



UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática



**TFG del Grado en Ingeniería
Informática**

CRET

**CanBus Reverse Engineering
Toolkit
Documentación Técnica**



Presentado por Adrián Marcos Batlle
en Universidad de Burgos — 6 de febrero
de 2019

Tutor: Álgvar Arnaiz-González y César
Represa Pérez

Índice general

Índice general	I
Índice de figuras	III
Índice de tablas	IV
Apéndice A Plan de Proyecto Software	1
A.1. Introducción	1
A.2. Planificación temporal	2
A.3. Estudio de viabilidad	4
Apéndice B Especificación de Requisitos	7
B.1. Introducción	7
B.2. Objetivos generales	7
B.3. Catalogo de requisitos	7
B.4. Especificación de requisitos	9
Apéndice C Especificación de diseño	21
C.1. Introducción	21
C.2. Diseño de datos	21
C.3. Diseño arquitectónico	22
C.4. Diseño de interfaces	25
Apéndice D Documentación técnica de programación	27
D.1. Introducción	27
D.2. Estructura de directorios	27
D.3. Manual del programador	28

D.4. Compilación, instalación y ejecución del proyecto	32
Apéndice E Documentación de usuario	33
E.1. Introducción	33
E.2. Requisitos de usuarios	33
E.3. Instalación	33
E.4. Manual del usuario	33

Índice de figuras

A.1. Resumen de horas dedicadas al proyecto.	4
C.1. Diagrama de la base de datos	21
C.2. Modelo MVC	22
C.3. Paquetes de la aplicación	23
C.4. Diagrama de clases y paquetes de la aplicación.	24
C.5. Diagrama de clases y paquetes de la interfaz gráfica.	25
C.6. Diagrama de navegabilidad	26
D.1. Comando Git sobre WSL.	29
D.2. SceneBuilder 10.0	29
D.3. IDE Eclipse	30
D.4. Comando Git sobre WSL.	31

Índice de tablas

A.1. Horas dedicadas al proyecto.	4
A.2. Costes de personal.	5
A.3. Costes de personal.	5
A.4. Costes totales.	5
A.5. Dependencias del proyecto y sus licencias	6
B.1. C-1 Gestión de proyectos.	10
B.2. C-2 Listar proyectos.	11
B.3. C-3 Crear proyecto.	11
B.4. C-4 Eliminar proyecto.	12
B.5. C-5 Abrir proyecto.	13
B.6. C-6 Cerrar proyecto.	14
B.7. C-7 Importar proyecto.	15
B.8. C-8 Exportar proyecto.	16
B.9. C-9 Obtención de los puertos serie disponibles.	17
B.10.C-10 Obtención de los puertos serie disponibles.. . . .	18
B.11.C-11 Etiquetado de datos.	18
B.12.C-12 Monitorización de los datos.	19

Apéndice A

Plan de Proyecto Software

A.1. Introducción

Una de las fases más importantes de cualquier proyecto es la planificación. En esta fase se estima el tiempo y los requisitos necesarios para la realización del proyecto. Para ello es necesaria una idea clara de lo que se quiere realizar de manera que todas las partes que componen el proyecto puedan ser analizadas individualmente.

Podemos dividir la fase de planificación en dos fases más pequeñas:

- Planificación temporal.
- Estudio de viabilidad.

En la primera fase se realiza una planificación de los tiempos esperados. Se especifican los tiempos necesarios para cada una de las partes que componen el proyecto, a la vez que una fecha de inicio y de fin de las mismas.

La segunda fase se centra en un estudio de la viabilidad del proyecto. En esta fase se pueden diferenciar dos partes:

- **Viabilidad económica:** Estimación de los costes y beneficios del proyecto.
Viabilidad legal: Regulaciones legales que pueden afectar al proyecto. En este caso serían la Ley de Protección de Datos y las licencias del software.

A.2. Planificación temporal

Para realizar una planificación del proyecto adecuada, se decidió aplicar *Scrum* (una metodología ágil).

Para ello, se dividió el trabajo en *sprints* los cuales iban siendo planificados según se terminaba el anterior. La duración media de estos era de una semana de trabajo.

En cada uno de los *sprint* se generaban una serie de tareas las cuales tenían que ser realizadas en ese intervalo de tiempo. Para ello se utilizó la plataforma *Zenhub*, en la cual las *issues* hacen de tareas, y los *Milestones* de *Sprints*.

Sprint 0

Este primer *sprint* fue en el que más complicado de planificar. Estaba incluido el desarrollo del *hardware* el cual no sabíamos seguro si se conseguirían los conocimientos necesarios para diseñarlo, ni si después funcionaría correctamente. Además, el desarrollo de la aplicación dependía del resultado de esta parte ya que iba a basarse en ello.

Al final, sobrepasando un poco el tiempo planificado, se consiguió poner el *hardware* en marcha, con lo que se podía comenzar con el desarrollo del *software*.

Sprint 1

Los objetivos de este *sprint* fueron asentar la idea general de la aplicación a desarrollar, su estructura principal y funcionalidades.

Además se comenzó a diseñar la estructura de la base de datos y el acceso al *hardware* a través de la comunicación con los puertos serie.

Las horas de trabajo estimadas fueron cumplidas como estaba planificado en el *Sprint*.

Sprint 2

Los objetivos de este *sprint* fueron comenzar con el desarrollo de la interfaz gráfica. Para ello era necesario la creación de gráficas de forma dinámica. Estas serían utilizadas para monitorizar los datos que fluían por el bus CAN en tiempo real.

