ESCUELA TECNICA SUPERIOR DE INGENIERÍA

(ICAI)

Máster en Big Data, Tecnología y Analítica Avanzada

PROCESAMIENTO EN LA NUBE DE DATOS STREAMING DE VEHÍCULOS AUTONOMOS CONECTADOS EN ENTORNOS COMPLEJOS DE CONDUCCIÓN, APLICACIÓN EN ROTONDAS.

Autor

Isabel Sutil Martín

Dirigido por

Sergio Vozmediano Ávilas

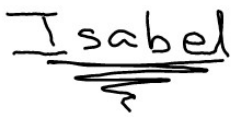
Borja Monsalve Piqueras

Proyecto en colaboración con la Universidad Europea de Madrid, desarrollado dentro del grupo de investigación de Sistemas Inteligentes de Control (SIC)

Madrid

Junio 2022

Isabel Sutil Martín, declara bajo su responsabilidad, que el Proyecto con título **PROCESAMIENTO EN LA NUBE DE DATOS STREAMING DE VEHÍCULOS AUTONOMOS CONECTADOS EN ENTORNOS COMPLEJOS DE CONDUCCIÓN, APLICACIÓN EN ROTONDAS** presentado en la ETS de Ingeniería (ICAI) de la Universidad Pontificia Comillas en el curso académico 2021/22 es de su autoría, original e inédito y no ha sido presentado con anterioridad a otros efectos. El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido tomada de otros documentos está debidamente referenciada.



Fdo.: Isabel Sutil Martín Fecha: 7 / 7 / 2022

Autoriza la entrega:

Directores del Proyecto

Sergio Vozmediano Ávilas

Fdo.: ..... Fecha: 11 / 07 / 2022

Borja Monsalve Piqueras

Fdo.: Borja Monsalve Fecha: 7 / 7 / 2022

ESCUELA TECNICA SUPERIOR DE INGENIERÍA

(ICAI)

Máster en Big Data, Tecnología y Analítica Avanzada

PROCESAMIENTO EN LA NUBE DE DATOS STREAMING DE VEHÍCULOS AUTONOMOS CONECTADOS EN ENTORNOS COMPLEJOS DE CONDUCCIÓN, APLICACIÓN EN ROTONDAS.

Autor

Isabel Sutil Martín

Dirigido por

Sergio Vozmediano Ávilas

Borja Monsalve Piqueras

Proyecto en colaboración con la Universidad Europea de Madrid, desarrollado dentro del grupo de investigación de Sistemas Inteligentes de Control (SIC)

Madrid

Junio 2022

Resumen

Actualmente en Europa no se permiten coches autónomos que superen el nivel 2 de los 6 niveles establecidos por la Sociedad de Ingenieros de Automoción (SAE) Internacional para clasificar los vehículos autónomos. El paso a el nivel SAE 3 implica que el coche se va a poder mover sin que el conductor interactúe. Sin una correcta comunicación nunca se va a poder llevar a cabo una total automatización del vehículo por que el conocimiento del entorno está limitado a los sensores del vehículo. Esto se debe a que los sensores solo proporcionan información del entorno cercano.

En este proyecto se ha diseñado e implementado una forma de comunicación en tiempo real entre vehículos, de forma segura y fiable mediante un sistema Big Data alojado en la nube. Este proyecto está orientado al escenario de conducción compleja que representan las rotondas.

La comunicación se realiza de la siguiente forma: cada vehículo autónomo recoge y publica sus datos en tiempo real en un servicio en la nube. Esto se realiza de forma segura mediante una autentificación con cifrado RSA (clave publica/privada), solo se almacenan datos reales y confiables. La conexión se realiza mediante subscripciones por parte de cada vehículo autónomo, el cual recibe datos de todos los vehículos que afectan en una rotonda de forma autenticada.

Los datos de todos los vehículos autónomos conectado se almacenan en tiempo real en una base de datos NoSQL en la nube.

Palabras clave:

Vehículos Autónomos Conectados, Grandes volúmenes de datos, Cloud Computing, rotondas.

Abstract

At the moment, autonomous cars are not allowed in Europe above level 2 of the 6 levels established by the International Society of Automotive Engineers (SAE) to classify autonomous vehicles. Upgrading to SAE level 3 implies that the car will be able to move without the driver's intervention. Without a proper communication it will never be possible to fully automatize the vehicles because the knowledge of the environment is limited to the vehicle's sensors. This is because the sensors only provide information from the immediate environment.

