UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

TFG del Grado en Ingeniería Informática

Análisis y predicción de datos obtenidos del funcionamiento de un AGV
Documentación Técnica

Presentado por Gonzalo Burgos de la Hera
en Universidad de Burgos — 25 de junio
de 2023
Tutor: Bruno Baruque Zanón y Jesús Enrique
Sierra García

Índice general

Índice de figuras

Índice de tablas

Apéndice A

Plan de Proyecto Software

A.1. Introducción

En el ámbito del desarrollo de proyectos de software, es esencial contar con una planificación sólida y realista, así como evaluar la viabilidad económica y legal del proyecto. Estos aspectos son clave para garantizar el éxito a largo plazo, la rentabilidad y la conformidad con las regulaciones aplicables.

El presente proyecto de software se enfoca en el análisis detallado de la planificación temporal, la viabilidad económica y la viabilidad legal, con el objetivo de proporcionar una base sólida para su implementación exitosa. Se realizará un exhaustivo examen de cada uno de estos aspectos, considerando tanto los recursos disponibles como las restricciones y requisitos específicos del proyecto.

En primer lugar, se llevará a cabo un análisis de la planificación temporal del proyecto. Esto implica establecer un cronograma realista y eficiente, teniendo en cuenta las tareas necesarias, los hitos clave y las dependencias entre ellas. Se identificarán los recursos requeridos y se asignarán de manera adecuada para asegurar una ejecución eficiente y oportuna del proyecto. Además, se considerarán posibles riesgos y se desarrollarán estrategias de mitigación para evitar retrasos y garantizar la finalización exitosa del software en el tiempo establecido.

En segundo lugar, se realizará un estudio exhaustivo de la viabilidad económica del proyecto de software. Esto implica evaluar los costos asociados con el desarrollo, implementación y mantenimiento del software, así como proyectar los beneficios y retornos de inversión esperados. Se analizarán los posibles ingresos generados por el software, los gastos operativos, el costo

de adquisición de herramientas y tecnologías necesarias, entre otros factores relevantes. Este análisis permitirá tomar decisiones informadas sobre la asignación de recursos financieros y evaluar la rentabilidad y sostenibilidad del proyecto.

Por último, pero no menos importante, se llevará a cabo un análisis exhaustivo de la viabilidad legal del proyecto de software. Se examinarán las leyes y regulaciones pertinentes, como la protección de datos, los derechos de propiedad intelectual y cualquier otro requisito legal aplicable. Se asegurará que el software cumpla con todas las normativas y se evitarán posibles problemas legales en el futuro. Además, se considerarán las implicaciones éticas y de privacidad para garantizar el cumplimiento de estándares éticos y legales relevantes.

En resumen, este proyecto de software se enfoca en el análisis riguroso de la planificación temporal, la viabilidad económica y la viabilidad legal. Al considerar estos aspectos de manera integral, se establecerán las bases necesarias para el desarrollo exitoso del software, asegurando su viabilidad financiera, su cumplimiento legal y su capacidad para cumplir con los objetivos establecidos.

A.2. Planificación temporal

A.3. Estudio de viabilidad

Viabilidad económica

En este apartado, se analizan los costes y beneficios que podría haber incurrido con el desarrollo de este proyecto en el caso de que se hubiese desarrollado en un entorno profesional.

Costes

Costes materiales En este apartado se analizan todos los costes materiales relacionados con el proyecto, principalmente hardware. Se estima una vida útil del hardware de 4 años, habiéndose utilizado 6 meses para el desarrollo de este proyecto.

Elemento	Coste	Coste amortizado
Ordenador	1500€	375€
Total	1500€	375€

Tabla A.1: Costes de hardware

Costes software Debido a que el proyecto ha sido desarrollado utilizando únicamente software de código abierto, los costes de software han sido de cero. Convendría, sin embargo, en el supuesto caso de aplicar este proyecto en un entorno industrial real, estudiar si merece la pena utilizar la versión empresarial y de pago de la base de datos utilizada, InfluxDB.

Costes personales El desarrollo del proyecto ha sido llevado a cabo por un único programador. Se supone que ha sido a jornada completa y que se trata de un desarrollador junior.

Elemento	Coste
Salario	1200€
IRPF	
Seguridad Social	
Salario bruto	
Total 6 meses	

Tabla A.2: Costes personales

Otros costes

Coste total

Beneficios

Viabilidad legal

Apéndice B

Especificación de Requisitos

B.1. Introducción

Una muestra de cómo podría ser una tabla de casos de uso:

B.2. Objetivos generales

B.3. Catalogo de requisitos

B.4. Especificación de requisitos

CU-1	**Ejemplo de caso de uso**
Versión	1.0
Autor	Alumno
Requisitos asociados	RF-xx, RF-xx
Descripción	La descripción del CU
Precondición	Precondiciones (podría haber más de una)
Acciones	1. Pasos del CU 2. Pasos del CU (añadir tantos como sean necesarios)
Postcondición	Postcondiciones (podría haber más de una)
Excepciones	Excepciones
Importancia	Alta o Media o Baja...

Tabla B.1: CU-1 Nombre del caso de uso.

Apéndice C

Especificación de diseño

C.1. Introducción

En esta sección se detalla la organización y flujo de datos del proyecto, el diseño de los diferentes servicios mencionados en apartados anteriores, y un diseño más en profundidad de toda la arquitectura.

C.2. Diseño de datos

Esta sección se divide en otras dos subsecciones: la primera detallará la organización de los datos en la base de datos, y la segunda detallará el flujo de datos entre los diferentes servicios.