Esta fase llevó un poco más tiempo de lo esperado.

Sprint 3

En este *sprint* se planteó la posibilidad de poder importar y exportar un proyecto de la aplicación. Para ello, se decidió usar ficheros JSON, los cuales contendrían todos los datos del proyecto.

En este punto se planteó la posibilidad de realizar una parte de la aplicación sobre una tecnología web. La idea era generar los ficheros JSON desde la aplicación Java para posteriormente poder importarlos a la web para su monitorización desde la misma. Por falta de tiempo, no se realizó esta segunda aplicación.

Sprint 4

El objetivo de este *sprint* era la generación de un gestor de proyectos interno en la aplicación. Para ello era necesario diseñar las funcionalidades para la creación de un nuevo proyecto, la asignación de valores identificados al mismo, así como sus opciones de guardar y eliminar de la base de datos.

Sprint 5

En este *sprint* se desarrolló la interacción con la base de datos, tanto para obtener datos de ella como para recuperarlos. Además se realizó un tratado de las posibles excepciones que puedan surgir por problemas internos de la aplicación.

Sprint 6

a
a
a a aaaa a a a
a aa aa a
a

Sprint 7

Resumen

En la siguiente tabla se muestran los tiempos dedicados a cada uno de los distintos tipos de tareas:

Categoría	Tiempo (horas)
<i>Documentation</i>	50
<i>Features</i>	150
<i>Research</i>	20
<i>Bug Fix</i>	20
TOTAL	240

Tabla A.1: Horas dedicadas al proyecto.

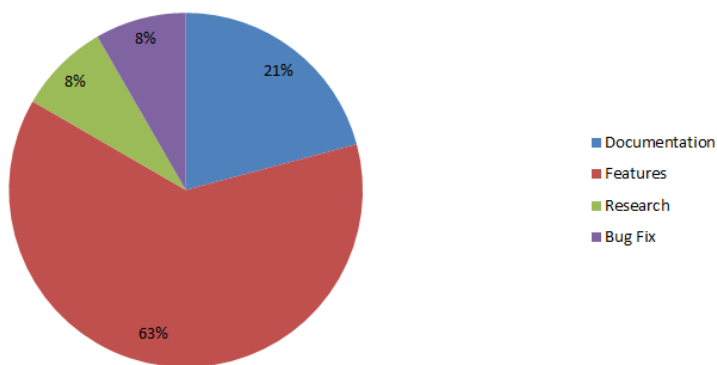


Figura A.1: Resumen de horas dedicadas al proyecto.

A.3. Estudio de viabilidad

En este punto se van a analizar por separado la viabilidad económica y la viabilidad legal del proyecto.

Viabilidad económica

Costes

Los costes van a ser desglosados en dos partes:

Costes de personal:

El proyecto ha sido desarrollado por una sola persona, empleada a tiempo completo durante tres meses. Considerando los siguientes valores:

Concepto	Coste
Salario mensual neto	1008,15 €
IRPF (9,64 %)	115,68 €
SS (30 %)	360 €
Salario mensual bruto	1200 €
Total tres meses	3600 €

Tabla A.2: Costes de personal.

Costes de *hardware*:

En este apartado se revisan los costes del *hardware* desarrollado en el proyecto.

Concepto	Coste
PCB (10 prototipos)	1,75 €
Transporte	7,11 €
Componentes (para 2 uds)	51,90 €
Aduanas	15 €
Soldador	15 €
Estaño	5 €
Flux	3 €
Total	98,76 €

Tabla A.3: Costes de personal.

Costes totales:

Concepto	Coste
Personal	3600 €
<i>Hardware</i>	98,76 €
Total	3698,76 €

Tabla A.4: Costes totales.

Viabilidad legal

A continuación se exponen todos los temas relacionados con las licencias del software y *hardware* de proyecto, así como de la documentación e

imágenes.

Tabla de licencias de las dependencias utilizadas:

Dependencia	Versión	Descripción	Licencia
Java	8.0.191	JDK	——
USBtinLib	1.2.0	Librería para la conexión con el <i>hardware</i> .	——
JSSC	2.8.0	Librería para la búsqueda de puertos serie.	——
sqlite-jdbc	3.23.1	Librería para realizar la conexión con la base de datos.	——
json	1.6	Librería para la importación y exportación de ficheros JSON.	——

Tabla A.5: Dependencias del proyecto y sus licencias

Apéndice B

Especificación de Requisitos

B.1. Introducción

A continuación se realiza la especificación de requisitos que define el comportamiento del sistema desarrollado en el proyecto.

B.2. Objetivos generales

Los objetivos generales del proyecto son los siguientes:

- Desarrollar una aplicación para el análisis y monitorización de los datos que fluyen por el bus CAN.
- Desarrollo de un *hardware* libre el cual permita conectarse y monitorizar dos buses de datos de forma simultanea.

B.3. Catalogo de requisitos

En este apartado se describen los requisitos específicos del proyecto.

Requisitos funcionales

- **RF-1: Gestor de proyectos:** La aplicación debe de ser capaz de gestionar proyectos.