This project has designed and implemented a secure and reliable way of real-time communications between vehicles through a Big Data system hosted in the cloud. This project focused on the complex driving scenario of traffic roundabouts.

Communications are performed as follows: each autonomous vehicle collects and publishes its data in real-time in a cloud service. This is done in a secure way by authentication with RSA encryption (public/private key), that way only real and reliable data is stored. The connections are made via authenticated subscriptions from each autonomous vehicle, each of which gets data from all the vehicles that affect a roundabout.

The data of all the connected autonomous vehicles is stored in a real-time NoSQL database hosted in the cloud.

Keywords:

Connected Autonomous Vehicles, Big Data, Cloud Computing, roundabouts.

Agradecimientos

En primer lugar, quiero agradecer a mis tutores del Trabajo Final de Master, Sergio y Borja, por toda su dedicación y ayuda.

Quiero dar las gracias a todos los miembros del equipo de investigación de Sistemas Inteligentes de Control (SIC) de la Universidad Europea de Madrid. A los Investigadores Principales: Javier Fernández Andrés y Nourdine Aliane. A Enrique Puertas por sus conocimientos en Big Data y a todos los compañeros de trabajo que me han ayudado y apoyado.

A todos los profesores de la Universidad Pontificia Comillas de ICAI que me han acompañado en el máster de Big Data, Analítica Avanzada y Tecnología, por la sabiduría y el apoyo que me han transmitido este año.

Por último, quiero agradecerles a todos mis amigos y familiares por acompañarme en esta etapa. Especialmente a mis padres Desiderio Sutil y Elvira Martín, por vuestro cariño y vuestros consejos. El apoyo siempre ha sido incondicional.

Índice de la memoria

Índice de ilustraciones

Índice de tablas

Capítulo 1. RESUMEN DEL PROYECTO

1.1 CONTEXTO Y JUSTIFICACIÓN

Actualmente en Europa no se permiten coches autónomos que superen el nivel 2 de los 6 niveles establecidos por la Sociedad de Ingenieros de Automoción (SAE) Internacional para clasificar los vehículos autónomos. Esto significa que siempre tiene que haber una persona responsable de conducir el vehículo. En la actualidad resulta complicado alcanzar una mayor autonomía en los vehículos debido a que el conocimiento del entorno está limitado a los sensores de cada vehículo. Esto se debe a que los sensores solo proporcionan información del entorno cercano.

1.2 PLANTEAMIENTO DEL PROBLEMA

El paso a el nivel SAE 3 implica que el coche se va a poder mover sin que el conductor interactúe con él. Uno de los principales retos a tener en cuenta es que la información del entorno está limitada a los datos que se pueden recoger de los sensores de cada vehículo. Para poder llegar a conocer mejor el entorno se necesita una correcta comunicación y conectividad entre los vehículos autónomos. Al poder disponer de la información de los sensores de los demás vehículos se va a poder conocer mejor el entorno, siempre teniendo en cuenta la complejidad que suponen los elementos no conectados al sistema.

1.3 OBJETIVOS DEL PROYECTO

El objetivo principal es el diseño e implementación de un sistema de procesamiento de datos en la nube orientado a la comunicación entre vehículos autónomos para solventar el entorno complejo de conducción que representan las rotondas.

1.4 DESCRIPCIÓN DE LA SOLUCIÓN

En este proyecto se ha diseñado e implementado una forma de comunicación real-time entre vehículos, de forma segura y fiable.

Los datos de los sensores de cada vehículo autónomo conectado se extraen del simulador de conducción realista CARLA Simulator. Estos datos son recolectados en el entorno complejo de conducción de una rotonda.

Para elaborar la conexión se ha diseñado e implementado una arquitectura Big Data en Google Cloud Platform para el procesamiento de datos de forma segura. La comunicación se realiza de la siguiente forma: cada vehículo autónomo recoge y publica sus datos en real time dentro del servicio de Pub/Sub de Google Cloud Platform. Esto se realiza de forma segura mediante una autentificación con cifrado RSA (clave publica/privada), solo se almacenan datos reales y confiables. La conexión se realiza mediante subscripciones al topic por parte de cada vehículo autónomo, el cual recibe datos de todos los vehículos que afectan en una rotonda de forma autenticada.

