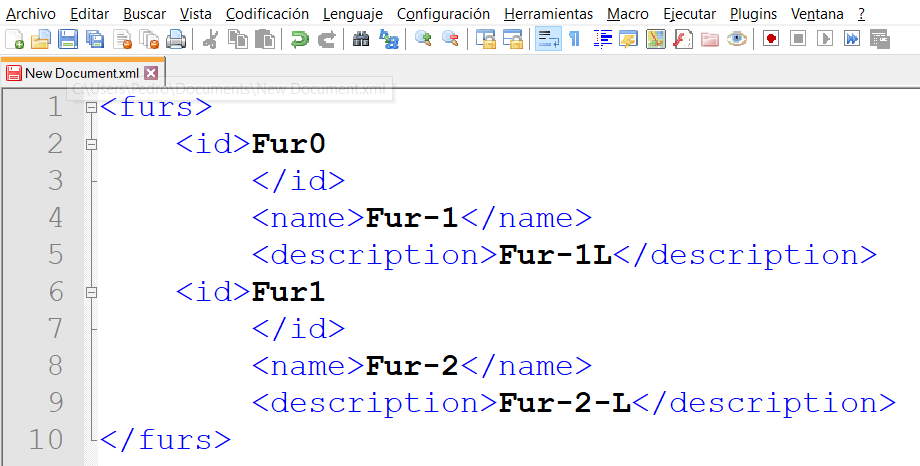
**Planificación del sprint** Los desarrolladores deben de documentarse respecto a archivos xml. Una vez se han documentado, deben crear un archivo xml local que contenga el nombre del proyecto, los FUR que hay definidos y su descripción. Debe de poderse guardar y volver a cargar el archivo xml. Por lo que se ha añadido al tablero el PBI:

FPA-52: Como tutor quiero que guardes un proyecto con su nombre para posteriormente volverlo a cargar.

**Desarrollo del sprint:** Inicialmente los desarrolladores se han documentado respecto a archivos xml para la tecnología UWP. Se han encontrado con bastantes problemas ya que la mayoría de la información estaba enfocada a WPF. Finalmente, los desarrolladores han aplicado los conceptos adquiridos para realizar el guardado de datos en el archivo xml.

**Revisión del sprint:** Se ha comentado la dificultad para encontrar información de archivos xml para la tecnología UWP. Seguidamente se ha mostrado el archivo xml creado cuando se guarda, pero no se ha conseguido realizar la carga de los archivos. El archivo xml creado se puede ver en la figura: **Figura 4.7**.

**Retrospectiva del sprint:** El equipo scrum ha acordado que los desarrolladores deben documentarse más respecto a la carga de archivo xml en UWP y buscar una solución al respecto. Asimismo, se ha devuelto el PBI al backlog debido a que no ha sido completado.



**Figura 4.7.** Archivo XML inicial

### Sprint-8 (1 Semana)

**Objetivo del sprint**: Guardar los FUR y aplicaciones creadas en archivo xml para posteriormente volverlas a cargar.

**Planificación del sprint** Se debe de poder guardar y cargar un proyecto. Posteriormente se debe de guardar y cargar los FUR (Funcional User Requirements) y aplicaciones creadas. Así pues, se han añadido al tablero del sprint los PBIs:

FPA-52: Como tutor quiero que guardes un proyecto con su nombre para posteriormente volverlo a cargar.

FPA-53: Como tutor quiero que guardes un FUR y aplicaciones para posteriormente poder volverlas a cargar.

**Desarrollo del sprint:** Al principio del sprint se ha conseguido cargar un archivo xml a nuestra aplicación con el nombre del proyecto. Luego se ha guardado cada FUR con sus respectivas aplicaciones que contienen una etiqueta correspondiente al FUR al que pertenecen. Más adelante se ha conseguido la carga de los FUR a la aplicación.

**Revisión del sprint:** En la revisión se ha comprobado la generación de un archivo xml con el nombre del proyecto, los FUR creados y las aplicaciones dentro de estos. Seguidamente se ha revisado la carga de datos desde el archivo xml, aunque solo se cargan los FUR.

**Retrospectiva del sprint:** El equipo scrum está de acuerdo en sacar el PBI que no se ha completado del tablero y devolverlo al product backlog.

### Sprint-9 (2 Semanas)

**Objetivo del sprint**: Crear aplicaciones y funciones de datos.

**Planificación del sprint:** Al inicio del sprint los desarrolladores deben documentarse respecto a elementos arrastrables. Después se debe de dar funcionalidad arrastrable a las aplicaciones y crear funciones de datos. Se ha añadido al tablero del sprint los PBIs:

FPA-53: Como tutor quiero que guardes un FUR y aplicaciones para posteriormente poder volverlas a cargar.

FPA-51: Como tutor quiero que añadas funcionalidad arrastrable a los FUR y mejora de la interfaz para obtener una mejor navegación a través de la aplicación.

FPA-6: Como medidor principal quiero crear funciones de datos (EIF, ILF) para indicar la funcionalidad de almacenamiento.

**Desarrollo del sprint:** Al principio del sprint los desarrolladores se han centrado en la documentación respecto a los elementos arrastrables en UWP. Después han intentado implementar las aplicaciones arrastrables.

**Revisión del sprint:** En la revisión se ha expuesto los problemas respecto a los elementos arrastrables, ya que no han podido ser implementados. Los PBIs vuelven al product backlog, ya que no se han completado.

**Retrospectiva del sprint:** Los desarrolladores deben documentarse más a fondo sobre elementos arrastrables. La DoD se ha actualizado y ahora debe de garantizar que todos los elementos dentro de ese se arrastran con él. Así mismo se ha añadido un criterio de aceptación al PBI de crear aplicación indicando que esta debe ser arrastrable.

### Sprint-10 (1 Semana)

**Objetivo del sprint**: Guardar y cargar aplicaciones y funciones de datos.

**Planificación del sprint:** Los desarrolladores deben documentarse respecto a los elementos arrastrables para implementarlos. Después deben de poder guardar y cargar datos en el archivo xml. Los PBIs añadidos al tablero son:

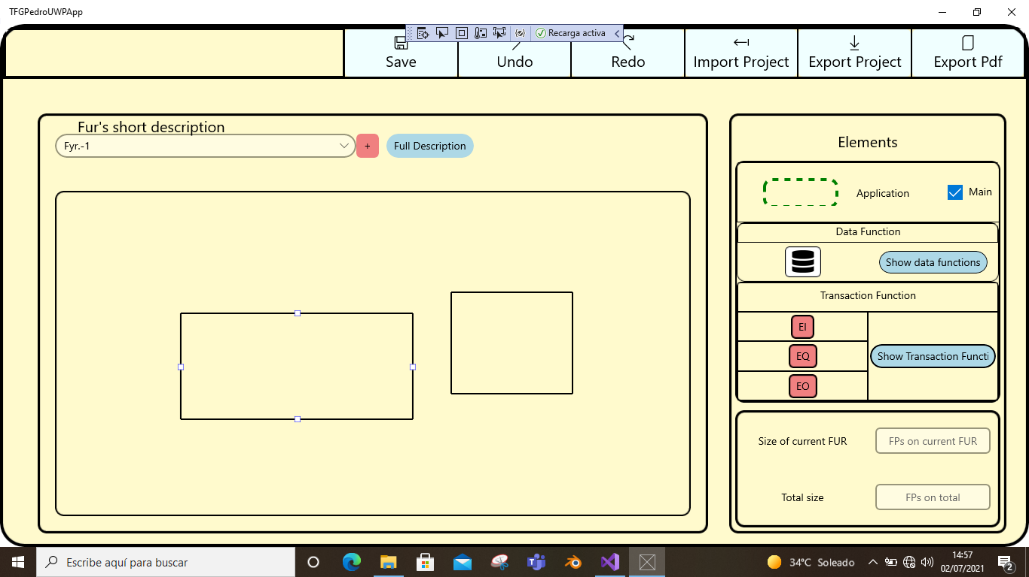
FPA-53: Como tutor quiero que guardes un FUR y aplicaciones para posteriormente poder volverlas a cargar.

FPA-51: Como tutor quiero que añadas funcionalidad arrastrable a los FUR y mejora de la interfaz para obtener una mejor navegación a través de la aplicación.

**Desarrollo del sprint:** En este sprint se empezó con la documentación respecto a los elementos arrastrables, seguidamente los desarrolladores centraron todo su esfuerzo en aplicar funcionalidad arrastrable a las aplicaciones.

**Revisión del sprint:** Durante la revisión se comprueba la funcionalidad arrastrable de las aplicaciones, como se muestra en el ejemplo de la figura: **Figura 4.8**. A pesar de tener el elemento arrastrable se ha discutido que este no tenía nombre, por lo que no estaría finalizado. Por lo tanto los PBIs vuelven al backlog debido a que no están completados. Se ha decidido que al arrastrar el rectángulo el nombre de este debe de moverse junto con él, es decir, moverse como una agrupación.

**Retrospectiva del sprint:** Ningún problema detectado que resolver ni cambio en la comunicación o herramientas del equipo.



**Figura 4.8.** Canvas con aplicaciones arrastrables

### Sprint-11 (1 Semana)

**Objetivo del sprint**: Crear aplicaciones arrastrables que se guarden y que se puedan volver a cargar.

**Planificación del sprint** Los desarrolladores deben crear aplicaciones arrastrables con un nombre y que estas se puedan guardar y volver a cargar. Se ha añadido el PBI:

FPA-53: Como tutor quiero que guardes un FUR y aplicaciones para posteriormente poder volverlas a cargar.

**Desarrollo del sprint:** Al inicio del desarrollo del sprint los desarrolladores han conseguido que las aplicaciones arrastrables tuvieran un nombre asignado. Una vez las aplicaciones se creaban correctamente se centraron en guardar y cargar datos en el archivo xml.

**Revisión del sprint:** Al principio se ha revisado el guardado y carga de aplicaciones respecto al archivo xml que se ha realizado correctamente. Seguidamente se ha analizado los arrastrables creados, pero ha surgido una discusión respecto a la dificultad de añadir elementos dentro de los arrastrables. Los desarrolladores han argumentado que incluso para añadir un simple nombre se ha tardado demasiado tiempo.

**Retrospectiva del sprint:** Se ha decidido pivotar de elementos arrastrables a agrupaciones de elementos con un tamaño fijo.

### Sprint-12 (1 Semana)

**Objetivo del sprint**: Agrupación de elementos para la creación de aplicaciones dentro de un FUR.

**Planificación del sprint**: Creación de aplicaciones dentro del canvas como una agrupación de elementos fijos. Se ha añadido al tablero el PBI:

FPA-54: Como tutor quiero que crees un conjunto de elementos como grupo para así poder mover los distintos elementos dentro del grupo.

**Desarrollo del sprint:** Al inicio del sprint los desarrolladores se han documentado respecto a las diferentes agrupaciones que UWP ofrece. Seguidamente han decido hacer uso del Grid que permite crear agrupaciones en su interior de manera fija y han creado aplicaciones como agrupación dentro del canvas del FUR.

**Revisión del sprint:** Durante la revisión se ha comprobado las agrupaciones de aplicación creadas dentro del canvas principal. Los desarrolladores han puesto principal hincapié en que las aplicaciones deben poder moverse en el canvas. Los desarrolladores han comentado que hay un tipo de Grid que permite movilidad entre sus elementos, se denomina GridView.

**Retrospectiva del sprint:** Se ha decido pivotar de Grid a GridView ya que este permite que sus elementos puedan moverse entre las distintas posiciones que ofrece.

### Sprint-13 (1 Semana)

**Objetivo del sprint**: Crear aplicaciones y funciones de datos arrastrables en el canvas principal.

**Planificación del sprint:** Los desarrolladores deben de crear un GridView con 8 aplicaciones y que estas se puedan mover de posición en el canvas. Se ha añadido al tablero el PBI:

FPA-55: Como tutor quiero que las aplicaciones queden bien ordenadas en el canvas principal o bien haciendo que las cuadriculas de cuadro sean las aplicaciones o haciendo uso del GridView adaptable.

**Desarrollo del sprint:** Al principio del sprint los desarrolladores se han documentado respecto al funcionamiento del GridView. Seguidamente han creado agrupaciones de GridView dentro de un GridView principal, por lo que tanto las aplicaciones como sus elementos pueden moverse.

**Revisión del sprint:** Se ha comprobado las diferentes agrupaciones creadas y la capacidad de estas de moverse dentro de un GridView. Se ha llegado al acuerdo que esta forma de representar las agrupaciones es intuitiva y se asemeja bastante a la idea del cliente.

**Retrospectiva del sprint:** Se ha modificado la DoD debido a la dificultad de cumplir el guardado y carga de agrupaciones. Por lo tanto, se ha dado prioridad a otras tareas y se dejara la carga y guardado para futuras versiones de la aplicación.

### Sprint-14 (1 Semana)

**Objetivo del sprint**: Dar funcionalidad completa a las funciones de datos y a las aplicaciones.

**Planificación del sprint: G**arantizar la funcionalidad de las aplicaciones y funciones de datos. Para ello, se ha añadido al tablero de este sprint los PBIs:

FPA-51: Como tutor quiero que añadas funcionalidad arrastrable a los FUR y mejora de la interfaz para obtener una mejor navegación a través de la aplicación.

FPA-6: Como medidor principal quiero crear funciones de datos (EIF, ILF) para indicar la funcionalidad de almacenamiento.

FPA-7: Como medidor principal quiero editar funciones de datos (EIF, ILF) para modificar la funcionalidad de almacenamiento.

FPA-8: Como medidor principal quiero borrar funciones de datos (EIF, ILF) para no tener funciones que no se utilizan o que son erróneas.

FPA-9: Como medidor principal quiero crear aplicaciones para alojar las funciones de datos.

FPA-11: Como medidor principal quiero borrar aplicaciones para poder así quitar las que no son correctas o que no sean necesarias.

**Desarrollo del sprint:** Los desarrolladores han solucionado pequeños detalles respecto a la funcionalidad de las aplicaciones y han dado funcionalidad a las funciones de datos.

**Revisión del sprint:** En la revisión se ha mostrado las nuevas funcionalidades respecto a las funciones de datos, comprobando que todas las requeridas funcionan correctamente excepto la edición. Por ello se ha discutido como poder solucionar la funcionalidad de edición.