- **RF-1.1: Crear proyecto:** El usuario debe de poder crear un nuevo proyecto con un nombre específico.
 - **RF-1.1.1: Añadir datos al proyecto:** El usuario debe de poder añadir los datos identificados al proyecto.
- **RF-1.2: Listar proyectos:** El usuario debe de poder listar los proyectos existentes en la aplicación.
- **RF-1.3: Eliminar proyectos:** El usuario debe de poder eliminar los proyectos existentes en la aplicación.
- **RF-1.3.1:** Confirmar la eliminación: El usuario debe de poder confirmar si desea eliminar el proyecto.
- **RF-1.4: Abrir proyecto:** El usuario debe de poder abrir un proyecto existente en la aplicación.
- **RF-1.5: Cerrar proyecto:** El usuario debe de poder cerrar un proyecto que esté abierto.
- **RF-1.6: Importar proyecto:** El usuario debe ser capaz de importar un proyecto a la aplicación.
- **RF-1.7: Exportar proyecto:** El usuario debe ser capaz de exportar un proyecto de la aplicación.
- **RF-2: Configuración de las interfaces:** El usuario debe de poder configurar las interfaces CAN correspondientes.
- **RF-2.1: Obtención de los puertos serie:** El usuario debe de poder obtener un listado de los puertos serie disponibles en el equipo que ejecuta la aplicación.
- **RF-2.2: Configuración de velocidad:** El usuario debe de ser capaz de configurar la velocidad con la que una interfaz va a ser conectada.
- **RF-2.3: Configuración del modo:** El usuario debe de ser capaz de configurar el modo con el que quiere conectarse al bus CAN.
- **RF-3: Configuración del análisis:** El usuario debe de ser capaz de configurar el tipo de análisis que quiere realizar.
- **RF-3.1: Ignorar *bytes* con valor cero:** El usuario debe ser capaz de seleccionar si desea ignorar los *bytes* que no envíen ningún valor por el bus.
- **RF-3.2: Separar los *bytes* en dos partes:** El usuario debe de ser capaz de separar los *bytes* en dos partes para mejorar el análisis si fuera necesario.

- **RF-4: Etiquetado de datos:** El usuario debe de ser capaz de etiquetar los datos una vez identificados.
- **RF-4.1: Monitorización de los datos:** El usuario debe de ser capaz de monitorizar los datos una vez han sido etiquetados.

Requisitos no funcionales

- **RNF-1: Monitorización:** La aplicación debe monitorizar correctamente los datos que están siendo analizados siempre que se esté conectado al bus.
- **RNF-2: Rendimiento:** La aplicación debe de funcionar fluidamente, sin que la interfaz gráfica se quede bloqueada.
- **RNF-3: Escalabilidad:** La aplicación debe de permitir la incorporación de módulos de forma sencilla.
- **RNF-4: Usabilidad:** La aplicación debe ser fácil de utilizar, intuitiva y con una estructura clara.

B.4. Especificación de requisitos

En esta sección se exponen los casos de uso de la aplicación.

Actores

El único actor del sistema será la persona que maneja la aplicación y que identificará las señales importantes.

Casos de uso

C-1	Gestión de proyectos
Requisitos asociados	RF-1, RF-1.1, RF-1.1.1, RF-1.2, RF-1.3, RF-1.4, RF-1.5, RF-1.6, RF-1.7
Descripción	Permite gestionar los proyectos de la aplicación.
Precondición	Existe una conexión a la base de datos.

C-1	Gestión de proyectos
------------	-----------------------------

Acciones

1. El usuario ejecuta la aplicación.
2. Se listan los proyectos disponibles.
3. Por cada proyecto, existe la opción de abrir y eliminar.
4. Es posible importar un nuevo proyecto.
5. Es posible exportar un proyecto.

Excepciones

- Error en la conexión con la base de datos.

Importancia Alta

Tabla B.1: C-1 Gestión de proyectos.

C-1	Listar proyectos
------------	-------------------------

Requisitos asociados RF-1.2

Descripción Permite listar los proyectos de la aplicación.

Precondición Existe una conexión a la base de datos.

Hay uno o más proyectos.

Acciones

1. El usuario ejecuta la aplicación.
2. El usuario lista los proyectos disponibles.

C-1	Listar proyectos
Excepciones	<ul style="list-style-type: none"> ■ No existe ningún proyecto. ■ Error en la conexión con la base de datos.
Importancia	Alta
Tabla B.2: C-2 Listar proyectos.	
C-3	Crear proyecto
Requisitos asociados	RF-1.1, RF-1.1.1
Descripción	Permite crear un nuevo proyecto y añadirle datos.
Precondición	Existe una conexión a la base de datos.
Acciones	<ol style="list-style-type: none"> 1. El usuario ejecuta la aplicación. 2. Se crea un nuevo proyecto. 3. Se guardan los datos del proyecto.
Excepciones	<ul style="list-style-type: none"> ■ Error en la conexión con la base de datos. ■ Proyecto con el mismo nombre existente.
Importancia	Alta
Tabla B.3: C-3 Crear proyecto.	

C-4 Eliminar proyecto	
Requisitos asociados	RF-1.3, RF-1.3.1
Descripción	Permite eliminar un proyecto existente.
Precondición	Existe una conexión a la base de datos
Acciones	Existe el proyecto a eliminar.
	<ol style="list-style-type: none"> 1. El usuario ejecuta la aplicación. 2. El usuario lista los proyectos. 3. El usuario selecciona el proyecto a eliminar. 4. El usuario confirma la eliminación del proyecto. 5. El proyecto se borra de la base de datos.
Excepciones	<ul style="list-style-type: none"> ■ Error en la conexión con la base de datos. ■ El proyecto no existe.
Importancia	Alta
Tabla B.4: C-4 Eliminar proyecto.	
C-5 Abrir proyecto	
Requisitos asociados	RF-1.4
Descripción	Permite abrir proyecto existente.
Precondición	Existe una conexión a la base de datos.
	Existe un proyecto.