Los datos de todos los vehículos autónomos conectado se almacenan real-time en una base de datos NoSQL en la nube: Cloud Firestore.

1.5 RESULTADOS OBTENIDOS

Se ha implementado un sistema Big Data que permite la conexión entre vehículos autónomos en un entorno simulado. Para ello se han recogido datos de los sensores de cada vehículo autónomo conectado y, en real time, se han procesado y almacenado en

la nube de forma automática. Todo ello se ha realizado en un entorno de conducción complejo: las rotondas.

Cada vehículo recoge y publica sus datos en la nube de forma segura para realizar la comunicación. Esto ocurre en tiempo real (cada 0.3 segundos). Así mismo, los datos de todos los vehículos autónomos conectado se almacenan en tiempo real en una base de datos NoSQL en la nube. Mediante la comunicación creada entre los vehículos autónomos, cada vehículo dispone de un mayor conocimiento del entorno ya que dispone de los datos fiables en tiempo real de los sensores de los demás vehículos autónomos que afectan en la rotonda.

Capítulo 2. CONTEXTO Y JUSTIFICACIÓN

2.1 MOTIVACIÓN Y CONTEXTO DEL PROYECTO

El vehículo autónomo es uno de los principales retos en el ámbito del transporte. Se están llevando a cabo numerosas investigaciones para llegar a conseguir un vehículo totalmente autónomo, sin la necesidad de un conductor. [1]

Actualmente la conducción autónoma en Europa se ve restringida por la Ley sobre responsabilidad civil y seguro en la circulación de vehículos a motor. Todos los avances tecnológicos tienen que ir acompañados de unas leyes que los regulen. Al analizar la última versión publicada de la ley (07/03/2022) podemos leer lo siguiente: "El conductor de vehículos a motor es responsable, en virtud del riesgo creado por la conducción de estos, de los daños causados a las personas o en los bienes con motivo de la circulación" (Ley sobre responsabilidad civil y seguro en la circulación de vehículos a motor, 2004, Capitulo 1, Artículo 1.1) [2]. Esto nos quiere decir que en Europa no se permiten coches autónomos que superen el nivel 2 de los 6 niveles establecidos por la Sociedad de Ingenieros de Automoción (SAE) Internacional para clasificar los vehículos autónomos. [3]

- SAE 0: No es un vehículo autónomo
- SAE 1: Conducción asistida: una persona conduce el vehículo, pero hay elementos que pueden ayudar a la conducción. Un ejemplo son los sistemas mantenimiento del carril (donde se indica al conductor cuando el coche se sale del carril). De esta forma se pueden evitar errores humanos.
- SAE 2: Conducción parcialmente autónoma: una persona conduce el vehículo, pero hay elementos que permiten que alguna tarea se realice de forma autónoma bajo la supervisión del conductor. Un ejemplo son los sistemas crucero de algunos vehículos. Mediante ellos el coche puede decidir cuándo

frenar/acelerar en función de las condiciones impuestas por el conductor y siempre bajo la supervisión del mismo.

- SAE 3: Conducción condicionalmente autónoma: el coche puede ir autónomamente bajo unas condiciones limitadas y en unos escenarios concretos pero el conductor debe intervenir si es requerido.
- SAE 4: Conducción altamente autónoma: No se necesita la intervención del conductor en cualquier momento, sin embargo, la conducción autónoma solo podrá realizarse bajo unas condiciones limitadas y en unos escenarios concretos.
- SAE 5: Conducción completamente autónoma: Conducción autónoma sin intervención del conductor en cualquier circunstancia.