**Retrospectiva del sprint:** El equipo scrum ha comentado la buena respuesta respecto a la cantidad de trabajo que se pedía durante este sprint, a pesar de que un PBI no ha sido completado.

### Sprint-15 (1 Semana)

**Objetivo del sprint**: El objetivo de este sprint seria sumar el tamaño de todos los elementos creados en un FUR.

**Planificación del sprint:** Solucionar la edición de las funciones de datos y calcular el tamaño de los elementos de un FUR. Se ha añadido al tablero los PBIs:

FPA-56: Como medidor principal quiero que se sume los tamaños de los elementos creados en un FUR.

FPA-7: Como medidor principal quiero editar funciones de datos (EIF, ILF) para modificar la funcionalidad de almacenamiento.

**Desarrollo del sprint:** Los desarrolladores han empezado solucionando la edición de las funciones de datos creando solamente una ventana tanto para la edición como para el borrado. Después han controlado todos los elementos sumatorios de una función de datos y se han realizado los cálculos en función de su complejidad.

**Revisión del sprint:** Durante la revisión se ha comprobado la edición de las funciones de datos y seguidamente los cálculos de estos. Se han detectado pequeños errores de cálculo, respecto a la cantidad de DET.

**Retrospectiva del sprint:** El equipo scrum ha estado de acuerdo en que no se pueden realizar despistes respecto a cálculos que son relativamente sencillos.

### Sprint-16 (1 Semana)

**Objetivo del sprint**: Crear funciones transaccionales, poder editarlas y borrarlas. **Planificación del sprint:** Añadir funciones transaccionales dentro de las aplicaciones y que puedan ser editadas y borradas. Se ha añadido al tablero los PBIs:

FPA-12: Como medidor principal quiero crear funciones transaccionales en un FUR para poder asignarlas a elementos.

FPA-13: Como medidor principal quiero modificar funciones transaccionales en un FUR para editarlas a nuestra conveniencia.

FPA-14: Como medidor principal quiero borrar funciones transaccionales en un FUR para así quitarla de donde no corresponde.

FPA-15: Como medidor principal quiero asignar funciones de datos a FUR y aplicaciones para que se sumen a la cuenta.

FPA-16: Como medidor principal quiero conectar funciones y membranas o aplicaciones para así poder realizar los cálculos.

**Desarrollo del sprint:** Al inicio, los desarrolladores han conseguido crear funciones transaccionales en forma de flecha. Durante el tiempo restante del sprint se ha intentado vincular de forma correcta la flecha con la función a la que corresponde, pero no se ha conseguido.

**Revisión del sprint:** La revisión se ha centrado en la creación de la función transaccional, donde los desarrolladores han mostrado la dificultad de vincularla con la aplicación correspondiente. Por lo que se ha decidido buscar una solución rápida y que sea funcional. Por lo tanto, se han cumplido los 2 PBI que no están relacionados con las funciones transaccionales.

**Retrospectiva del sprint:** El equipo scrum ha decidido cambiar el dibujar las flechas por pasar asignar un botón cuando se cree una función transaccional y que este se asigne a la función de datos correspondiente.

### Sprint-17 (1 Semana)

**Objetivo del sprint**: Crear Funciones transaccionales, editarlas y eliminarlas.

**Planificación del sprint:** Cambiar el modo de creación de las funciones transaccionales. Así mismo, debe implementar tanto la edición como el borrado de estas. Para ello se ha añadido al tablero de este sprint los PBIs:

FPA-12: Como medidor principal quiero crear funciones transaccionales en un FUR para poder asignarlas a elementos.

FPA-13: Como medidor principal quiero modificar funciones transaccionales en un FUR para editarlas a nuestra conveniencia.

FPA-14: Como medidor principal quiero borrar funciones transaccionales en un FUR para así quitarla de donde no corresponde.

**Desarrollo del sprint:** Al inicio del desarrollo los desarrolladores han solucionado la creación de funciones transaccionales. Seguidamente se han centrado en implementar la función de eliminar una función transaccional.

**Revisión del sprint:** En la revisión se ha comprobado la creación y eliminación de funciones transaccionales. Después se ha discutido los motivos por los que la edición no se ha podido realizar y se han comentado soluciones al respecto. Por lo que se han completado todos los PBIs, excepto el de edición que vuelve al Product Backlog.

**Retrospectiva del sprint:** El equipo scrum ha sido capaz de buscar soluciones para la creación y eliminación, aun así, deben de buscar una solución a la eliminación.

### Sprint-18 (2 Semanas)

**Objetivo del sprint**: Editar la edición de las funciones transaccionales.

**Planificación del sprint:** Solucionar la edición de funciones transaccionales. También debe de eliminar cualquier elemento sin funcionalidad. Se ha añadido al tablero del sprint el PBI:

FPA-13: Como medidor principal quiero modificar funciones transaccionales en un FUR para editarlas a nuestra conveniencia.

**Desarrollo del sprint:** Durante el sprint los desarrolladores han estado centrados en la edición de funciones transaccionales. Al final de este se ha explorado la aplicación y se ha eliminado elementos sin funcionalidad o con funcionalidad incompleta como es el caso de guardar y cargar, ya que estos no van a poder finalizarse en esta versión de la aplicación.

**Revisión del sprint:** Se ha revisado la edición de las funciones transaccionales y también se ha revisado la aplicación en general para concretar que todas las funcionalidades hasta el momento son correctas. Se ha decidido eliminar los botones de guarda y carga de datos ya que no hay tiempo suficiente para realizarlo en esta versión. Se ha decido que la aplicación esta lista para su despliegue.

**Retrospectiva del sprint:** Ningún problema detectado que resolver ni cambio en la comunicación o herramientas del equipo.

### Sprint-19 (1 Semana)

**Objetivo del sprint**: Desplegar aplicación en tienda.

**Planificación del sprint:** Este sprint está basado en el despliegue de la aplicación en la tienda de Windows, para ello se ha añadido al tablero del sprint el PBI:

FPA-57 - Como medidor principal quiero que la aplicación este desplegada en tienda para poder usarla.

**Desarrollo del sprint:** Inicialmente los desarrolladores se han documentado respecto al despliegue de la aplicación en la tienda de Windows. Los pasos para poder desplegar en tienda son:

– Registrarse como desarrollador en Microsoft, que conlleva un pago de 17 euros.

– En el proyecto en la herramienta Visual Studio hay que modificar los ajustes y marcar la casilla de .NET native tool chain.

– Reservar un nombre de la aplicación desde Microsoft Partner Center o desde Visual Studio.

– Generar un paquete desde la solución del proyecto en visual studio.

– Al tener el nombre de la aplicación reservada, se puede realizar una sumisión, donde hay que rellenar varios apartados; precio y disponibilidad, propiedades de la aplicación, clasificación de edad, añadir el paquete, indicar el lenguaje y opción de publicación.

Para que la herramienta pueda ser publicada debe de pasar todas las pruebas en la sumisión. Podemos ver el despliegue más detallado en el anexo; Anexo II.

**Revisión del sprint:** Al inicio se ha mostrado la aplicación ya desplegada en la tienda. Todos los integrantes estaban de acuerdo en el potencial de la tecnología escogida (UWP). También se ha comentado el product backlog con las historias restantes, como se ve en la figura: **Figura 4.9**.

**Retrospectiva del sprint:** El equipo ha comentado el potencial de la tecnología escogida (UWP). La explicación del despliegue de forma más detallada la podemos encontrar en el anexo: Anexo II

Texto

Descripción generada automáticamente

**Figura 4.9**. Product Backlog Final

Una vez se han explicado los diferentes sprints realizados durante el desarrollo del proyecto, hay que matizar que la entrega de valor continua al cliente nos ha ayudado a descubrir problemas rápidamente. Al descubrir problemas rápido podemos atajarlos rápido para que tengan el menor impacto posible. Aun no habiendo realizado todas las historias de usuario durante los sprints realizados, se han ido implementando las que tenían más valor para el cliente. Los desarrolladores han tenido una gran experiencia con la metodología Scrum, la cual utilizaran en muchos proyectos futuros.

# Herramienta

## Introducción

En este capítulo se va a explicar la aplicación creada durante el desarrollo del proyecto, es decir, el producto final que se ha conseguido. Inicialmente se va a realizar una breve descripción general de la herramienta. A continuación se va a utilizar diferentes pantallas que nos ofrece la aplicación para poder explicar su funcionamiento más detalladamente. Para ello, se explicarán los elementos de cada pantalla y cómo funcionan, con el objetivo de que los leyentes de este documento puedan entenderla y utilizarla.

## Descripción

La herramienta desarrollada durante este proyecto se ha denominado “FPA TOOL PN” y ha sido desarrollada por Pedro Navarro. Está disponible en la tienda de Windows y es gratuita.

La herramienta que se ha desarrollado es un software que proporciona la funcionalidad de cálculo por puntos función. Es una aplicación de escritorio que ha sido desarrollada con la tecnología FPA de IFPUG. Para poder realizar el cálculo de puntos función en la aplicación, es necesaria la interacción entre el usuario y el sistema.

Una breve explicación de la herramienta seria; es una herramienta que permite realizar cálculos en puntos función, para ello se parte de un requisito funcional al que se añaden una o más aplicaciones. Una función peculiar de estas aplicaciones es que pueden cambiarse de posición entre ellas, así como también pueden alojar funciones de datos. Las funciones de datos tienen una serie de elementos, de estos elementos depende su complejidad. Al igual que las aplicaciones, las funciones de datos pueden intercambiarse entre sí pero siempre dentro de la aplicación en la que se encuentran. Las funciones de datos a su vez alojan funciones transaccionales que al igual que las funciones de datos tienen una serie de elementos y estos afectan su complejidad. El cálculo final del FUR (Functional User Requirements) en puntos función dependerá de la cantidad de funciones de datos y funciones transaccionales añadidas y la complejidad de cada una de esas funciones.

Para poder entender mejor la herramienta, a continuación se va a explicar más detalladamente todas las funcionalidades y todos los elementos que nos podemos encontrar una vez la tengamos instalada.

## Pantallas de la aplicación

Al inicializar la herramienta FPA TOOL PN encontraremos una interfaz bastante sencilla. Si nos fijamos en la figura: **Figura 5.1**, podemos ver que la interfaz inicial se divide principalmente en tres partes. Estas tres partes son:

– **Parte de arriba**: donde se encuentra el título que se le ha dado a la aplicación.

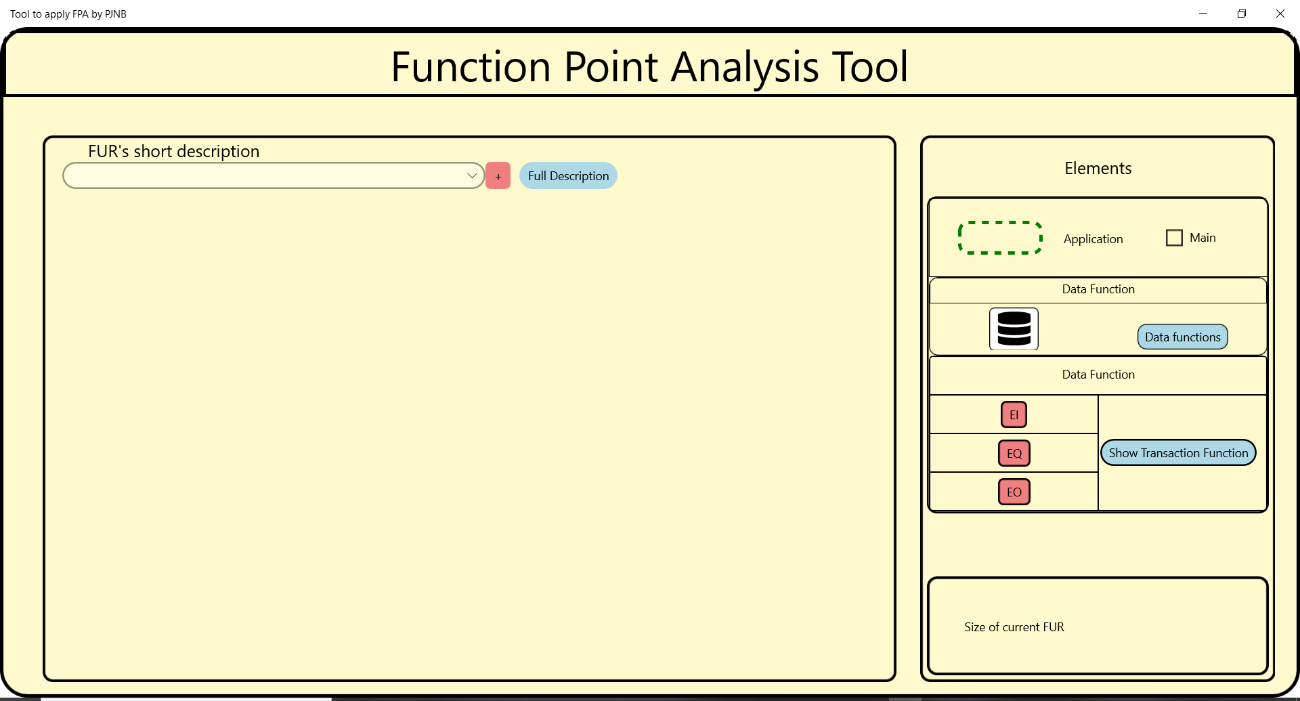
– **Parte de la izquierda**: cubre la mayor parte de la pantalla y define el área o canvas donde se van a ir añadiendo los elementos. En esta parte hay dos botones, uno representado como “+” que permite acceder a la pantalla flotante de crear un nuevo FUR, la cual se muestra en la figura: **Figura 5.2**. A la izquierda del botón tenemos un desplegable donde se irán almacenando los diferentes FUR que se añaden. Este desplegable es muy importante, ya que todos los elementos del canvas están relacionados con el FUR seleccionado. El área o canvas cambiara de elementos dependiendo del FUR que se seleccione. Por último tenemos también otro botón “Full Description”, este accede a la pantalla flotante que muestra la descripción completa del FUR que este seleccionado en el desplegable, como se muestra en la figura: **Figura 5.3**.