C-5	Abrir proyecto
Acciones	
	<ol style="list-style-type: none"> 1. El usuario ejecuta la aplicación. 2. Lista los proyectos disponibles. 3. El usuario selecciona el proyecto que desea abrir. 4. El usuario abre el proyecto.
Excepciones	
	<ul style="list-style-type: none"> ■ Error en la conexión con la base de datos. ■ El proyecto no existe.
Importancia	Alta
Tabla B.5: C-5 Abrir proyecto.	
C-6	Cerrar proyecto
Requisitos asociados	RF-1.5
Descripción	Permite abrir proyecto existente.
Precondición	<p>Existe una conexión a la base de datos.</p> <p>Hay un proyecto abierto.</p>

C-6	Cerrar proyecto
Acciones	<ol style="list-style-type: none"> 1. El usuario tiene un proyecto abierto en la aplicación. 2. El usuario cierra el proyecto. 3. El usuario confirma que desea guardar los cambios. 4. Se cierra el proyecto.
Excepciones	<ul style="list-style-type: none"> ■ El proyecto no está abierto. ■ Error en la conexión con la base de datos.
Importancia	Alta
Tabla B.6: C-6 Cerrar proyecto.	
C-7	Importar proyecto
Requisitos asociados	RF-1.6
Descripción	Permite importar un proyecto.
Precondición	<p>Existe una conexión a la base de datos.</p> <p>Existe un fichero para importar.</p>

C-7	Importar proyecto
Acciones	<ol style="list-style-type: none">1. El usuario ejecuta la aplicación.2. El usuario lista los proyectos disponibles.3. El usuario selecciona el fichero que desea importar.4. El usuario lista los proyectos disponibles de nuevo.
Excepciones	<ul style="list-style-type: none">■ Error en la conexión con la base de datos.■ Existe otro proyecto con el mismo nombre.
Importancia	Alta

Tabla B.7: C-7 Importar proyecto.

C-8	Exportar proyecto
Requisitos asociados	RF-1.7
Descripción	Permite exportar un proyecto.
Precondición	Existe una conexión a la base de datos. Hay 1 o más proyectos disponibles.

C-8	Exportar proyecto
Acciones	
<ol style="list-style-type: none"> 1. El usuario ejecuta la aplicación. 2. El usuario lista los proyectos disponibles. 3. El usuario selecciona el proyecto que desea exportar. 4. El usuario guarda el fichero con el proyecto. 	
Excepciones	
<ul style="list-style-type: none"> ■ Error en la conexión con la base de datos. ■ No existen proyectos en la aplicación. 	
Importancia	Alta

Tabla B.8: C-8 Exportar proyecto.

C-9	Obtención de los puertos serie disponibles.
Requisitos asociados	RF-2, RF-2.1
Descripción	Permite obtener los puertos serie disponibles en el equipo.
Precondición	Puertos serie disponibles en el equipo.
Acciones	
<ol style="list-style-type: none"> 1. El usuario ejecuta la aplicación 2. El usuario se dispone a configurar una de las interfaces. 3. El usuario lista los puertos serie disponibles. 	

C-9	Obtención de los puertos serie disponibles.
------------	--

Excepciones

- No existe ningún puerto serie disponible.

Importancia Alta

Tabla B.9: C-9 Obtención de los puertos serie disponibles.

C-10	Configuración de la interfaz CAN
-------------	---

Requisitos asociados RF-2.1, RF-2.2, RF-2.3, RF-3, RF-3.1, RF-3.2

Descripción Permite conectarse a una interfaz CAN.

Precondición Existe un puerto serie en la máquina.

Acciones

1. El usuario ejecuta la aplicación.
2. El usuario selecciona un puerto serie de la máquina.
3. El usuario configura la velocidad de conexión.
4. El usuario configura el modo de conexión.
5. El usuario configura si desea ignorar los *bytes* con valor cero.
6. El usuario configura si desea separar los *bytes* en dos partes.

Excepciones

- No existe un puerto serie en la máquina.

Importancia Alta

C-10 Configuración de la interfaz CAN	
Tabla B.10: C-10 Obtención de los puertos serie disponibles..	
C-11 Etiquetado de datos	
Requisitos asociados	RF-4
Descripción	Permite asignar etiquetas a los datos identificados por el usuario.
Precondición	Existe un puerto serie en la máquina.
Acciones	<ol style="list-style-type: none"> 1. El usuario ejecuta la aplicación. 2. El usuario configura una interfaz. 3. El usuario crea un nuevo proyecto. 4. El usuario comienza el análisis. 5. El usuario identifica uno de los datos, le establece una etiqueta y lo envía al <i>dashboard</i>.
Excepciones	<ul style="list-style-type: none"> ■ No existe un puerto serie. ■ No se puede iniciar la comunicación con el bus CAN. ■ No existe un proyecto.
Importancia	Alta
Tabla B.11: C-11 Etiquetado de datos.	

C-12	Monitorización de los datos
Requisitos asociados	RF-4.1
Descripción	Permite monitorizar los datos que circulan por el bus CAN.
Precondición	Existe un puerto serie en la máquina. Existe una conexión con el bus CAN.
Acciones	<ol style="list-style-type: none"> 1. El usuario ejecuta la aplicación. 2. El usuario configura una de las interfaces. 3. El usuario lista los proyectos disponibles. 4. El usuario abre uno de los proyectos. 5. El usuario comienza la monitorización de los datos.
Excepciones	<ul style="list-style-type: none"> ■ No existe conexión con la base de datos. ■ No existe un puerto serie en el equipo. ■ No se ha podido iniciar la comunicación con el bus CAN. ■ No existe un proyecto.
Importancia	Alta

Tabla B.12: C-12 Monitorización de los datos.