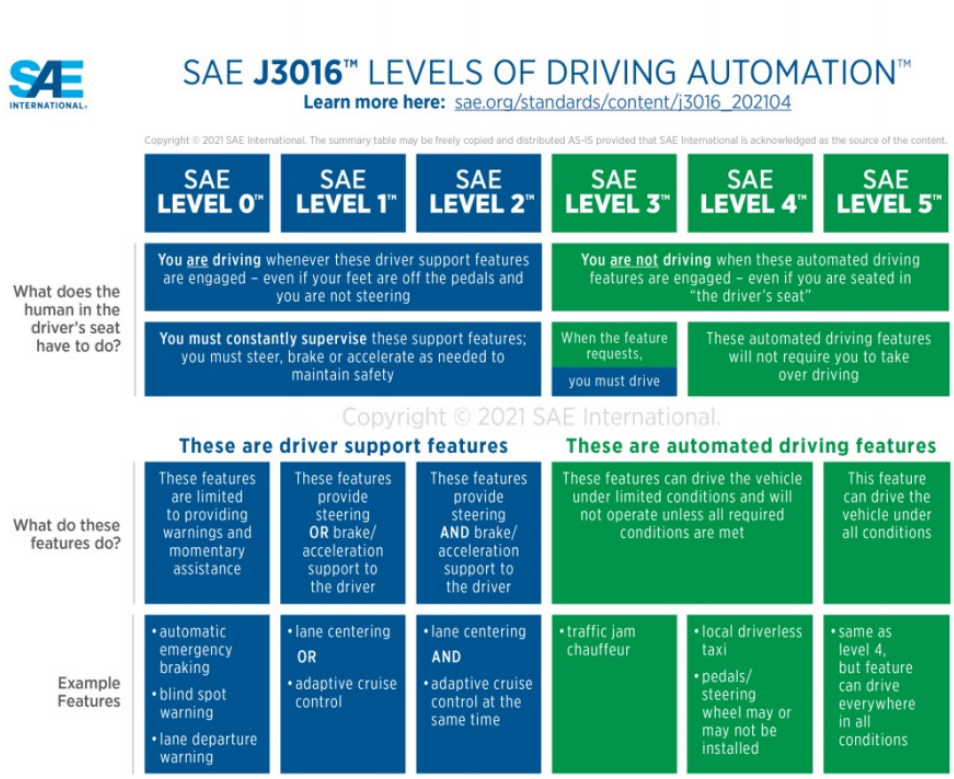


Ilustración 1. Niveles SAE de vehículos autónomos [4]

Pese a que aún podemos ver coches con distintos niveles de autonomía en la conducción en nuestras carreteras el gobierno está apostando por la innovación. Desde la Unión Europea (UE) se han establecido una serie de ayudas al conductor (ADAS) que

han de ser obligatorios en los nuevos coches homologados. [5] Por ejemplo, se han de incluir detectores de fatiga y atención, alertas de cambio de carril involuntario (LDW) y sistemas de frenada de emergencia entre otros. Con esto se pretende que todos los coches dispongan de una mayor cantidad de medidas de seguridad obligatorias y se avance en el camino de los coches autónomos. Estos coches se asocian con un nivel SAE 2.

2.2 ESTADO DEL ARTE

El concepto de la conducción autónoma se considera que fue introducido en la Exposición Mundial de Nueva York "Futurama" en 1939-1940 cuando el diseñador industrial Norman Bel Geddes presento la idea de un coche eléctrico que se guiaba captando señales electromagnéticas generadas por elementos incrustados en la calzada. Siguiendo esta idea General Motors construyó un prototipo en 1958 que se guiaba captando la intensidad que circulaba por un cable incrustado en la carretera.

Sin embargo, los primeros avances significativos llegaron de la mano del profesor Ernst Dickmanns, experto en Inteligencia y visión artificial de la Universidad de Munich. En el año 1987 consiguió implementar en una furgoneta un sistema que, mediante computación paralela, técnicas de visión artificial y cálculos probabilísticos, logro circular por una autopista sin tráfico llegando a alcanzar velocidades de hasta 100 km/h. Posteriormente, en 1995, Dickmanns logro adaptar un Mercedes-Benz Clase S para que realizara un trayecto Munich – Copenague. El vehículo estaba supervisado por una persona que tuvo que intervenir en intervalos medios de 9 km. El coche logro realizar un 95% del trayecto de forma totalmente autónoma, llegando a realizar adelantamientos y alcanzando velocidades de 175 km/h en autopistas alemanas. A raíz de estos logros la Comisión Europea financio con 800 millones de euros un proyecto EUREKA, denominado Prometheus, para el desarrollo del vehículo autónomo.

El siguiente gran hito tiene lugar en Estados Unidos, 2004, cuando la Agencia de Proyectos de Investigación de la Defensa (DARPA) fundó una competición el objetivo de acelerar el desarrollo de tecnologías de vehículos autónomos (DARPA Grand

Challenge). [6] Competición ganada por Google en 2005, momento en el que la empresa comienza el desarrollo de coches autónomos.