– **Parte de la derecha:** en esta parte se encuentran los distintos elementos que se pueden añadir al área o canvas mencionado previamente. En la parte de arriba tenemos un rectángulo con líneas discontinuas que representa el elemento aplicación, a la derecha de este tenemos un checkbox que está vinculado al rectángulo. Cuando el checkbox está marcado la aplicación tendrá como particularidad que es la principal y se mostrara el contorno de esta en rojo. Debemos de tener en cuenta que en cada FUR solo puede haber una aplicación principal, por lo que el resto de las aplicaciones que se añadan deberán de ser secundarias y se representan con un contorno de color verde, podemos ver en la figura: **Figura 5.6**, un FUR con una aplicación principal (rojo) y otra secundaria (verde). La diferencia entre si es aplicación principal o secundaria es principalmente que la aplicación principal contiene funciones de datos de tipo ILF y la aplicación secundaria de tipo EIF.

Seguidamente, debajo del rectángulo tenemos los elementos de función de datos. Hay un botón con forma de base de datos, que nos llevara a la pantalla emergente de creación de funciones de datos, que podemos ver en la figura: **Figura 5.7**. A la derecha hay otro botón “Data Function”, este accede a la pantalla emergente que muestra todas las funciones de datos que hay en el FUR actual, podemos ver esta lista en la figura: **Figura 5.11** .

Debajo de las funciones de datos, nos encontramos con las funciones transaccionales. Podemos apreciar 3 botones de color rojo que representan los tres tipos de funciones transaccionales que existen. Los tres botones nos llevan a la pantalla flotante de creación de función transaccional, que podemos ver en la figura: **Figura 5.12**. Al igual que para las funciones de datos, para las transaccionales también hay un botón “Show Transaction Function” que muestra el listado de funciones transaccionales, véase en la figura: **Figura 5.17**.

Debajo de las funciones de datos tenemos el resultado del cálculo en puntos función de todos los elementos que se añadan en el FUR en el que nos encontramos. Se inicializará la cuenta a 0 FPs en el momento en el que se añada un FUR a la herramienta y se recalcula cuando se van añadiendo elementos.

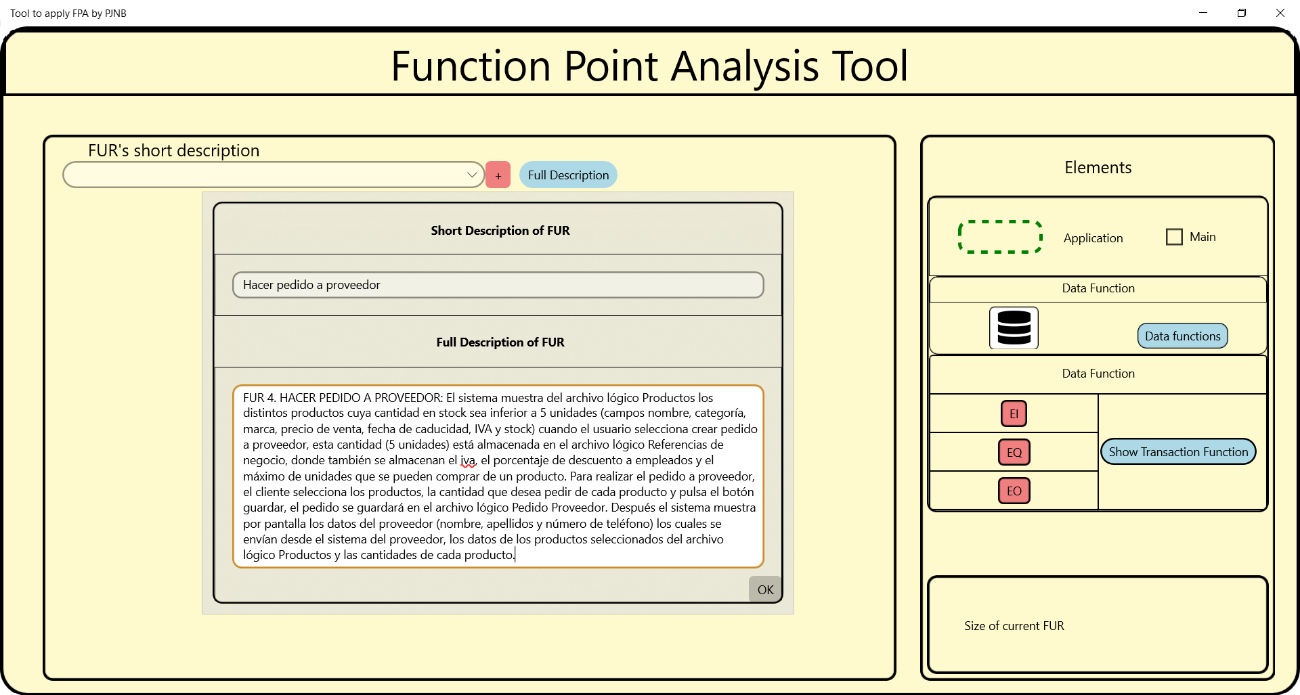


**Figura 5.1.** Pantalla Principal

Una vez hemos explicado la pantalla principal, se va a explicar las distintas pantallas flotantes que tenemos en la aplicación. Vamos a ir explicándolas en orden de cómo se iría navegando para poder realizar un cálculo.

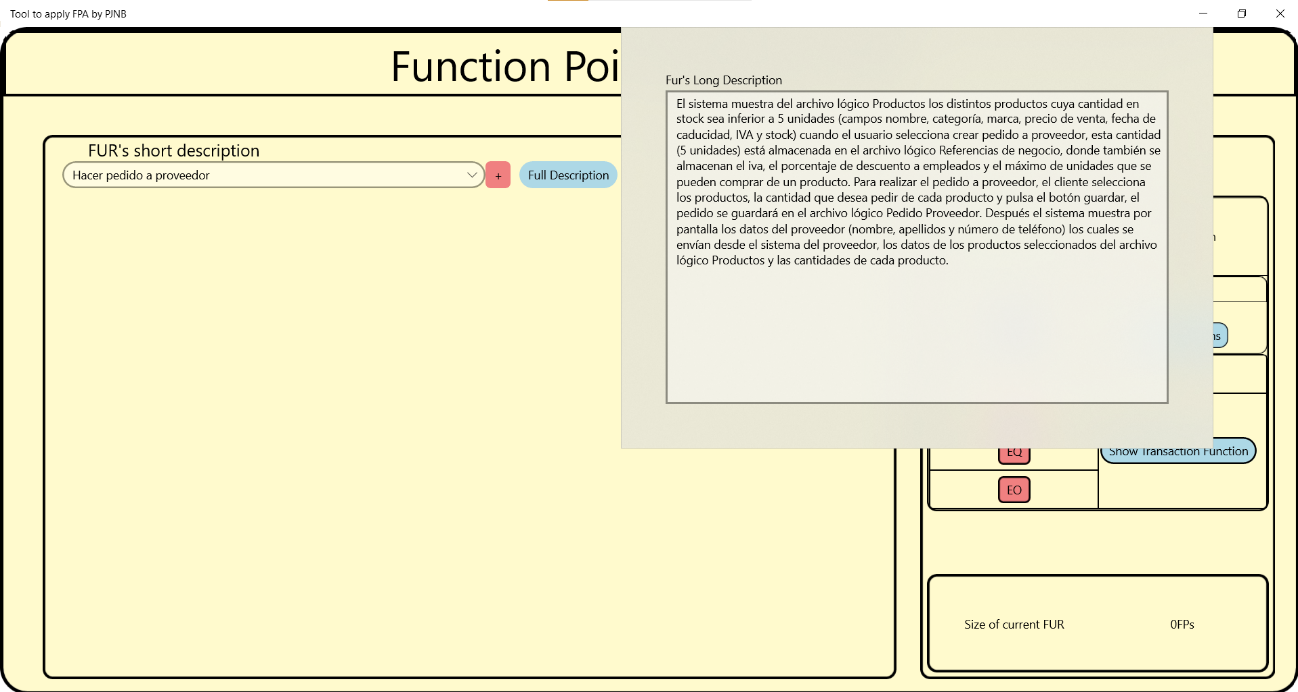
Primero lo que debemos hacer es crear un FUR, por lo que como hemos mencionado antes desde la pantalla principal clicamos el botón “+” para llegar a la pantalla flotante de crear FUR. Esta pantalla flotante la podemos visualizar en la figura: **Figura 5.2**, en esta pantalla se encuentran dos cajas de texto y un botón, la primera caja de texto la utilizamos para escribir una descripción corta del FUR que vamos a añadir. La segunda caja de texto, donde dice “full Description of the FUR”, vamos a añadir la descripción completa del FUR.

Para finalizar en esta pantalla hay un botón “OK”, que como indica su nombre sirve para confirmar las descripciones añadidas y para añadir el nuevo FUR al listado de FURs, es decir, al desplegable de la pantalla principal.



**Figura 5.2.** Crear FUR

Para poder visualizar la descripción completa del FUR actual, es decir, del FUR que esta seleccionado en el desplegable de la pantalla principal, debemos de clicar en el botón “Full Description” y nos llevara a la pantalla flotante que se muestra en la figura: **Figura 5.3**. Esta pantalla es muy sencilla, solamente tiene un bloque de texto con la descripción completa del FUR.



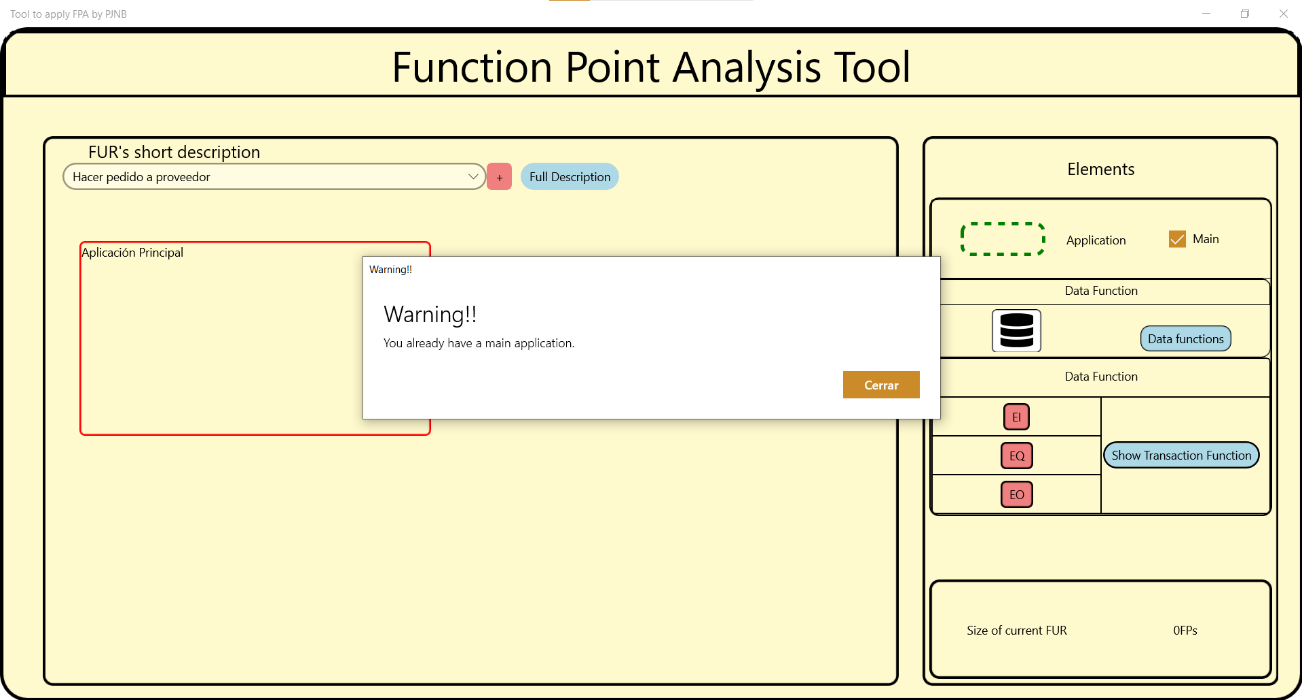
**Figura 5.3.** Mostrar descripción larga de FUR

Una vez tenemos el FUR creado, podemos añadir diferentes elementos. Primero se debe de añadir una aplicación, para ello se debe de clicar el rectángulo de la pantalla principal. Debemos de tener en cuenta que si el checkbox llamado Main esta clicado quiere decir que se añade una aplicación principal (rojo) y si no lo esta se añade una aplicación secundaria (verde). Cuando clicamos el rectángulo aparece una pantalla flotante, está la podemos visualizar en la figura: **Figura 5.4**. Esta pantalla flotante tiene una caja de texto y un botón. La caja de texto sirve para indicar el nombre de la aplicación que se va a añadir y el botón, “OK”, sirve para crearla en el canvas. Si la aplicación que queremos añadir es principal y ya hay una creada que también es principal, cuando hacemos click en el botón “OK”, nos aparecerá un mensaje indicándonos que ya hay una aplicación principal, este mensaje lo podemos ver en la figura: **Figura 5.5**.