Apéndice C

Especificación de diseño

C.1. Introducción

En esta sección se expone cómo se han resuelto las especificaciones indicadas anteriormente.

Se define los datos que va a utilizar la aplicación, el diseño de sus interfaces y su arquitectura.

C.2. Diseño de datos

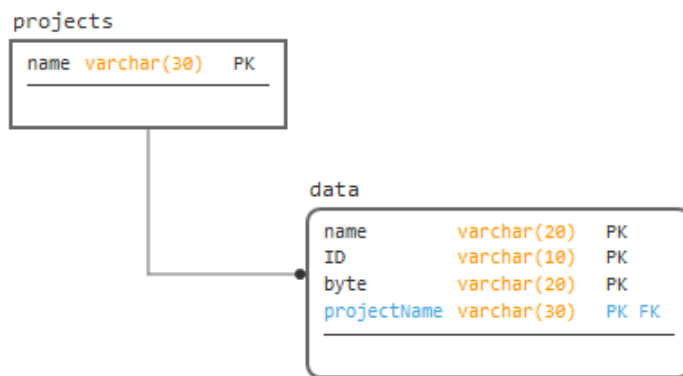


Figura C.1: Diagrama de la base de datos

C.3. Diseño arquitectónico

Model-View-Controller (MVC)

Aplicando este patrón de diseño, se utilizan 3 componentes, las vistas, los modelos y los controladores, de tal forma que se separa la lógica de la vista de la aplicación. Con el uso de este método, si realizamos una modificación en una parte de nuestro código, la otra parte no se ve afectada.

Por ejemplo, si modificamos la base de datos, solo modificaríamos el modelo encargado de esa acción, sin tener que modificar por ejemplo, la parte de la vista.

Está compuesto por tres componentes:

Model: Normalmente esta parte se encarga de los datos (no siempre). Esto puede ser por ejemplo, consultando a una base de datos para obtener información.

View: Componen la representación visual de los datos, es decir, todo lo que tenga que ver con la interfaz gráfica.

Controller: Es un mediador entre los modelos y las vistas. Se encarga de recibir las órdenes del usuario a través de la vista, realizar la petición de datos al modelo y devolverlo de nuevo a la vista para que sea mostrado al usuario.

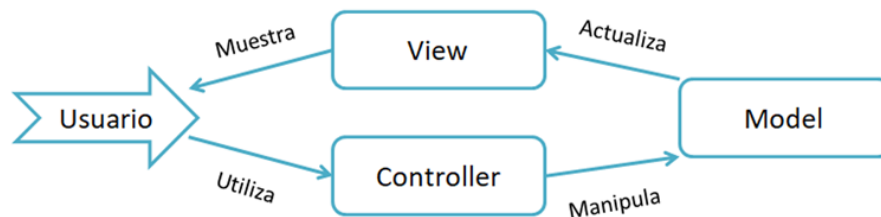


Figura C.2: Modelo MVC

Arquitectura general

ESQUEMA DE LA ARQUITECTURA

Diseño de paquetes

Para conseguir una organización interna del proyecto, se agruparon las distintas funcionalidades de la aplicación en paquetes. De esta manera se consigue una aplicación modular y con funcionalidades independientes.

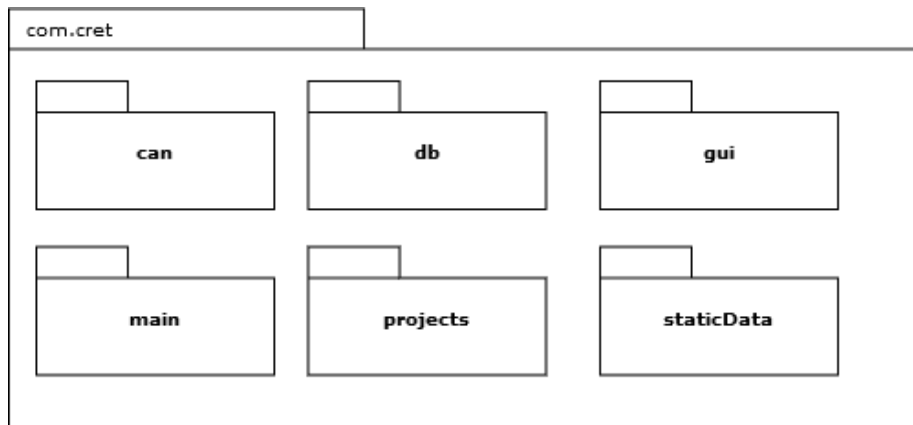


Figura C.3: Paquetes de la aplicación

A continuación se describen cada uno de los paquetes:

- **can:** En este paquete se almacena todo lo relacionado con la interacción con las interfaces CAN.
- **db:** En este paquete se almacena todo lo relacionado con la conexión a la base de datos, la obtención y la inserción de datos en la misma.
- **gui:** En este paquete se almacena todo lo relacionado con la interfaz gráfica.
- **main:** En este paquete se encuentra el *main* de la aplicación, así como el *preloader*.
- **projects:** En este paquete se almacena todo lo relacionado con la gestión de proyectos de la aplicación.
- **staticData:** En este paquete se almacenan una serie de estructuras las cuales son utilizadas por distintos componentes del proyecto. En el se almacenan todos los datos que son representados por las gráficas de la aplicación.

Diseño de clases

En este apartado se describe cada una de las clases que forman la aplicación y su funcionamiento.