Actualmente encontramos numerosas empresas investigando sobre vehículos autónomos:

- Google, con su empresa Waymo, están mejorando su sistema de conducción autónomo Waymo Driver mediante cámaras de visión y un LiDAR perimetral.
- Tesla está trabajando en su versión 4.0 del sistema Autopilot. Modelo construido mediante cámaras de visión, sensores ultrasónicos y un sensor radar.
- Amazon ha comprado la empresa Zoox y a finales de 2020 presento su primer modelo. Este vehículo se ha diseñado con sensores radar, LiDAR y cámaras de visión.
- Uber y Volvo han presentado el modelo conjunto XC90. Vehículo con cámaras de visión y sensores radar y LiDAR.
- Apple trabaja en su proyecto Titan con el fin de desarrollar software de conducción autónoma que puedan vender a fabricantes de automóviles

En general, todos los vehículos autónomos consisten en un sistema que toma decisiones en función de la información de los datos extraídos de los sensores del vehículo.

Este proyecto final de máster se ha realizado en colaboración con la Universidad Europea de Madrid (UEM) dentro del grupo de investigación de Sistemas Inteligentes de Control (SIC). El trabajo se encuentra en el marco de dos líneas de investigación que se están realizando en el centro:

- El proyecto “Sistema de arbitraje distribuido para conducción cooperativa conectada y autónoma en entornos complejos (CCAD)” aprobado en la convocatoria nacional del Ministerio de Ciencia, Innovación y Universidades de 2019 (Plan Nacional de I+D+I) [7].
- El programa científico “SEGuridad de los Vehículos AUTOmóviles (SEGVAUTO 4.0)” financiado por la Comunidad de Madrid y la UE. Programa liderado por el

Instituto Universitario de Investigación del Automóvil (INSIA) de la Universidad Politécnica de Madrid (UPM) [8]. Dentro de este programa se engloban objetivos relacionados con el transporte automatizado, en donde se centra este proyecto.

- Objetivo 4: Conducción autónoma: comunicaciones, servicios cooperativos e integración con la infraestructura
- Objetivo 5: Fusión sensorial e interpretación del entorno.
- Objetivo 6: Sistemas inteligentes de localización, navegación y predicción de intenciones.

2.3 PLANTEAMIENTO DEL PROYECTO

Actualmente, en junio de 2022, se va a presentar un proyecto en la UE respaldado por la Comisión Europea de las Naciones Unidas para Europa (UNECE) en el que, si entra en vigor, se permitirán los coches autónomos de nivel SAE 3 en Europa. [9] Se propone aceptar un sistema de mantenimiento en el carril (ALKS) que se activará por el conductor cuando este bajo unas condiciones previamente establecidas y en carreteras previamente definidas donde los peatones y ciclistas tendrán prohibido el acceso. [10]. El conductor podrá recuperar el control del vehículo en cualquier momento anulando el sistema.

El paso a el nivel SAE 3 implica que el coche se va a poder mover sin que el conductor interactúe. Uno de los principales retos a tener en cuenta es que la información del entono está limitada a los datos que se pueden recoger de los propios sensores del de cada vehículo autónomo Los sensores solo nos proporcionan información del entono cercano. Esta problemática se podría solucionar al conocer los datos de los sensores de los demás vehículos. [7] La comunicación y conectividad es imprescindible en el sistema para el correcto intercambio de información, siempre teniendo en cuenta la complejidad que suponen los elementos no conectados al sistema.

En este proyecto se sigue una línea de investigación sobre los vehículos autónomos y conectados para resolver escenarios de alta complejidad. Se ha diseñado un sistema centrado en la situación de una rotonda, debido a la peligrosidad de las mismas. Según

la Dirección General de Tráfico (DGT) una de cada 5 infracciones son ocasionadas en las rotondas. [11]

Actualmente las tecnologías Big Data permiten procesar una gran multitud de datos de distintos orígenes simultáneamente. Estos datos, denominados streaming, proceden, en caso de este proyecto, de distintos sensores situados en los distintos coches interconectados.

En este proyecto se ha diseñado e implementado una forma de comunicación real-time entre vehículos, de forma segura y fiable.

Los datos de los sensores vehículos autónomos conectados se extraen del simulador de conducción realista CARLA Simulator. Estos datos son recolectados en el entorno complejo de conducción de una rotonda.

Para elaborar la conexión se ha diseñado e implementado una arquitectura Big Data en Google Cloud Platform para el procesamiento de datos de forma segura. Todos los datos real time se almacenan en la nube.