Interfaz de usuario gráfica, Texto

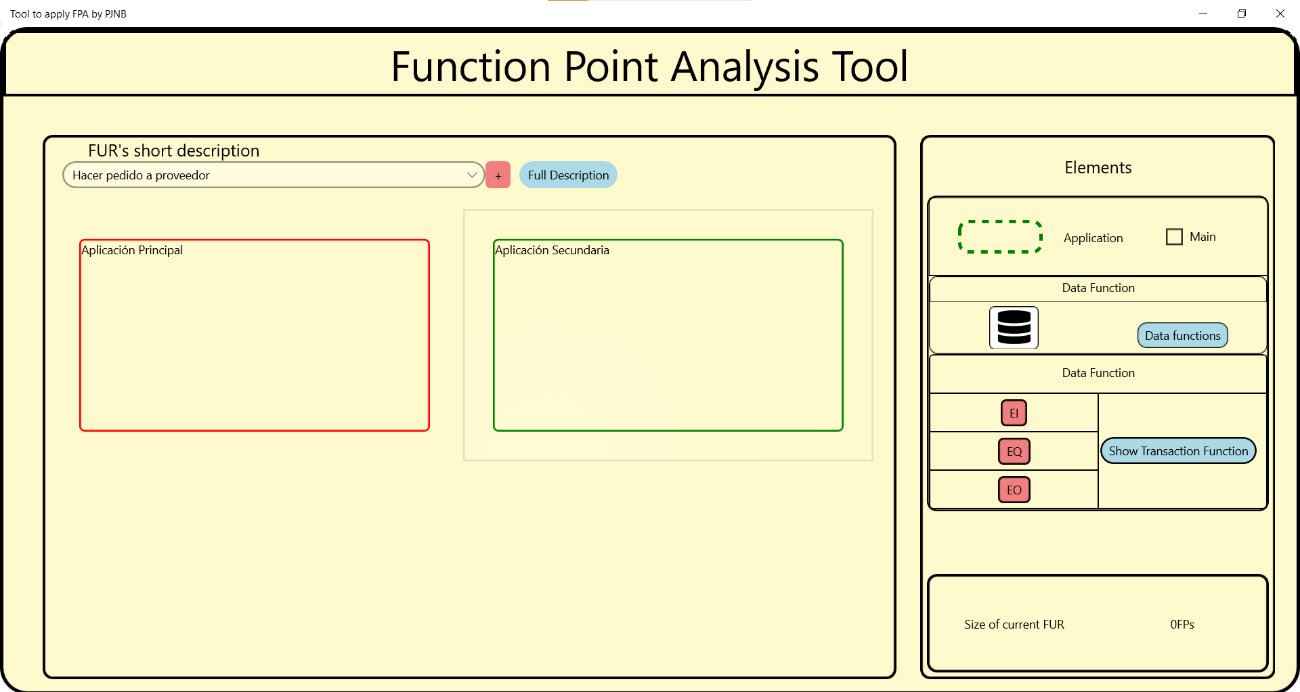
Descripción generada automáticamente

**Figura 5.4.** Crear aplicación principal



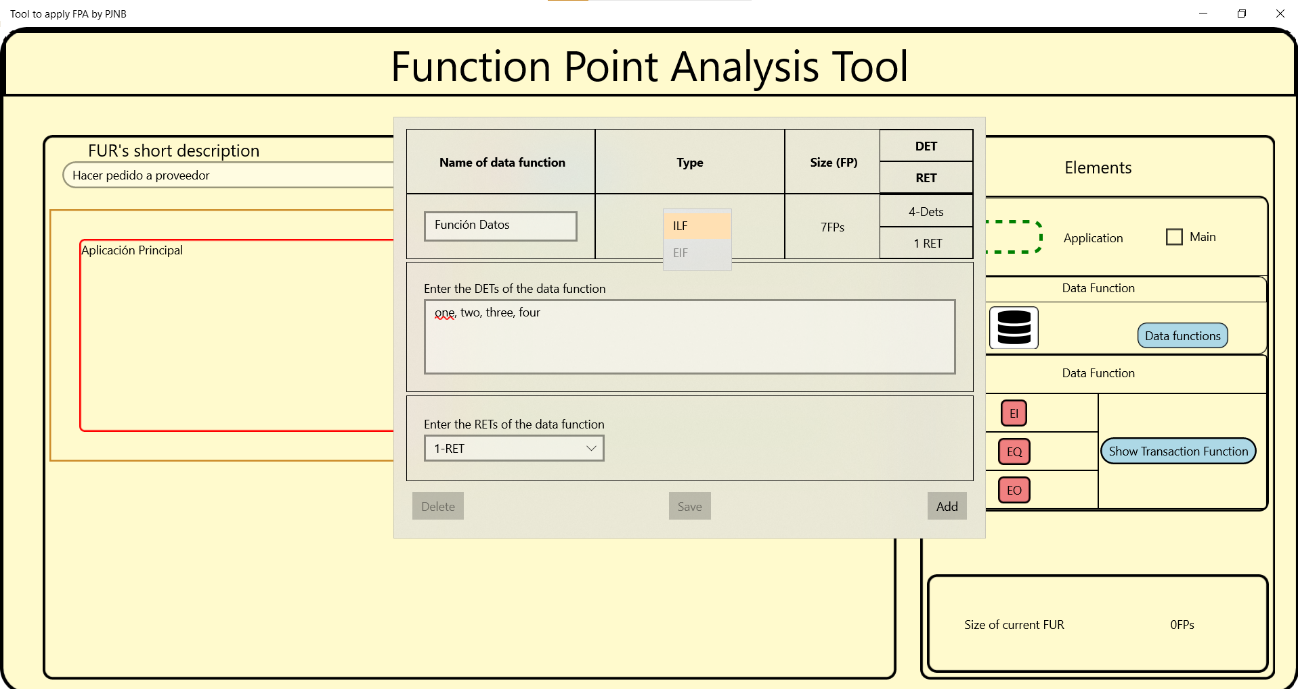
**Figura 5.5**. Mensaje aplicación principal ya existe.

En la figura: **Figura 5.6**, podemos ver un ejemplo de un FUR con dos aplicaciones, una principal que es la de color rojo denominada aplicación principal, y otra de color verde que es secundaria y denominada aplicación secundaria.



**Figura 5.6.** Grid con dos aplicaciones

Al tener una aplicación creada en el canvas, ya se puede añadir las funciones de datos. Para añadir una función de datos primero se debe de clicar la aplicación donde queremos que se aloje. Considerando que una aplicación está marcada, se procede a clicar en el botón que tiene una imagen de una base de datos situado en la pantalla principal, este botón como se ha mencionado antes nos lleva a la pantalla emergente de creación de función de datos. Para poder explicar la pantalla de crear funciones de datos vamos a hacer uso de la figura: **Figura 5.7**. En esta pantalla emergente podemos ver tres apartados diferentes con contenido en cada uno de ellos y en la parte inferior tres botones. Para entenderlo mejor, nos vamos a fijar primero en la parte de abajo de la pantalla, donde tenemos un desplegable que se utiliza para indicar la cantidad de RETs que tiene la función de datos, para ello el usuario deberá seleccionar la cantidad que va desde el 1 al 5 y también se puede seleccionar mayor que 5. En la parte intermedia tenemos una caja de texto donde se deben de añadir los DETs de la función de datos, estos deben de estar separados por coma para que puedan ser contabilizados correctamente. En la parte superior tenemos 3 columnas y 2 filas, la primera fila indica las descripciones de los elementos de la segunda fila. fijándonos ahora en la segunda fila; la primera columna tiene una caja de texto en la que se debe de indicar el nombre de la función de datos que vamos a crear. En la segunda columna hay un desplegable con 2 valores, ILF y EIF. Este desplegable tendrá marcado por defecto una de las dos opciones dependiendo si la función de datos se está creando en una aplicación principal o en una secundaria, en el caso de principal estará marcado ILF y si es secundaria EIF. La tercera columna tiene 3 apartados, el primero indica los puntos función de la función de datos, este es calculado en base a los otros dos apartados, que son el número de DETs y RETS. Los DETs son calculados haciendo uso de la coma en la caja de texto y los RETs proceden de lo que se ha marcado en el desplegable. El cálculo de los FPs se realiza haciendo uso de la tabla de complejidad de las funciones de datos que es la siguiente: **Tabla 6.1** y de las tablas de conversión de la complejidad que son: **Tabla 6.2** y **Tabla 6.3**. Para finalizar esta pantalla tiene 3 botones en la parte inferior, dos de ellos están deshabilitados debido a que solo se habilitan cuando se accede desde el modo edición de funciones de datos, el botón restante es “add” y sirve para añadir la función de datos a la aplicación seleccionada. Un ejemplo de función de datos añadida a una aplicación lo podemos ver en la figura **Figura 5.8**. Como podemos ver se ha añadido una función de datos a la aplicación principal del FUR, com es una aplicación principal, la función de datos es un ILF. Este se muestra con un botón con una imagen de una base de datos, junto con su nombre, el número de DETs y el número de RETs que se le han indicado en la creación. También podemos apreciar cómo se ha actualizado el bloque de texto de puntos función totales del FUR, que en este caso pasan a ser 7FPs.



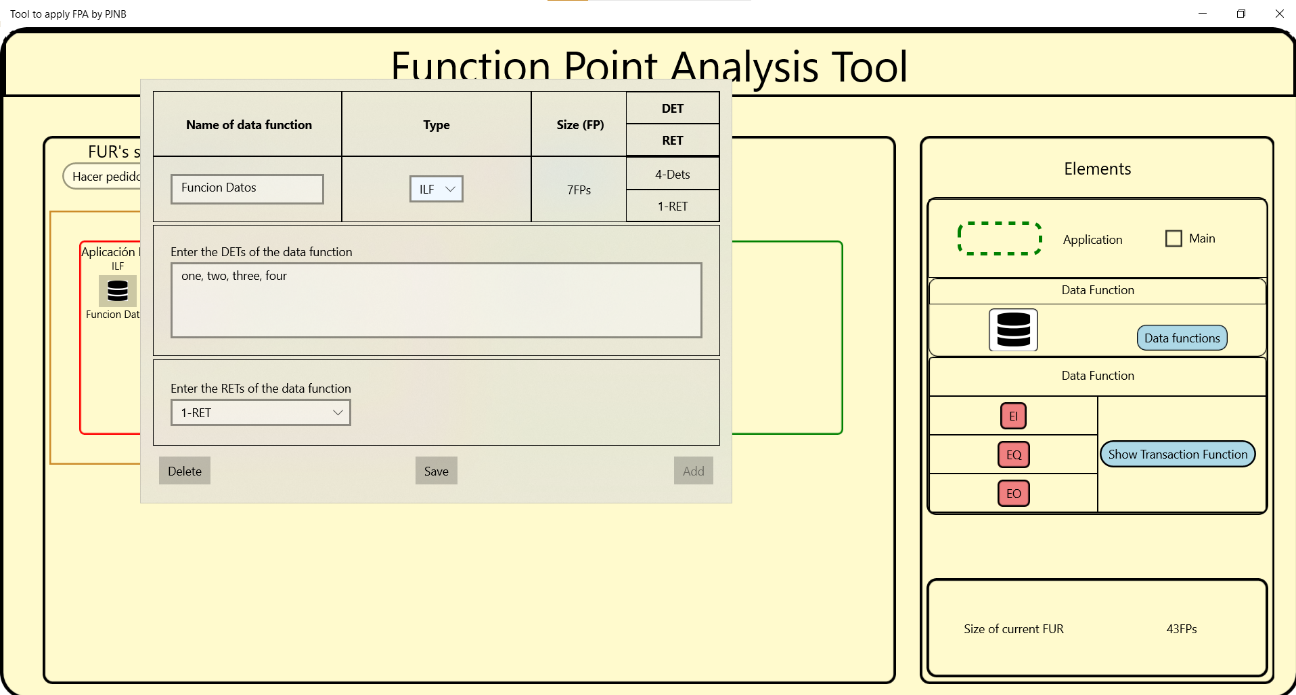
**Figura 5.7**. Crear función de datos

Interfaz de usuario gráfica, Diagrama

Descripción generada automáticamente

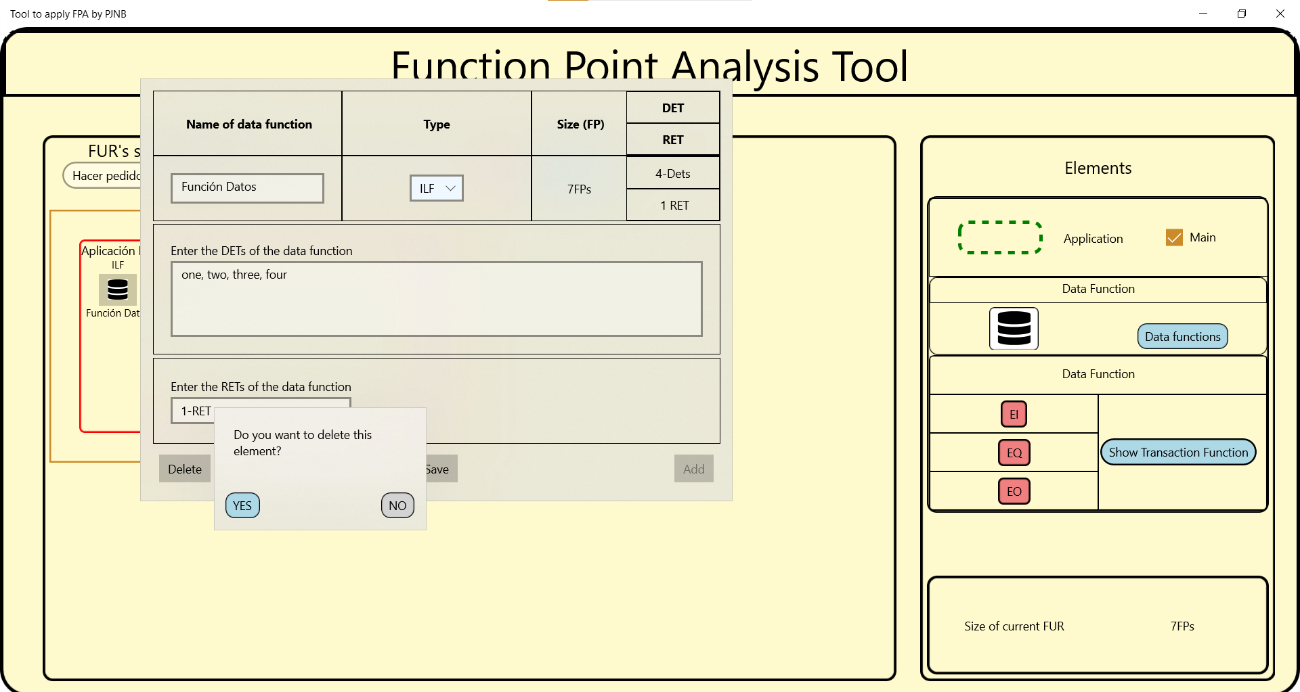
**Figura 5.8.** Función de datos añadida a la aplicación

Una vez tenemos creada una función de datos, podemos editarla o borrarla. Para ello debemos de pulsar el botón de la función de datos que se ha creado, es decir, el botón que se encuentra dentro de una aplicación. Este botón nos llevara a la misma pantalla que cuando se iba a crear, pero en este caso los botones de edición y borrar están habilitados, mientras que el de añadir pasa a estar deshabilitado, como podemos ver en la figura: **Figura 5.9**.