Base de datos

Como se ha especificado en el anexo B, la base de datos elegida es InfluxDB. Esta es una base de datos de series temporales no relacional, por lo que las características de esta base de datos no siguen las de una base de datos relacional.

El elemento de mayor nivel organizativo en InfluxDB es la organización. Una organización es un entorno de trabajo en el que se definen el resto de elementos de la base de datos: usuarios, "buckets", tareas, etc.

En InfluxDB, los datos se almacenan en lo que se conoce como "buckets". La característica principal de estos "buckets" es que tienen un periodo de retención, es decir, el tiempo que perduran los datos almacenados. Así, por

ejemplo, un "bucket" con un periodo de retención de un minuto, los datos insertados hace más de un minuto son automáticamente eliminados.

Dentro de cada "bucket", los datos están organizados según los siguientes elementos:

1. Time: Al ser InfluxDB una serie de datos de series temporales, este campo cobra una gran importancia. Este campo almacena el tiempo de cada entrada según el estándar RFC3339 [1].

2. Field set: El conjunto de campos, o field set, almacena pares de claves y valore: las claves guardan metadatos y son cadenas de caracteres, y los valores almacenan los datos propiamente dichos.

3. Tag set: el conjunto de etiquetas, o tag set, almacena, al igual que el field set, pares de clave-valor. El tag set, sin embargo, solo almacena metadatos, y tanto la clave como el valor almacenan cadenas de caracteres. Este elemento es opcional, aunque es muy recomendado utilizarlo, pues la etiquetas están indexadas, por lo que las consultas en este tipo de datos son mucho más rápidas.

4. Measurement: la medida, o measurement, actúa como contenedor de los campos anteriores, y sería el equivalente a una tabla en una base de datos relacional.

Con esta información en mente, la organización de la base de datos queda definida de la siguiente manera. Se utilizan dos "buckets", uno para los datos recibidos por el AGV o por el simulador con un periodo de retención infinito (los datos no se eliminan nunca), y otro para almacenar las predicciones realizadas con un periodo de retención bajo, especificado en un archivo de configuración.

En cada uno de estos "buckets" se almacenan las series temporales siguiendo el esquema de las tablas C.1 y **??**

Elemento	Descripción	
Measurement	AGVDATA	
Tag set	Clave agvid	Valor string
Field set	Clave encoder_izquierdo in.current_l in.current_h in.i_medida_bat in.guide_error out.set_speed_right out.set_speed_left out.display	Valor int int int int float int int int

Tabla C.1: Bucket AGV

Flujo de datos

La comunicación entre servicios ocurre de tres formas diferentes:

- 5G Wifi: Los AGV envían datos al "AGV Coordinator" como cadenas de bytes a este servicio. Como este servicio no está desarrollado como parte de este proyecto, no se detallará más.
- UDP/JSON: Tanto el "AGV Coordinator" como el "Simulator" envían los datos codificados en forma de JSON a través de una conexión UDP. Estos datos son recibidos por el servicio "Reciever", que los introducirá en la base de datos.
- Database API: El "Reciever" es el encargado de insertar los datos de los AGV o del simulador, utilizando para ello la API de la base de datos a través de consultas.

C.3. Diseño procedimental

C.4. Diseño arquitectónico

Apéndice D

Documentación técnica de programación

D.1. Introducción

En este apartado del anexo se detallará la organización de los repositorios del proyecto, así como un manual de programador, una guía de instalación y ejecución del mismo y las pruebas realizadas al sistema.

D.2. Estructura de directorios

La organización de los directorios queda definida de la siguiente manera:

```
memoria
├── build
├── img
└── tex
```

El directorio memoria guarda todos los archivos relacionados con la memoria y estos anexos. En el subdirectorio build se encuentran los archivos pdf finales, en img se encuentran las imágenes utilizadas y en tex se guardan los archivos de LaTeX de cada uno de las secciones de la memoria y los anexos.

```
prototipos
├── datos
├── forecasting
├── graphite
└── influxdb
```

```
├── mongo
├── prometheus
├── sqlite
└── timescaledb
```

En el directorio prototipos se guardan todos los prototipos creados durante la realización del proyecto. Es código realizado de forma rápida con el fin de adecuarme al uso de cada una de las diferentes herramientas consideradas. No tiene mucha más importancia más allá de eso.

```
rendimiento
├── database
└── forecast
```

En el directorio de rendimiento se guarda todo el código de las pruebas de rendimiento realizadas para la comparación de las bases de datos y modelos de predicción. En el subdirectorio de forecast se encuentra también el código utilizado para la optimización de los hiperparámetros de los modelos comparados.

```
servicios
├── forecasting
│   └── src
├── reciever
│   └── src
└── simulator
    ├── src
    └── data
```

En el directorio de servicios se encuentra todo el código y archivos de configuración de cada uno de los servicios desarrollados en este proyecto. El código se encuentra en el directorio "src" de dentro de cada uno de los subdirectorios de cada servicio. En la carpteta "data", dentro del directorio del simulador, se guardan los archivos csv utilizados para leer desde este servicio.

D.3. Manual del programador

D.4. Compilación, instalación y ejecución del proyecto

D.5. Pruebas del sistema

Apéndice E

Documentación de usuario

E.1. Introducción

E.2. Requisitos de usuarios

E.3. Instalación

E.4. Manual del usuario

Bibliografía

[1] Chris Newman and Graham Klyne. Date and Time on the Internet: Timestamps. RFC 3339, July 2002.