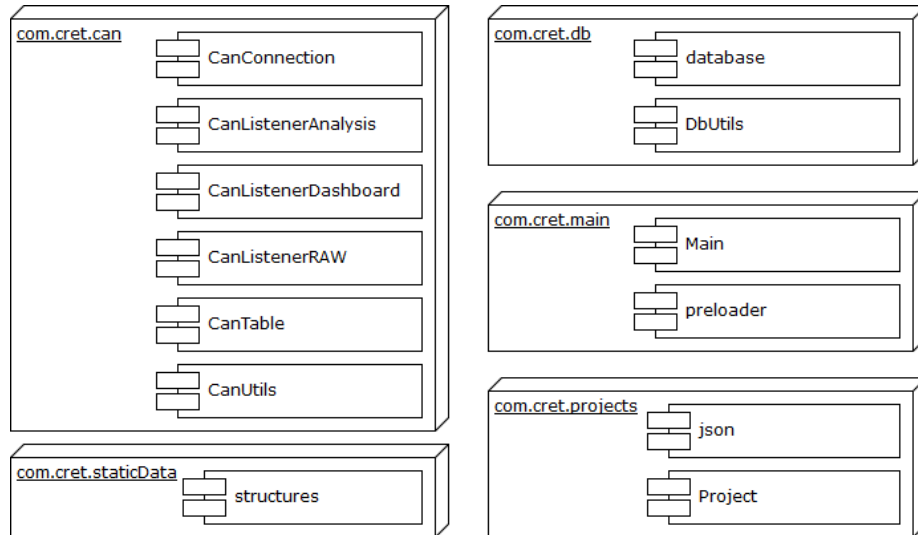


Figura C.4: Diagrama de clases y paquetes de la aplicación.

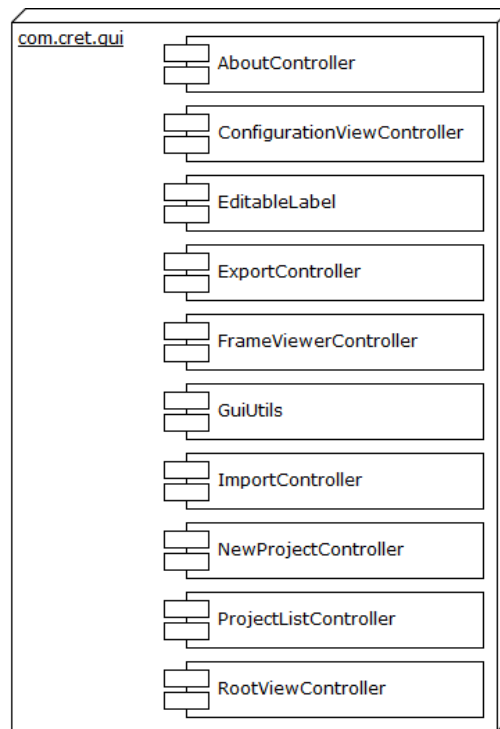


Figura C.5: Diagrama de clases y paquetes de la interfaz gráfica.

- **CanConnection:** -
- **CanListenerAnalysis:** -
- **CanListenerDashboard:** -
- **CanListenerRAW:** -
- **CanTable:** -
- **CanUtils:** -
- **database:** -
- **DbUtils:** -
- **Main:** -
- **Preloader:** -
- **json:** -

- **Project:** -
- **structures:** -
- **AboutController:** -
- **ConfigurationViewController:** -
- **EditableLabel:** -
- **ExportController:** -
- **FrameViewerController:** -
- **GuiUtils:** -
- **ImportController:** -
- **NewProjectController:** -
- **ProjectListController:** -
- **RootViewController:** -

C.4. Diseño de interfaces

El diseño de las interfaces se ha llevado a cabo a través de la utilidad SceneBuilder. Se ha intentado seguir una estructura limpia y clara, para facilitar su interacción con el usuario.

A continuación se muestran algunas de las vistas de la aplicación:

CAPTURAS DE LA APP

El siguiente diagrama nos muestra el flujo de navegabilidad por la aplicación.

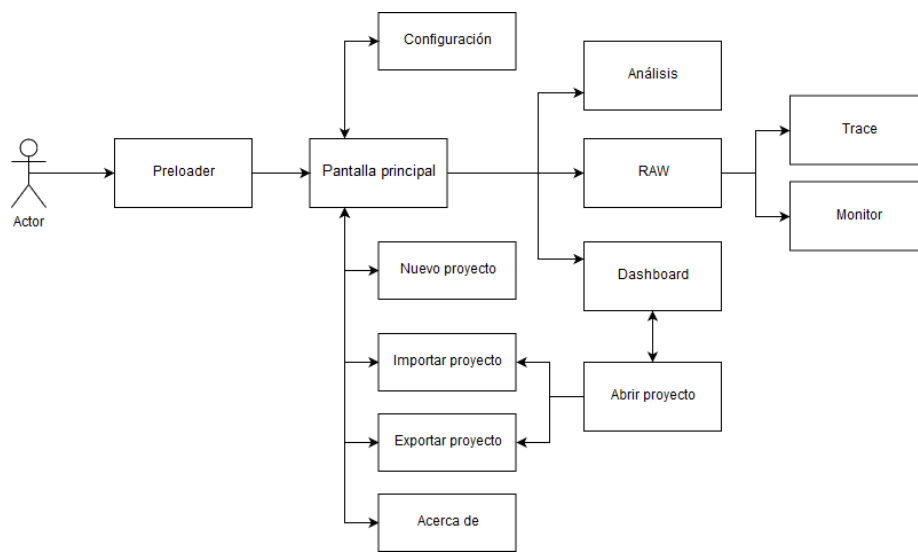


Figura C.6: Diagrama de navegabilidad

Apéndice *D*

Documentación técnica de programación

D.1. Introducción

En este anexo se describe la documentación técnica de programación, incluyendo la preparación del entorno de desarrollo, la estructura de la aplicación y las configuraciones necesarias.