**Figura 5.9.** Edición o eliminación de función de datos seleccionada

En el caso de querer editar la función de datos, se hace directamente en la misma pantalla que cuando se crea y una vez realizadas las modificaciones se hace click en el botón “save” para poder guardar los cambios. En cambio si se quiere borrar la función de datos, debemos de hacer click en el botón “Delete”, pero no se borrará directamente, sino que nos aparecerá una nueva pantalla emergente como podemos apreciar en la figura: **Figura 5.10**, que sirve para verificar si estamos seguros de querer eliminar la función de datos seleccionada, para ello tiene dos botones uno para proseguir con la eliminación, “yes” y otro para no eliminarla que es el botón “no”.



**Figura 5.10.** Verificar eliminación de función de datos

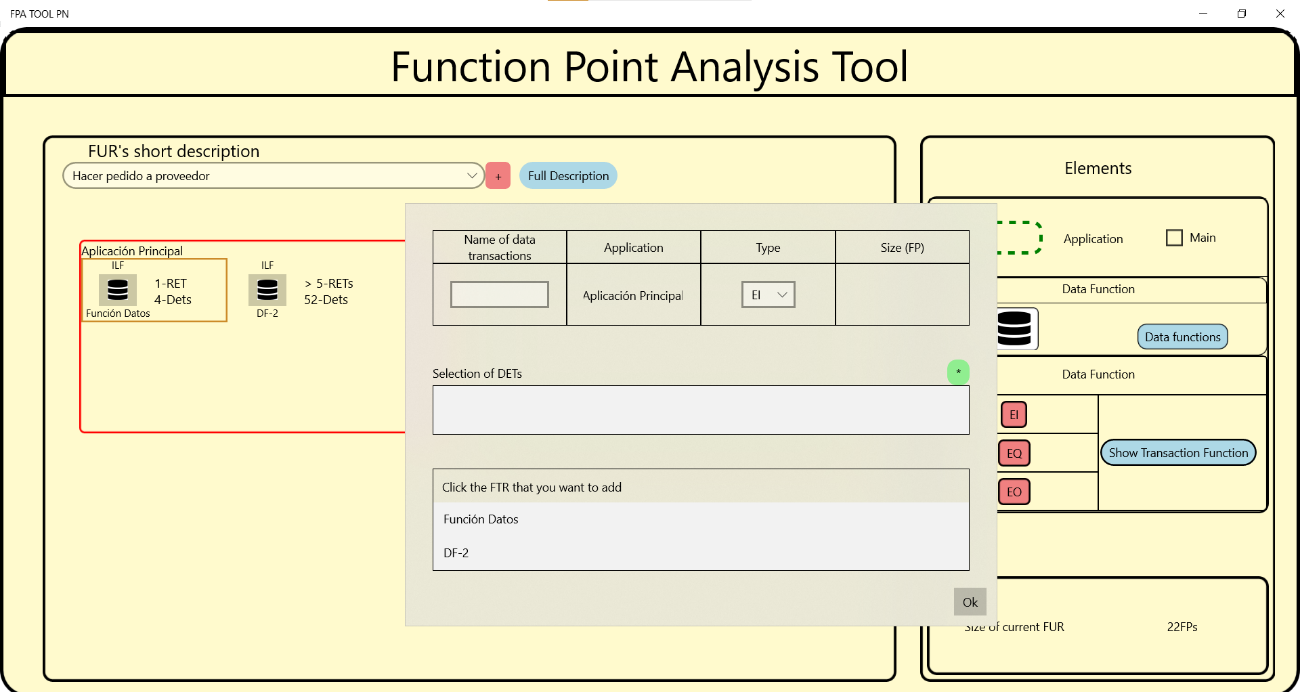
En cualquier momento, desde el inicio de la aplicación, podemos ver el listado funciones de datos creadas, para ello se debe de clicar el botón “Data functions” de la pantalla principal, que nos mostrara una lista de todas las funciones de datos con 4 campos, como podemos apreciar en la figura: **Figura 5.11**. El primer campo representa el nombre de la función de datos, el segundo el tipo, el tercero el tamaño en puntos función y el último la aplicación a la que pertenece.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

**Figura 5.11:** Listado de Funciones de datos de FUR actual

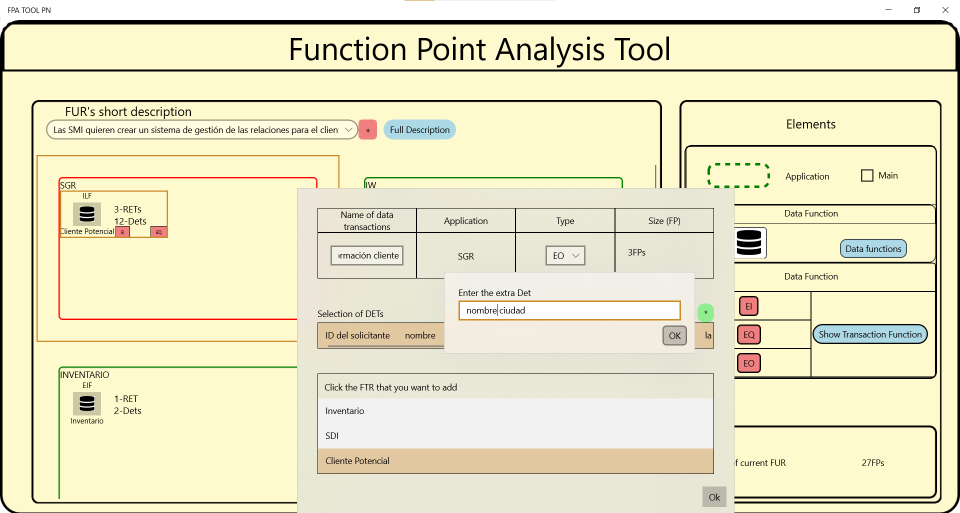
Cuando se tiene una o más funciones de datos creadas, se puede crear funciones transaccionales. Para poder añadir una función transaccional, debemos de tener seleccionada la función de datos a la que queremos vincular la función transaccional. Después de seleccionar la función de datos, podemos hacer click en uno de los tres botones rojos de la pantalla principal en la parte de funciones transaccionales. Los tres botones nos llevan a la misma pantalla emergente, está la podemos visualizar en la figura: **Figura 5.12**. La pantalla de crear función transaccional es muy parecida a la de crear función de datos. En esta pantalla también tenemos 3 partes, si nos fijamos en la parte de abajo podemos ver un listado con los FTR que tenemos en este FUR. Este listado hace referencia a las funciones de datos que hay en el FUR. Por defecto no hay ningún FTR marcado en el listado. Al hacer click en un FTR de la lista, directamente nos aparece en el listado de la parte intermedia los DETs correspondientes a ese FTR, seleccionados por defecto. El máximo número de FTR que podemos añadir a una función transaccional estará limitado por la cantidad de funciones de datos que tengamos en el FUR y por consiguiente en el listado de FTRs. En el apartado intermedio se muestran inicialmente los DETs seleccionados a partir de los FTRs añadidos, pero si se quiere se puede deseleccionar cualquiera de los DETs del listado. También tenemos un botón “\*” que nos permite añadir Dets extra que no provienen de ningún FTR, para ello accede a la pantalla emergente que podemos ver en la figura: **Figura 5.13**. En esta pantalla tenemos una caja de texto donde añadimos el Det y lo verificamos con el botón “OK”. Al darle a “OK” el DET es añadido al listado de DETs de la pantalla de creación de funciones transaccionales. En la parte de arriba de la pantalla creación de funciones transaccionales tenemos 4 columnas. La primera corresponde con el nombre de la función transaccional, la segunda con la aplicación en la que se está creando, la tercera con el tipo de transacción (EI, EQ, EO) y la cuarta representa los puntos función de la función transaccional. Estos puntos función estarán calculados en relación con la cantidad de FTRs seleccionados y la cantidad de DETs seleccionados. Para los cálculos se utilizan las tablas de complejidad; **Tabla 6.4** y **Tabla 6.5**. Cuando se tiene la complejidad de la función transaccional, se traduce en puntos función, haciendo uso de las tablas; **Tabla 6.6** y **Tabla 6.7**. Para verificar la creación de la función transaccional con los datos seleccionados previamente, se utiliza el botón “OK”, lo que hace que se actualice la cuenta de FPs totales del FUR, en la pantalla principal. Así mismo, aparece un botón indicando el tipo de función transaccional en la función de datos a la que se ha añadido. En la figura: **Figura 5.17** podemos ver en la aplicación principal diferentes funciones de datos que contienen varias funciones transaccionales.



**Figura 5.12**. Crear Función Transaccional

Al tener la función transaccional creada, se puede proceder tanto a eliminarla como a modificarla. Para ello se debe hacer click en el botón de la función transaccional que queremos modificar o borrar. Este nos lleva a la pantalla emergente de edición y borrado, podemos verlo en la figura: **Figura 5.14**. En esta pantalla tenemos una lista con las funciones transaccionales de un tipo (EI, EO o EQ) que tiene una función de datos. En esta pantalla se muestra el nombre de cada función transaccional, el tipo, el tamaño en FPs y la aplicación a la que pertenece. Si se selecciona una función transaccional y se le da al botón de Delete, nos aparece otra pantalla emergente para verificar si realmente queremos borrarla, como se aprecia en la figura: **Figura 5.15**.

En caso de querer editarla debemos de hacer click en el botón “edit” y nos llevara a otra pantalla emergente que es igual que la de creación de función transaccional, aunque ahora aparecen dos nuevos botones como podemos apreciar en la figura: **Figura 5.16**. Los nuevos botones son “Load” y “save”, el botón de “load” sirve para seleccionar todas las FTRs que estaban seleccionadas en esa función transaccional, ya que al darle a “edit” no se cargan correctamente. El botón de “save” sirve para guardar los cambios que se realizan.

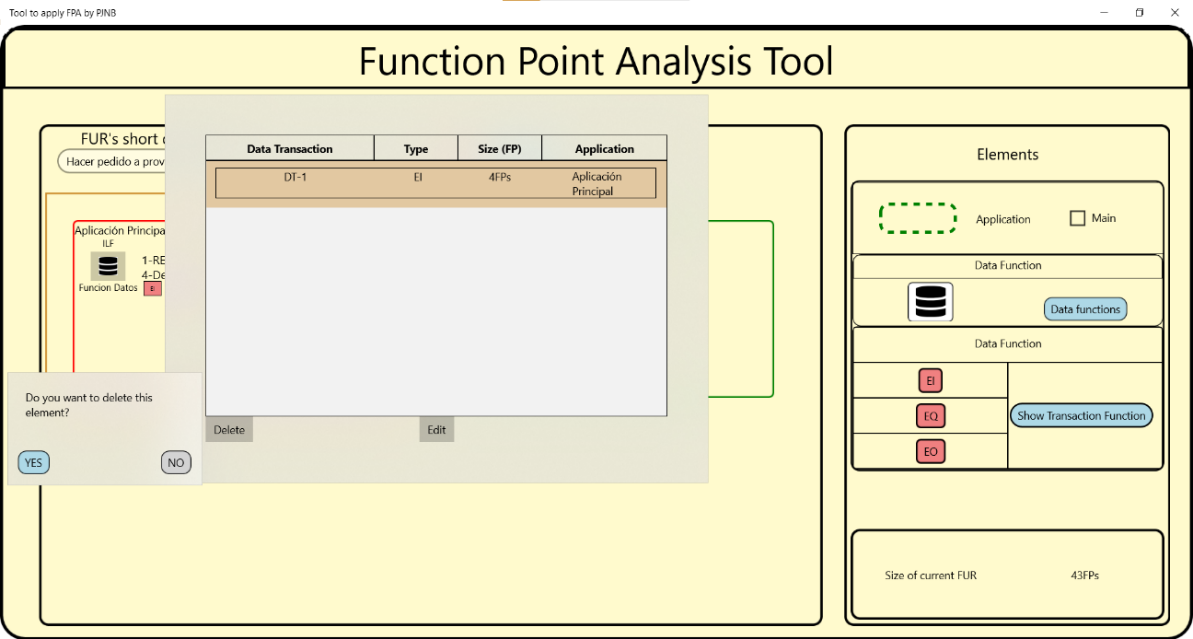


**Figura 5.13.** Extra Det

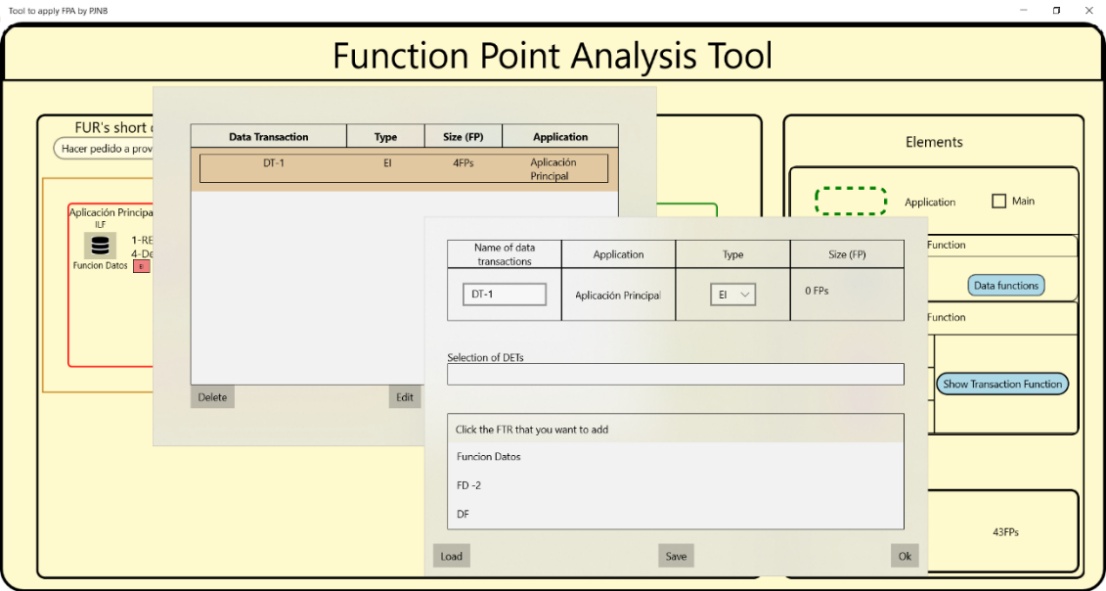
Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

**Figura 5.14:** Editar o Borrar Función Transaccional



**Figura 5.15.** Borrar Función Transaccional



**Figura 5.16.** Editar Función Transaccional

Al igual que para las funciones de datos, también existe una lista de todas las funciones transaccionales del FUR actual. Para ello se debe de pulsar el botón “Show Transaction Functions” de la pantalla principal. Esta lista muestra las funciones transaccionales con 4 características, como se aprecia en la figura: **Figura 5.17**. El primer campo corresponde con el nombre de la función transaccional, el segundo con el tipo, el tercero con los puntos función y el cuarto con la aplicación en la que se encuentra la función transaccional.

Interfaz de usuario gráfica, Tabla

Descripción generada automáticamente

**Figura 5.17.** Lista Funciones Transaccionales

En definitiva la aplicación creada nos permite aplicar el cálculo mediante FPA, aun así hay que matizar que solo nos realiza los cálculos de cada FUR, por lo que para la próxima versión se debería de plantear el realizar el cómputo total de todos los FUR que hay en un software.

# Pruebas

## Introducción

En este capítulo se va a realizar un ejemplo que demuestra el funcionamiento correcto de la herramienta de cálculo de punto función. El ejercicio se va a resolver tanto realizando los cálculos manualmente, como con la herramienta creada. Seguidamente se compararán los valores obtenidos tanto manualmente como a través de la herramienta para comprobar que son los correctos. El ejercicio corresponde a la asignatura de Gestión de Proyectos Software (Bermejo, 2021).

## Ejercicio práctico.

**Case Study 5, del Practice Exam 1 de la CFPS Examination Guide**.

Case Study 5, del Practice Exam 1 de la CFPS Examination Guide.

SME está implementando un Sistema de Gestión de Relaciones (SGR) al cliente. La Aplicación Información Web (IW), una aplicación existente, tendrá que enviar información al SGR cada noche recuperando todas las solicitudes de información (SDI) presentados ese día y actualmente mantenidos en el archivo lógico SDI dentro de la aplicación IW.

La siguiente información se envía en esta transacción diaria: ID del solicitante; nombre, segundo nombre, y apellido; organización del solicitante; la dirección del solicitante (dirección, ciudad, estado y código postal); fecha de la solicitud; artículos solicitados; y las cantidades de los artículos solicitados.

La aplicación SGR validará y procesará el envío diario en un nuevo archivo lógico Cliente Potencial. Se generarán informes separados por ciudad cada mañana por parte de la Aplicación SGR y se entregan a los coordinadores de cada ciudad. La impresión contendrá toda la información del archivo lógico Cliente Potencial, así como el número

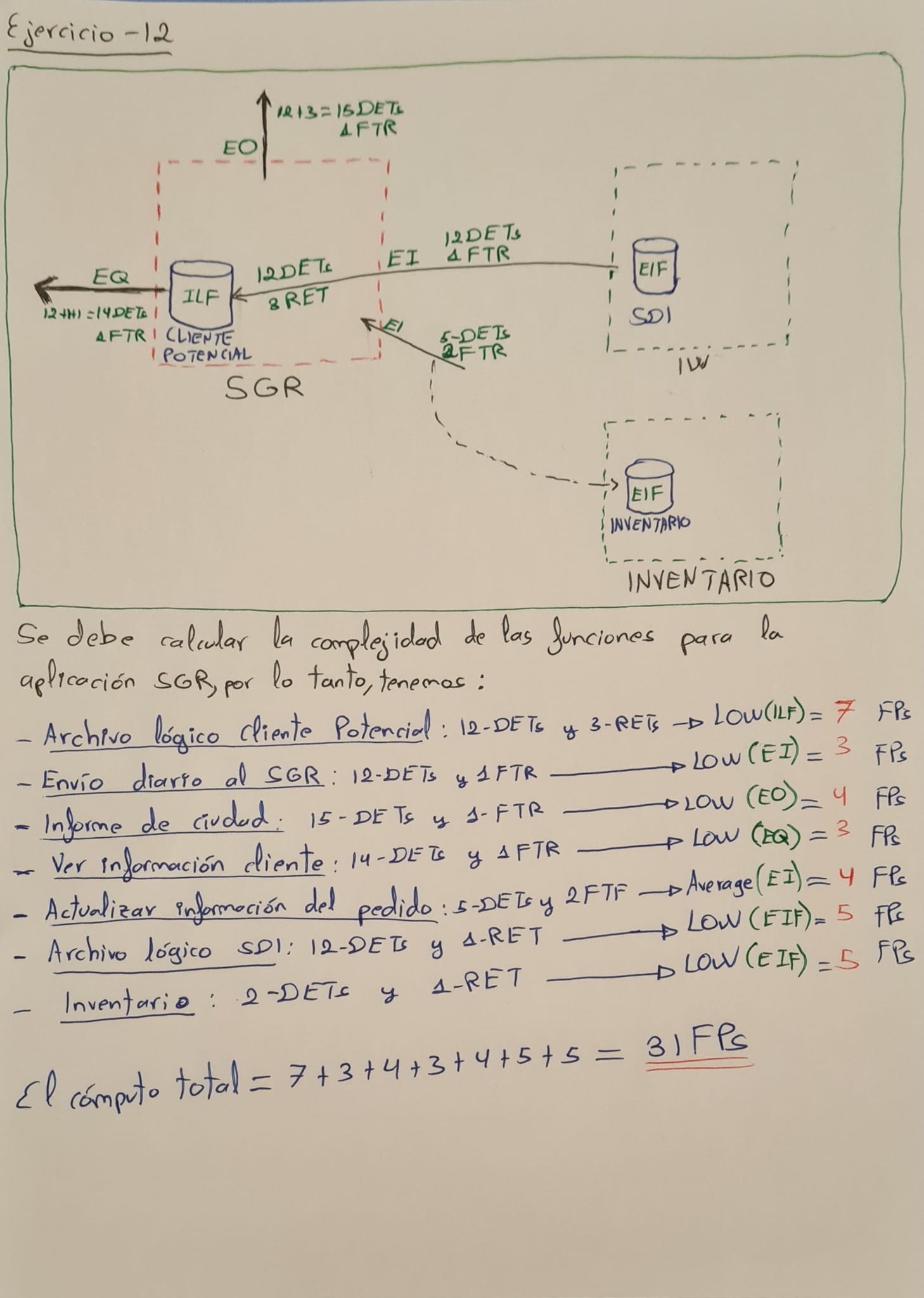
total de solicitudes de información, que se calcula en el momento que se produce el informe. El código de ciudad y el nombre, recuperados de una tabla de códigos, también se imprimirán en cada informe.

Cada coordinador tendrá la posibilidad de recuperar a través de la pantalla toda la información mantenida en el archivo lógico Cliente Potencial introduciendo el ID de solicitante y la tecla de acción; si no se encuentra el ID del solicitante, se devolverá un mensaje de error. Mediante la misma pantalla, el coordinador puede cambiar los

artículos y / o cantidades utilizando el ID de solicitante y una tecla de función preasignada; un mensaje de error puede ser devuelto si el artículo solicitado recientemente no está contenido en el archivo lógico Inventario, mantenido por la aplicación Inventario; en otro caso se envía un mensaje de confirmación.

Identificar la complejidad de las funciones para la aplicación SGR

El ejercicio resuelto manualmente lo podemos ver en la figura; **Figura 6.1**, donde se dibujan todos los elementos y se realizan los cálculos necesarios.



**Figura 6.1.** Ejercicio 12 manualmente

Tanto para calcular manualmente, como para que lo calcule la herramienta, el ejercicio expuesto en el cuadro de texto se debe desglosar para poder entenderlo correctamente. Para ello se va a dividir el ejercicio a razón de las funciones de datos y transaccionales.

– **Archivo lógico Cliente Potencial (ILF):**  Contiene la información recibida del SDI, es decir, 12 DETs debido a los atributos y 3 RETs, debido a que hay dos atributos que se refieren a un conjunto.

– **Archivo lógico SDI:**  Contiene los atributos: ID del solicitante; nombre, segundo nombre, y apellido; organización del solicitante; la dirección del solicitante (dirección, ciudad, estado y código postal); fecha de la solicitud; artículos solicitados; y las cantidades de los artículos solicitados.

– **Archivo lógico Inventario (EIF):**  Contiene los artículos y cantidades.

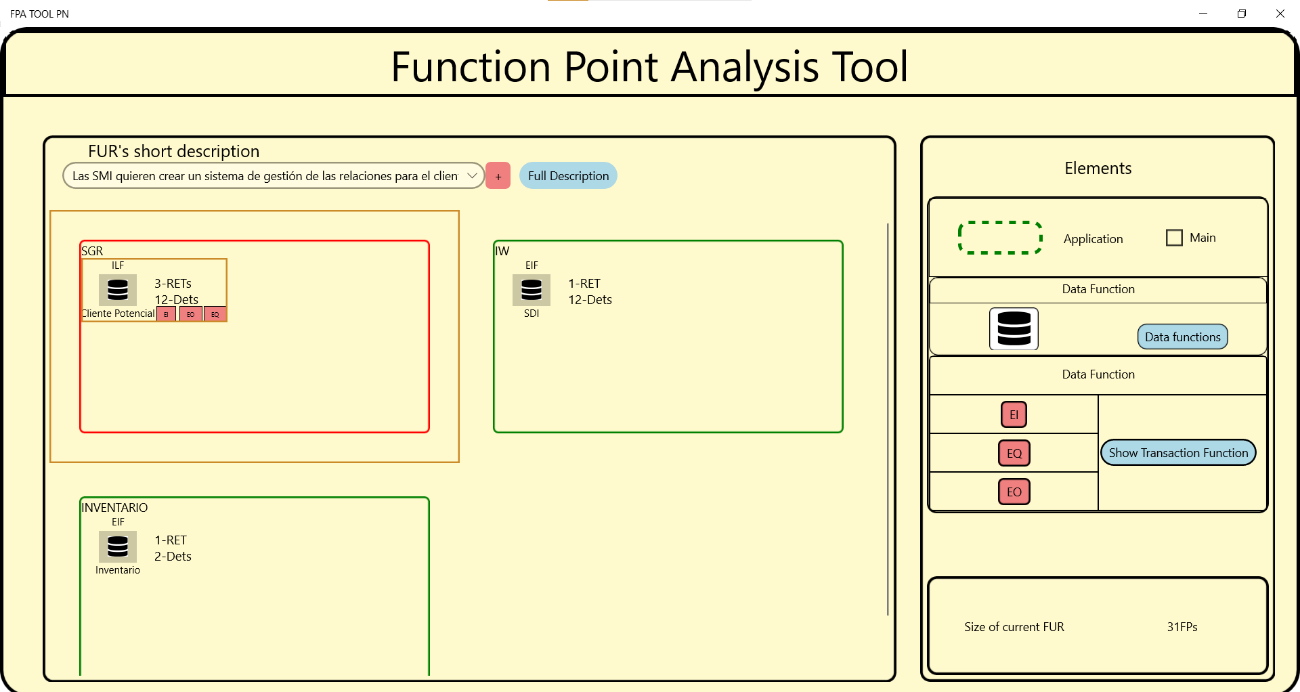
– **Envío diario al SGR (EI):**  El envío diario desde el SDI al SGR de todos los atributos, estos atributos entran a la aplicación principal por lo que es un EI.

– **Informe de Ciudad (EO):**  Informe que se genera desde el SGR, que contiene el número total de solicitudes, toda la información del Cliente potencial, el código de la ciudad y el nombre de la ciudad. Como al generar el informe se deben de calcular la cantidad de solicitudes, esta transacción es de tipo EO.

– **Ver información del Cliente (EQ):**  Cualquier cliente puede recuperar información que esta mantenida en el archivo lógico Cliente Potencial, haciendo uso de su ID y presionando OK. Al no realizar ningún calculo al generar la información se considera una función transaccional de tipo EQ

– **Actualizar información del pedido (EI):**  el cliente puede realizar cambios respecto a los artículos y sus cantidades. Se considera la transacción de tipo EI.

Tanto los cálculos manuales como de la aplicación coinciden. En la figura: **Figura 6.2**, podemos visualizar todos los elementos mencionados en el enunciado del ejercicio. Hay una aplicación principal y dos aplicaciones secundarias. Cada una ellas tienen una función de datos asignada. En el caso de la aplicación principal, su función de datos contiene 4 funciones transaccionales. Una vez están todos los elementos añadidos podemos ver a la derecha de la figura el cálculo total del FUR actual.



**Figura 6.2.** Ejercicio 12 elementos creados en la herramienta

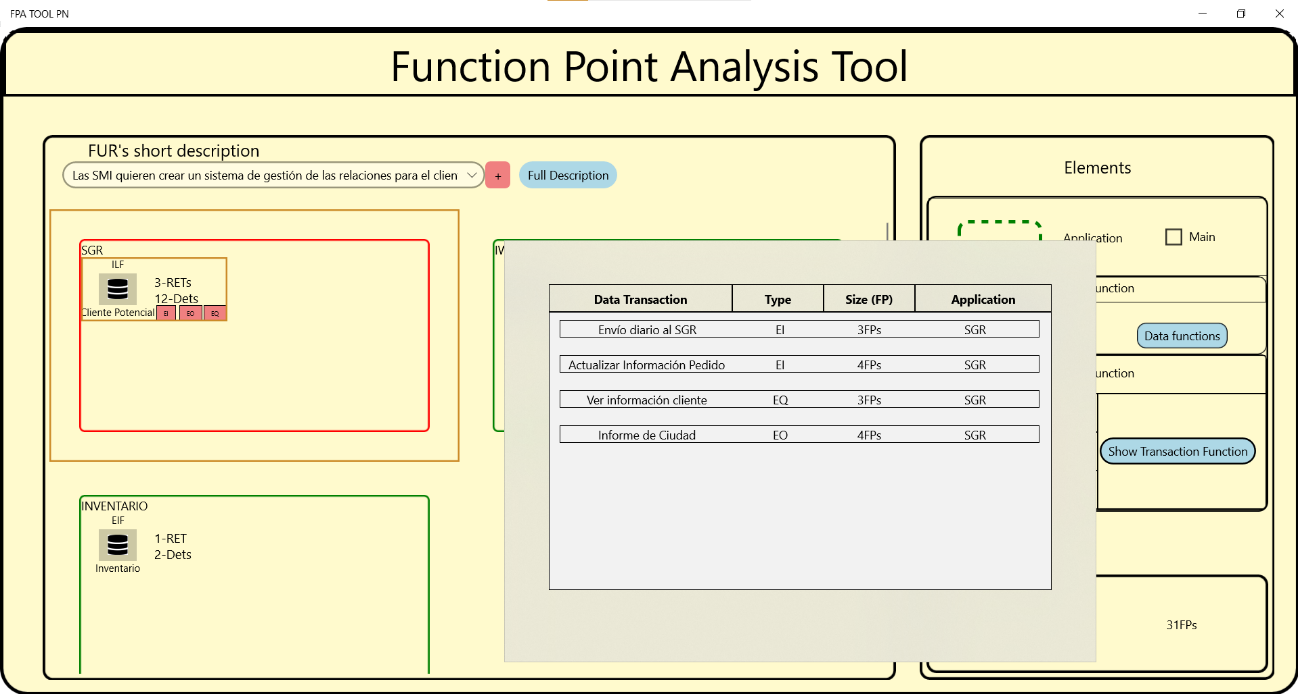
En la figura **Figura 6.3**, podemos ver el listado de las funciones de datos que han sido creadas en este FUR. Cada función de datos aparece con su nombre, tipo, tamaño y aplicación a la que pertenecen.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

**Figura 6.3.** Ejercicio 12 Listado de Funciones de Datos

En la figura: **Figura 6.4**, podemos ver el listado de las funciones transaccionales que han sido creadas en este FUR. Cada función transaccional aparece con su nombre, tipo, tamaño y aplicación a la que pertenecen.