D.2. Estructura de directorios

El proyecto tiene la siguiente estructura:

- **/documentacion/**: Documentación del proyecto.
- **/documentacion/img/**: Imágenes utilizadas en la documentación.
- **/documentacion/javadoc/**: Documentación *Javadoc*.
- **/hardware/**: Esquemas del *hardware* realizado en formato para Ki-CAD.
- **/hardware/gerber/**: Ficheros *gerber* para la producción del hardware.
- **/src/**: Código fuente de la aplicación Java
- **/resources/**: Recursos de la aplicación.

D.3. Manual del programador

El siguiente manual tiene como objetivo servir de referencia para futuros programadores. Se explica como montar el entorno de desarrollo, las dependencias necesarias así como su compilación y ejecución.

Entorno de desarrollo

Para el proyecto se necesita tener instaladas las siguientes dependencias:

- Java JDK 8
- JavaFX
- Git
- SceneBuilder
- Eclipse

Java JDK 8

Es uno de los lenguajes más utilizados hoy en día. El uso del lenguaje Java nos permite la compatibilidad de la aplicación en distintas plataformas. Se puede obtener del siguiente enlace:

ENLACE A JAVA JDK8

JavaFX

Se trata de un conjunto de productos y tecnologías para la creación de interfaces gráficas en Java. Para la ejecución de estas aplicaciones solamente es necesaria la instalación de JRE en la máquina.

Se puede obtener del siguiente enlace: ENLACE A JAVA FX

Git

Para utilizar el repositorio en el que se encuentra la aplicación, es necesario el uso de Git. Este programa nos permitirá clonar el repositorio a nuestra máquina y empezar a trabajar con ello.

Podemos instalarlo en Windows habilitando el WSL (Windows Subsystem for Linux), y ejecutando el comando *apt-get install git* con permisos de *root*.

```
ms02@DESKTOP-8ITLFGI:~/repo$ git clone https://github.com/amb0070/CRET.git
Cloning into 'CRET'...
Username for 'https://github.com': amb0070@alu.ubu.es
Password for 'https://amb0070@alu.ubu.es@github.com':
remote: Enumerating objects: 128, done.
remote: Counting objects: 100% (128/128), done.
remote: Compressing objects: 100% (78/78), done.
remote: Total 705 (delta 51), reused 84 (delta 25), pack-reused 577
Receiving objects: 100% (705/705), 10.25 MiB | 7.75 MiB/s, done.
Resolving deltas: 100% (325/325), done.
ms02@DESKTOP-8ITLFGI:~/repo$
```

Figura D.1: Comando Git sobre WSL.

SceneBuilder

Se trata de una interfaz gráfica creada para el desarrollo de interfaces gráficas en JavaFX. Sigue el modelo drag and drop, y toda la información es almacenada en un fichero FXML, un formato especial que extiende de XML.

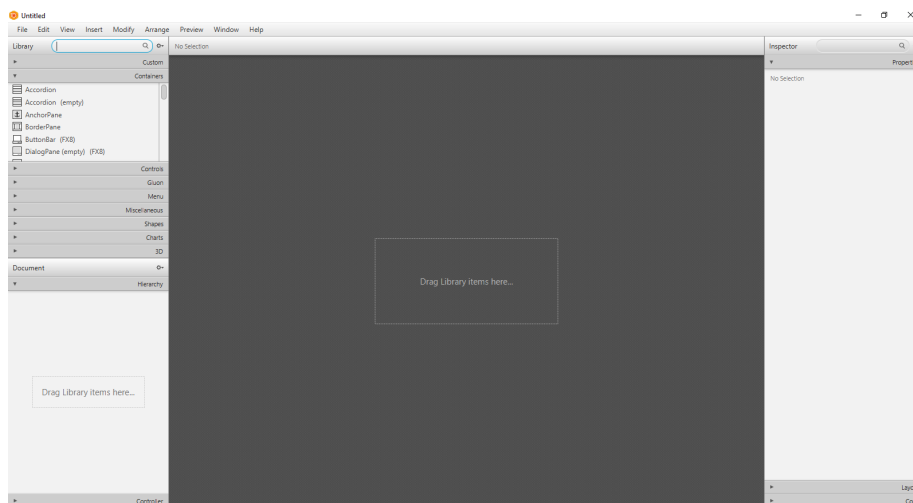


Figura D.2: SceneBuilder 10.0

Eclipse

Para el desarrollo del código Java, se ha utilizado la IDE Eclipse. En ella se han importado las librerías correspondientes, e instalado plugins para el soporte de JavaFX. De esta manera, es posible llamar a SceneBuilder desde Eclipse.