**Figura 6.4.** Ejercicio 12 Listado de Funciones Transaccionales

Tanto para el cálculo manual como el cálculo que la herramienta realiza, depende de la complejidad de las funciones. Para poder calcular las funciones de datos en función a los DETs y RETs que tienen, la herramienta hace uso de las tablas de complejidad y conversión para funciones de datos, estas son: **Tabla 6.1**, **Tabla 6.2** y **Tabla 6.3**.

Así mismo, para el cálculo de las funciones transaccionales, la herramienta realiza los cálculos en función de los DETs y FTRs que cada función tiene y hace uso de las tablas y complejidad de funciones transaccionales, estas son:**Tabla 6.4**, **Tabla 6.5**, **Tabla 6.6** y **Tabla 6.7**.

Tanto la herramienta como el cálculo manual nos ha llevado al mismo resultado, este se ha calculado sumando los puntos función de las funciones de datos y de las funciones transaccionales.

Las funciones de datos se calculan en base a las tablas de complejidad y conversión de los ILF y EIF, por lo tanto:

– **Cliente Potencial (ILF):** 12 Dets y 3 Rets = LOW = 7 FPs

– **SDI (EIF):** 12 Dets y 1 Ret = LOW = 5 FPs

– **Inventario (EIF):**  2 Dets y 1 Ret = LOW = 5 FPs

Las funciones transaccionales se calculan en base a las tablas de complejidad y conversión de los EI, EO y EQ, por lo tanto:

– **Envío diario al SGR (EI):** 12 Dets y 1 FTR = LOW = 3 FPs

– **Informe de ciudad (EO):** 15 Dets y 1 FTR = LOW = 4 FPs

– **Ver información del cliente (EQ):**  14 Dets y 1 FTR = LOW = 3 FPs

– **Actualizar información del pedido (EI):** 5 Dets y 2 FTR = AVERAGE = 4 FPs

Por lo tanto el cómputo total es la suma de todas las funciones:

Funciones de datos:

Funciones Transaccionales:

TOTAL de puntos función del ejercicio:

**Tabla 6.1.** Matriz de Complejidad para ILF y EIF

|  |  |  |  |
| --- | --- | --- | --- |
|  | 1-19 DETs | 20-50 DETs | >50 DETs |
| 1 RET | Low | Low | Average |
| 2-5 RETs | Low | Average | High |
| >5 RETs | Average | High | High |

**Tabla 6.2**. Traducción de complejidad del ILF

|  |  |
| --- | --- |
| Complejidad | Puntos Función |
| Low | 7 |
| Average | 10 |
| High | 15 |

**Tabla 6.3**. Traducción de complejidad del EIF

|  |  |
| --- | --- |
| Complejidad | Puntos Función |
| Low | 5 |
| Average | 7 |
| High | 10 |

**Tabla 6.4.** Matriz de Complejidad para EI

|  |  |  |  |
| --- | --- | --- | --- |
|  | 1-4 DETs | 5-15 DETs | >15 DETs |
| 0-1 FTRs | Low | Low | Average |
| 2 FTRs | Low | Average | High |
| >2 FTRs | Average | High | High |

**Tabla 6.5.** Matriz de Complejidad para EO y EQ

|  |  |  |  |
| --- | --- | --- | --- |
|  | 1-5 DETs | 6-19 DETs | >19 DETs |
| 0-1 FTRs | Low | Low | Average |
| 2-3 FTRs | Low | Average | High |
| >3 FTRs | Average | High | High |

**Tabla 6.6**. Traducción de complejidad del EI y EQ

|  |  |
| --- | --- |
| Complejidad | Puntos Función |
| Low | 3 |
| Average | 4 |
| High | 6 |

**Tabla 6.7**. Traducción de complejidad del EO

|  |  |
| --- | --- |
| Complejidad | Puntos Función |
| Low | 4 |
| Average | 5 |
| High | 7 |

Una vez se ha realizado el ejercicio de forma manual y haciendo uso de la herramienta desarrollada, podemos concluir que es mucho más cómodo realizar la cuenta con una herramienta que automatiza el reconocimiento, y suma de complejidad y tamaño de todos los elementos del proceso FPA. No solo es más cómodo que realizarlo manualmente, sino que también , es más cómodo que realizarlo utilizando el software de LEDA, ya que este realiza los cálculos en un fichero Excel y no muestra el proceso visualmente como si lo hace la herramienta FPA TOOL PN.

# Conclusiones

Después de haber analizado todo el desarrollo realizado para la creación de la aplicación FPA TOOL PN, he entendido la importancia que tiene utilizar una metodología ágil durante el desarrollo de un proyecto. El hecho de entregar valor de forma continua al cliente permite conseguir los objetivos con bastantes garantías. En este proyecto el único desarrollador era el alumno, aun así con los conceptos obtenidos ya estaría capacitado para poder trabajar en cualquier equipo que utilice la metodología ágil y en especial Scrum.

Este proyecto me ha ayudado a refrescar conceptos estudiados durante todo el grado y poder ponerlos en práctica. He comprobado que con lo que he aprendido en los últimos años soy capaz de adaptarme a cualquier problema e intentar buscar las mejores soluciones posibles.

He comprendido que no solo es importante la parte relacionada con el código del proyecto, sino que el proceso de preparación de requisitos y el proceso de desarrollo del marco Scrum son igual de importantes. Todo alrededor del proyecto tiene su importancia y si nos olvidamos de esto puede que no consigamos los objetivos que se han marcado.

De todas las competencias adquiridas en la rama de ingeniería del software he utilizado las siguientes:

– Capacidad de identificar, evaluar y gestionar los riesgos potenciales asociados que pudieran presentarse. El uso de una metodología ágil me ha permitido mediante diferentes sprints entregar valor al cliente desde el principio del desarrollo, esto me ha ayudado a detectar los riesgos que podrían plantearse desde los inicios y por lo tanto poder gestionarlos desde etapas tempranas del proyecto.

– Capacidad para valorar las necesidades del cliente y especificar los requisitos software para satisfacer estas necesidades, reconciliando objetivos en conflicto mediante la búsqueda de compromisos aceptables dentro de las limitaciones derivadas del coste, del tiempo, de la existencia de sistemas ya desarrollados y de las propias organizaciones. Durante todo el desarrollo del proyecto he especificado las necesidades del cliente y he conseguido satisfacer la mayoría de ellas, ya que he aportado valor continuamente durante todo el desarrollo. Se han adaptado los objetivos a razón de lo que aportaba más valor al cliente teniendo en cuenta el coste, el tiempo y el alcance.

Como aportación personal, creo que un producto tendrá mejor calidad si seguimos un proceso adecuado, como ya habíamos experimentado en otros proyectos realizados durante el grado.

Con el desarrollo de este proyecto me considero capaz de acometer cualquier proyecto de una magnitud similar o superior. He conseguido sentar las bases para la realización de proyectos futuros. Así mismo, he ganado la confianza necesaria para ir a cualquier empresa y aplicar los métodos y técnicas aprendidas durante este proyecto.

Bibliografía

Abellán, E. (05 de 03 de 2020). *Scrum. qué es y cómo funciona esta metodología.* Obtenido de https://www.wearemarketing.com/es/blog/metodologia-scrum-que-es-y-como-funciona.html

Abran, A., St-Pierre, D., maya, M., & Desharnais, J.-M. (30 de 10 de 1998). *Full Function Points forEmbedded and Real-Time Software.* Obtenido de https://www.researchgate.net/publication/240816084\_Full\_Function\_Points\_for\_Embedded\_and\_Real-Time\_Software

Administracion. (s.f.). *El parlamento europeo recomienda la metodología IFPUG para el desarrollo de los precios del software.* Obtenido de https://www.ifpug.org/european-parliament-recommends-ifpug-methodology-for-pricing-software-development/?lang=es

Albaladejo. (04 de 08 de 2008). *Qué es SCRUM.* Obtenido de https://proyectosagiles.org/que-es-scrum/

Albaladejo, X. (09 de 03 de 2021). *Proyectos agilers.org.* Obtenido de https://proyectosagiles.org/triangulo-hierro/

Anzola, J. C. (2011). *Pfa CLM 2011. Proceso de Desarrollo de software.* Obtenido de https://es.scribd.com/document/358380224/PFA-CLM-2011

Atlassian. (25 de 07 de 2018). *Jira Software.* Obtenido de https://www.atlassian.com/es/software/jira/guides/getting-started/overview#jira-software-hosting-options

Balsamiq. (2008). *Quick and Easy Wireframing Tool.* Obtenido de https://balsamiq.com/wireframes/?gclid=CjwKCAjwoZWHBhBgEiwAiMN66UZ97dAXlQqIVUnBy49Xiv-glhfLkPxhf6z3oHJga-lSxp601erMuRoCqsoQAvD\_BwE

Bermejo. (2021). *Tamaño del software: Puntos Función según FPA.* Obtenido de https://campusvirtual.uclm.es/pluginfile.php/283997/mod\_resource/content/6/Tema%203%20-%20Tama%C3%B1o%20Funcional%20con%20FPA.pdf

Bermejo, P. (5 de 10 de 2020). *Tamaño Software: Longitud, Complejidad y Funcionalidad.* Obtenido de https://campusvirtual.uclm.es/pluginfile.php/283996/mod\_resource/content/3/Tema%202%20-%20Tama%C3%B1o%20del%20Software.pdf

Borade, J. (08 de 10 de 2018). *Software Project Effor and Cost Estimation Techniques.* Obtenido de https://www.researchgate.net/profile/Jyoti-Borade-2/publication/313243865\_Software\_Project\_Effort\_and\_Cost\_Estimation\_Techniques/links/5bb7020ea6fdcc9552d3e638/Software-Project-Effort-and-Cost-Estimation-Techniques.pdf

Buglione, L. (18 de 09 de 2019). *40 Years of Function Points: Past, Present, Future.* Obtenido de https://www.ifpug.org/40-years-of-function-points-past-present-future/

Casillas, J. A. (01 de 02 de 2019). *¿Que son los puntos de historia?* Obtenido de https://muyagile.com/que-son-los-puntos-de-historia/

Cosmic. (07 de 07 de 2020). *he standard methodology for sizing software.* Obtenido de https://cosmic-sizing.org/cosmic-sizing/intro/

Cuesta Meza, A., López Trujillo, M., & Joyanes Aguilar, L. (16 de 08 de 2019). *Métodos de métricas orientados a los puntos de función.* Obtenido de https://core.ac.uk/download/pdf/230227093.pdf

*Estimation Techniques - Wideband Delphi.* (2021). Obtenido de https://www.tutorialspoint.com/estimation\_techniques/estimation\_techniques\_wideband\_delphi.htm

Eudaimonia. (24 de 01 de 2020). *Scrum Process.* Obtenido de https://dev.to/eudaimonia\_ar/improve-your-performance-with-scrum-methodology-idg

Gómez. (2013). *Métodos de Medición en Puntos Función (III): Mk-II FPA.* Obtenido de https://www.laboratorioti.com/2013/01/28/metodos-de-medicion-en-puntos-funcion-iii-mk-ii-fpa/

Gómez, J. (14 de 02 de 2013). *Método de Estimación Puntos Casos de Uso (Use Case Points).* Obtenido de http://www.laboratorioti.com/2013/02/14/metodo-de-estimacion-puntos-casos-de-uso-use-case-points/

Gómez, J. (2013). *Métodos de Medicion en Puntos Función (II): Nesma.* Obtenido de https://www.laboratorioti.com/2013/01/21/metodos-de-medicion-en-puntos-funcion-ii-nesma/

Gómez, J. (2013). *Métodos de Medición en Puntos Función (IV): COSMIC FFP.* Obtenido de https://www.laboratorioti.com/2013/01/30/metodos-de-medicion-en-puntos-funcion-iv-cosmic-ffp/

Gómez, J. (2013). *Métodos de Medición en Puntos Función (y V): FiSMA.* Obtenido de https://www.laboratorioti.com/2013/02/04/metodos-de-medicion-en-puntos-funcion-y-v-fisma/

González López, P. (2018). *Tema 5: Diferentes aproximaciones a la ingeniería de requisitos.* Obtenido de https://curso1920.campusvirtual.uclm.es/pluginfile.php/3542078/mod\_resource/content/7/TEMA5-Scrum-LeanUX.pdf

Gutierrez Ramirez, L. (11 de 08 de 2011). *Complejidad Ciclomatica - Programacion.* Obtenido de https://issuu.com/zuzunime/docs/paper\_componentes

Highsmith, J. (13 de 02 de 2001). *Manifesto Ágil.* Obtenido de https://agilemanifesto.org/iso/es/manifesto.html