APÉNDICE D. DOCUMENTACIÓN TÉCNICA DE PROGRAMACIÓN

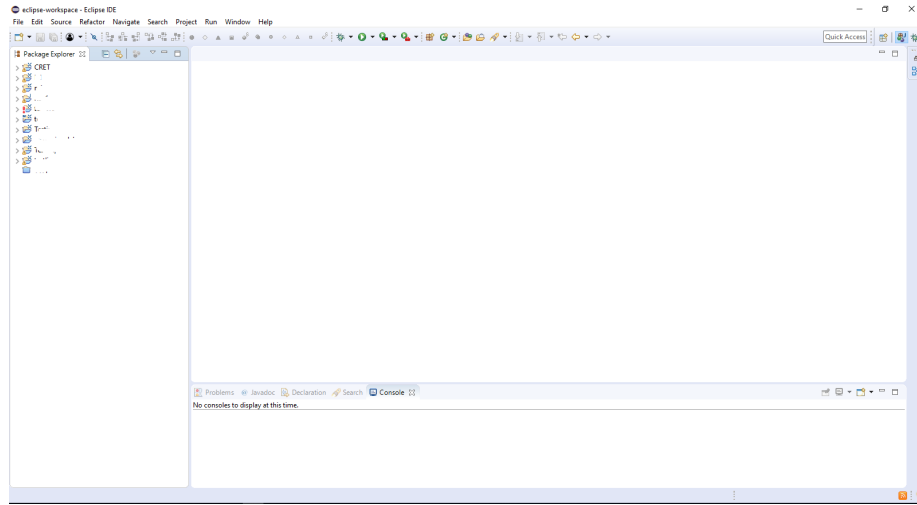


Figura D.3: IDE Eclipse

Obtención del código fuente

El código fuente de la aplicación está hospedado en un repositorio Git en GitHub. Para obtener una copia de este código, se pueden seguir los siguientes pasos:

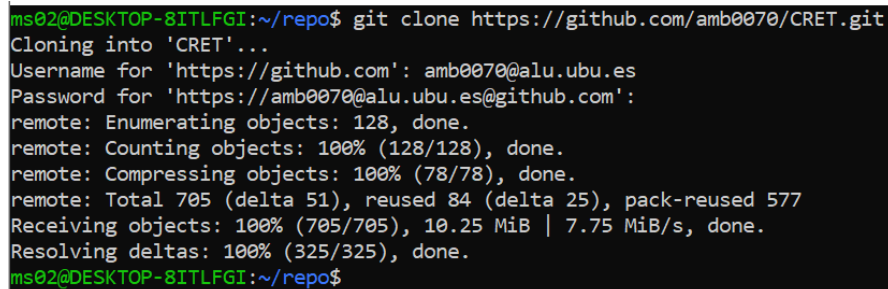
Abrir el WSL en Windows.

Posicionarse en el directorio en el que se desea copiar el código. (Si deseamos acceder al sistema de ficheros de Windows desde WSL, la ruta será `/mnt/c`).

Introducir el siguiente comando:

```
git clone https://github.com/amb0070/CRET.git
```

Una vez finalizada la descarga del repositorio, se podrá acceder al código fuente y a los recursos del proyecto.



```
ms02@DESKTOP-8ITLFGI:~/repo$ git clone https://github.com/amb0070/CRET.git
Cloning into 'CRET'...
Username for 'https://github.com': amb0070@alu.ubu.es
Password for 'https://amb0070@alu.ubu.es@github.com':
remote: Enumerating objects: 128, done.
remote: Counting objects: 100% (128/128), done.
remote: Compressing objects: 100% (78/78), done.
remote: Total 705 (delta 51), reused 84 (delta 25), pack-reused 577
Receiving objects: 100% (705/705), 10.25 MiB | 7.75 MiB/s, done.
Resolving deltas: 100% (325/325), done.
ms02@DESKTOP-8ITLFGI:~/repo$
```

Figura D.4: Comando Git sobre WSL.

Importar proyecto a Eclipse

Una vez descargado el proyecto, abrimos Eclipse.

Hacemos *click* en *File > Import...*

Seleccionamos *Projects from File System or Archive*, y seleccionamos el directorio donde hemos clonado el repositorio.

De esta manera, queda importado el proyecto a nuestro *workspace* de Eclipse.

CAPTURAS

Importar librerías necesarias

Para importar las librerías necesarias y utilizadas por el proyecto, podemos seguir los siguientes pasos:

Sobre la carpeta del proyecto, en el árbol izquierdo, hacer click derecho.

Seleccionar sub-menú *Build Path > Configure Build Path*.

Hacemos *click* en el botón *Add External JARs...* y seleccionamos los ficheros .jar de las librerías.

CAPTURAS

D.4. Compilación, instalación y ejecución del proyecto

Compilación

La compilación del proyecto se realiza desde Eclipse.

Para exportarlo como un fichero .jar, podemos hacer click derecho sobre el árbol izquierdo del proyecto, y *click* en *Export...*

Seleccionamos *Java >Runnable JAR file*.

Incluimos las librerías en el fichero .jar, y exportamos.

CAPTURAS

Ejecución

No es necesario realizar una instalación de la aplicación. Simplemente tener instalado en la máquina el JRE.

La ejecución de la aplicación no requiere de ningún elemento externo, pero si que es necesario tener el *hardware* conectado a la máquina para su funcionamiento.

De igual manera, el proyecto puede ser ejecutado desde Eclipse, pero para su funcionamiento es necesario el *hardware* que se ha diseñado.

Apéndice E

Documentación de usuario

E.1. Introducción

En el siguiente manual se definen los requerimientos de la aplicación, como instalarla y ejecutarla.

E.2. Requisitos de usuarios

Los requisitos mínimos para la utilización de la aplicación son los siguientes:

Hardware para conectarse al bus CAN (usbtin) Java

E.3. Instalación

No es necesaria una instalación de la aplicación. Simplemente la utilización de una versión de Java compatible.

E.4. Manual del usuario