International Function Point Users Group. (1 de 10 de 2009). *a Internacional de Usuarios función de punto Group anuncia el lanzamiento de la versión 4.3 del Manual de Prácticas de conteo.* Obtenido de https://www.ifpug.org/the-international-function-point-users-group-announces-the-release-of-version-4-3-of-the-counting-practices-manual/?lang=es

International Function Point Users Group. (2021). *Acerca de Snap.* Obtenido de https://www.ifpug.org/about-snap/?lang=es

International Function Point Users Group. (2021). *Software Certification.* Obtenido de https://www.ifpug.org/certification/software-certification/

ISO. (2002). *ISO/IEC 20968:2002(en). Software engineering — Mk II Function Point Analysis — Counting Practices Manual.* Obtenido de https://www.iso.org/obp/ui/fr/#iso:std:iso-iec:20968:ed-1:v1:en:sec:1

Lorenzo, J. (2021). *International Function Point Users Group.* Obtenido de https://www.ifpug.org/introduction/?lang=es

Lundholm, P., & Brodd, M. (23 de 04 de 2014). *Time vs Story Points Estimation.* Obtenido de https://blog.crisp.se/2014/04/23/perlundholm/time-vs-story-points-estimation

Lynch, W. (09 de 01 de 2019). *The Brief of History of Scrum.* Obtenido de https://warren2lynch.medium.com/the-brief-of-history-of-scrum-15efb73b4701

*Medición del software.* (04 de 10 de 2011). Obtenido de http://www.lsi.us.es/docencia/get.php?id=6916

Microsoft. (2021). *Microsoft | Visual Studio.* Obtenido de https://visualstudio.microsoft.com/es/thank-you-downloading-visual-studio/?sku=Enterprise&rel=16#getting-started

NAVARRO MARTÍNEZ, E. (2019). *Unit 2. Software Testing: Techniques & Strategies.* Obtenido de https://curso1920.campusvirtual.uclm.es/pluginfile.php/4671985/mod\_resource/content/3/Unit%202-students.pdf

Nesma. (27 de 08 de 2014). *¿Qué es el análisis de puntos función?* Obtenido de https://nesma.org/faq/function-point-analysis-fpa-function-points/?lang=es

P.K.Suri, & Ranjan, P. (21 de 06 de 2012). *Comparative Analysis of Software Effort Estimation Techniques.* Obtenido de http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.258.7871&rep=rep1&type=pdf

Patton, J. (25 de 09 de 2014). *User Story Mapping: Discover the Whole Story, Build the Right Product.* O'Reilly. Obtenido de https://www.jpattonassociates.com/user-story-mapping/

Pierce, R. (2020). *Fibonnacci Sequence.* Obtenido de https://www.mathsisfun.com/numbers/fibonacci-sequence.html

rawat, u., & Rai, A. (14 de 11 de 2019). *Software Engineering | Project size estimation techniques.* Obtenido de https://www.geeksforgeeks.org/software-engineering-project-size-estimation-techniques/

Romero, E. (30 de 04 de 2021). *GitLab – Un servicio web de control y desarrollo de software colaborativo.* Obtenido de https://estebanromero.com/herramientas-emprender-desarrollar-proyectos/gitlab-un-servicio-web-de-control-y-desarrollo-de-software-colaborativo/

Sáez Vacas, F. (11 de 12 de 2008). *Medidas de la complejidad del software.* Obtenido de http://dit.upm.es/~fsaez/intl/libro\_complejidad/14-medidas-de-la-complejidad-del-software.pdf

Sandeep, J. (05 de 08 de 2020). *Software Engineering | Halstead’s Software Metrics.* Obtenido de https://www.geeksforgeeks.org/software-engineering-halsteads-software-metrics/

Schwaber, K., & Sutherland, J. (18 de 11 de 2020). Obtenido de https://scrumguides.org/scrum-guide.html

Slack Technologies. (2021). *¿Qué es Slack?* Obtenido de https://slack.com/intl/es-es/help/articles/115004071768-%C2%BFQu%C3%A9-es-Slack-

The International Function Point Users Group. (1 de 2005). *Function Point Counting Practices Manual.* Obtenido de https://epmc2.monsite-orange.fr/file/d6ab0a1755c60de1840c1337f50b64d2.pdf

Visual Paradigm. (2021). *How to Write Product Vision for Scrum Project?* Obtenido de https://www.visual-paradigm.com/scrum/how-to-write-scrum-product-vision/

Weiss, E. (03 de 2002). *Measuring LOC and other basic measurement.* Obtenido de https://files.ifi.uzh.ch/rerg/amadeus/teaching/seminars/seminar\_ws0203/Seminar\_3.pdf

1. Historias de Usuario.

En este anexo encontramos una tabla con las historias de usuario creadas durante el desarrollo del proyecto, con sus criterios de aceptación y los puntos de historia.

**Tabla 7.1.** Historias de Usuario

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Historia de Usuario | Criterios de Aceptación | Puntos Historia |
| FPA-1 | Como Tutor quiero que se inicialice el Product Backlog antes de comenzar el desarrollo Scrum, para así tener un conjunto priorizado de primeros PBIs a elegir para el Sprint 1. | La herramienta Jira debe de estar instalada para poder inicializar el Product Backlog. | 1 |
| FPA-2 | Como medidor principal quiero crear un proyecto nuevo para poder empezar a aplicar FPA (Functional Point Analysis). | Se activan los botones de creación de elementos y la cuenta se inicializa a 0. | 3 |
| FPA-3 | Como medidor principal quiero abrir un proyecto ya creado para continuar con la cuenta. | El proyecto debe haber sido guardado previamente. | 8 |
| FPA-4 | Como medidor principal quiero deshacer y rehacer cambios para corregir errores | Los elementos y cuenta vuelven a su estado anterior o posterior. | 5 |
| FPA-5 | Como medidor principal quiero insertar una descripción textual de un FUR para poder insertar las funciones de datos. | El texto debe aparecer debajo del canvas de ese FUR | 8 |
| FPA-6 | Como medidor principal quiero crear funciones de datos (EIF, ILF) para indicar la funcionalidad de almacenamiento. | Tiene que asignarse a una aplicación y se etiqueta automáticamente si es un ILF o EIF. También se le da un nombre y se añade a la lista de nodos de funciones de datos | 8 |
| FPA-7 | Como medidor principal quiero editar funciones de datos (EIF, ILF) para modificar la funcionalidad de almacenamiento. | Si se asigna a otra aplicación o se modifica se debe etiquetar correctamente y se tiene que actualizar la lista de nodos de funciones de datos. | 5 |
| FPA-8 | Como medidor principal quiero borrar funciones de datos (EIF, ILF) para no tener funciones que no se utilizan o que son erróneas. | Tiene que anularse la asignación a una aplicación, y también tiene que quitarse de la lista de nodos de funciones de datos. | 3 |
| FPA-9 | Como medidor principal quiero crear aplicaciones para alojar las funciones de datos. | Debe de permitirme indicar si es aplicación principal o secundaria y debe de permitirme alojar funciones de datos. | 13 |
| FPA-10 | Como medidor principal quiero editar aplicaciones para modificar las características de la aplicación. | Que cuando realice una modificación esta tenga efecto en la aplicación | 5 |
| FPA-11 | Como medidor principal quiero borrar aplicaciones para poder así quitar las que no son correctas o que no sean necesarias | Cuando se borre una aplicación tienen que recalcularse las cuentas y que no aparezca la membrana. | 3 |
| FPA-12 | Como medidor principal quiero crear funciones transaccionales en un FUR para poder asignarlas a elementos | Las funciones transaccionales deben aparecer asignadas al FUR. | 8 |
| FPA-13 | Como medidor principal quiero modificar funciones transaccionales en un FUR para editarlas a nuestra conveniencia | Cuando se editen se debe de mostrar las nuevas actualizaciones. | 3 |
| FPA-14 | Como medidor principal quiero borrar funciones transaccionales en un FUR para así quitarla de donde no corresponde | Se debe de haber borrado esa función transaccional borrada. | 2 |
| FPA-15 | Como medidor principal quiero asignar funciones de datos a FUR y aplicaciones para que se sumen a la cuenta. | Debe haber aplicaciones donde poder asignar las funciones de datos | 5 |
| FPA-16 | Como medidor principal quiero conectar funciones y membranas o aplicaciones para así poder realizar los cálculos | las conexiones deben de ser posibles, hay algunos elementos que no se pueden asociar entre sí. Se deben actualizar las cuentas cuando se realicen las conexiones. | 13 |
| FPA-17 | Como medidor principal quiero exportar un informe para mostrar el tamaño funcional y los distintos elementos añadidos | Se genera un fichero con el tamaño funcional y una representación gráfica de la aplicación. | 8 |
| FPA-18 | Como medidor principal quiero guardar el proyecto para continuar su edición más adelante | Debe de haber elementos creados que puedan ser guardados | 13 |
| FPA-20 | Como medidor principal quiero ver las tablas de complejidad | Se debe de buscar las tablas de complejidad | 5 |
| FPA-21 | Como colaborador quiero comentar FURs, DETs, RETs, FTRs para opinar en relación con las funciones creadas. | El comentario que acabo de crear debe aparecer | 8 |
| FPA-22 | Como automatización quiero calcular la complejidad de los FUR y la aplicación para obtener el tamaño funcional del programa. | Se muestra el cálculo de complejidad de los FUR y de la aplicación. | 13 |
| FPA-23 | Como Tribunal quiero un cronograma para que el alumno demuestre que tenía un plan a nivel de hitos desde el principio. | La herramienta Jira debe de estar instalada para poder crear el cronograma. | 1 |
| FPA-24 | Como tutor queremos que recopiles la información necesaria para el Estado del Arte para así poder seguir con el desarrollo de la memoria. | Hay que tener alguna fuente de información como internet, libros, etc. para poder buscar información. | 5 |
| FPA-25 | Como tutor queremos que instancies la plantilla de TFG. | Se debe haber descargado la plantilla de la página oficial de la universidad. | 1 |
| FPA-26 | Como tutor queremos que se inicialice el repositorio que almacena el código fuente. |  | 2 |
| FPA-43 | Como tutor queremos que crees un Hola Mundo arrastrable en UWP para poder así empezar con el desarrollo del código |  | 5 |
| FPA-46 | Como tutor queremos que escribas la información necesaria para el Estado del Arte para así poder seguir con el desarrollo de la memoria. |  | 5 |
| FPA-48 | Como tutor quiero que crees un prototipo detallado de cómo se vera la interfaz para tener una idea más clara de la aplicación. | Se necesita una herramienta de prototipado para poder crearlo. | 5 |
| FPA-49 | Como tutor quiero que añadas funcionalidad al proyecto para ver como interactúa la interfaz. | El repositorio del proyecto debe haber sido creado previamente para poder incrementar la funcionalidad de este. | 8 |
| FPA-50 | Como tutor quiero que mejores el sistema de ventanas y los campos editables de la interfaz para obtener una mejor navegación a través de la aplicación |  | 5 |
| FPA-51 | Como tutor quiero que añadas funcionalidad arrastrable a los FUR y mejora de la interfaz para obtener una mejor navegación a través de la aplicación. |  | 8 |
| FPA-52 | Como tutor quiero que guardes un proyecto con su nombre para posteriormente volverlo a cargar. |  | 8 |
| FPA-53 | Como tutor quiero que guardes un FUR y aplicaciones para posteriormente poder volverlas a cargar. |  | 23 |
| FPA-54 | Como tutor quiero que crees un conjunto de elementos como grupo para así poder mover los distintos elementos dentro del grupo |  | 8 |
| FPA-55 | Como tutor quiero que las aplicaciones queden bien ordenadas en el canvas principal o bien haciendo que las cuadriculas de cuadro sean las membranas o haciendo uso del GridView adaptable. |  | 8 |
| FPA-56 | Como medidor principal quiero que se sume los tamaños de los elementos creados en un FUR |  | 5 |
| FPA-57 | Como medidor principal quiero que la aplicación este desplegada en tienda para poder usarla. |  | 8 |

1. Despliegue en tienda Microsoft

En este anexo vemos los pasos que hay que realizar para desplegar una aplicación en la tienda de Microsoft. Estos son los siguientes:

### Registrarse como desarrollador en Microsoft, que conlleva un pago de 17 euros.

Se debe de acceder a la página de developer de Microsoft y crearse una cuenta.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

**Figura 7.1.** Registrarse como desarrollador

### Seleccionar .NET native tool chain en los ajustes del proyecto en Visual Studio

Se debe de seleccionar la casilla antes de la exportación del paquete.

Una captura de pantalla de una computadora

Descripción generada automáticamente

**Figura 7.2.** Seleccionar .NET native tool chain

### Reservar un nombre de la aplicación.

Se debe verificar que el nombre no existe y reservarlo.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

**Figura 7.3.** Reservar nombre

### Se genera un paquete desde la solución del proyecto.

Nos aseguramos de que todo funciona correctamente y creamos el paquete.

Una captura de pantalla de una computadora

Descripción generada automáticamente

**Figura 7.4.** Generar paquete .msix

### Comprobamos con el kit de certificación.

Se comprueba que el paquete es correcto con la herramienta de certificación que nos ofrece Windows.

Una captura de pantalla de una computadora

Descripción generada automáticamente

**Figura 7.5**. Revisión del paquete con kit Certification

### Realizamos la sumisión

Se realiza una sumisión donde hay que rellenar varios apartados; precio y disponibilidad, propiedades de la aplicación, clasificación de edad, añadir el paquete, indicar el idioma y opción de publicación.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

**Figura 7.6.** Sumisión en Microsoft Partner Center

### Esperar a que los desarrolladores de Windows certifiquen la aplicación

Los desarrolladores comprueban que la aplicación funciona correctamente antes de certificarla, ya que tiene que pasar un control de calidad.

Captura de pantalla de computadora

Descripción generada automáticamente

**Figura 7.7**. Desarrolladores de Microsoft deben verificar la aplicación

### Error surgido durante el despliegue

Al revisar el paquete con la herramienta kit de certificación salía un error en las librerías de msi, por lo que se tuvo que desplegar la aplicación desde otro PC (Asus y Windows 10). EL equipo de desarrollo piensa que es algún problema de residuos de Windows 10.

Interfaz de usuario gráfica, Aplicación, Word

Descripción generada automáticamente

**Figura 7.8**. Error librería .msi