Capítulo 3. OBJETIVOS

3.1 OBJETIVO GENERAL

El objetivo principal es el diseño e implementación de un sistema de procesamiento de datos en la nube orientado a la comunicación entre vehículos autónomos para solventar el entorno complejo de conducción que representan las rotondas.

3.2 OBJETIVOS ESPECÍFICOS

EL objetivo general del proyecto se desglosa en los siguientes objetivos específicos:

1. Diseño de un sistema de comunicación entre los vehículos autónomos.
2. Implementación de un entorno complejo de conducción realista en un simulador de conducción.
3. Procesamiento de los datos en la nube.
4. Almacenamiento de los datos en la nube.
5. Análisis de los resultados obtenidos

3.3 BENEFICIOS DEL PROYECTO

Actualmente, en el ámbito de los coches autónomos, existe la necesidad de establecer comunicación entre los vehículos. Esto se debe a que la información del entorno está limitada a los datos que se pueden recoger de los propios sensores del de cada vehículo autónomo. Los sensores solo nos proporcionan información del entorno cercano.

Esta problemática se podría solucionar al conocer los datos de los sensores de los demás vehículos. Es por ello que, actualmente, se considera que la autonomía total no va a poder llegar a alcanzarse sin una correcta conexión e intercambio de información entre los vehículos.

Es de especial interés tener una correcta comunicación en los entornos complejos de conducción debido a que se pueden ocasionar un mayor número de incidentes. Es por ello que este proyecto se centra en generar una solución a la situación de una rotonda, debido a su nivel de complejidad y número de accidentes que actualmente se registran en las mismas.

Mediante las tecnologías actuales Big Data podemos analizar una gran cantidad de volúmenes de datos en tiempo real que anteriormente no se podía realizar. Esto es esencial para la comunicación entre vehículos debido a la gran cantidad de emisión de datos que nos proporcionan los sensores incorporados en los mismos. Todo este análisis y comunicación tiene que ser en tiempo real. El correcto intercambio de información entre los Vehículos Autónomos Conectados (VAC) mediante el sistema desarrollado en este proyecto permite que cada vehículo pueda disponer de un mayor conocimiento del entorno. Los beneficios a los que aspira este proyecto son:

- Generar una conducción más segura.
- Reducir el número de accidentes en rotondas.
- Reducir el número de heridos o muertos en las carreteras.
- Contribuir a la investigación de los coches totalmente autónomos diseñando sistemas de comunicación eficaces.

Capítulo 4. GESTIÓN DEL PROYECTO

4.1 METODOLOGÍA

Este proyecto se ha desarrollado mediante una metodología Agile. En 2021 se realizó un manifiesto determinando que son las metodologías agiles. Este estaba compuesto por 12 principios fundamentados en los siguientes 4 valores: [12]

1. **Individuals and Iteractions over Processes and Tools**: Individuos e interacciones sobre procesos y herramientas. En este pilar se destaca la importancia del valor humano teniendo en cuenta las habilidades, talentos y creatividad. Todo ello sustentado de los procesos y herramientas disponibles.
2. **Working software over Comprehensive Documentation:** Software funcionando frente a documentación exhautiva. En este pilar se da importancia a tener un producto funcionando del que poder recibir un feedback. Se anima a no tener todo muy planificado y detallado en la documentación cuando las cosas no estan siendo funcionales.
3. **Customer Collaboration over Contract Negotiation:** Colaboración con el cliente frente a negociaciones contractuales. En las metodologías agiles se incluyen al cliente como parte del equipo desarrollador. Para que el producto final sea el esperado se tienen en cuenta sus necesidades desde el primer momento y es parte del proceso aportando feedback.
4. **Responding to Change over Following a Plan:** Respuesta al cambio frente a seguir un plan. Los proyectos no tienen por qué seguir un desarrollo de vida software lineal. Como el cliente siempre forma parte del proceso se han de poder variar los objetivos del proyecto. Los requisitos pueden ser cambiantes y el proyecto se ha de adaptar.

Este proyecto, al formar parte de una investigación puede llegar a tener muchos enfoques distintos. Se ha escogido la metodología Agile debido a la incertidumbre a la

hora de realiza este proyecto. El proyecto se ha adaptado a todos los cambios que han sido requeridos ya que se ha mantenido una comunicación constante para poder obtener el mejor resultado. La metodología ágil por la cual se ha desarrollado este proyecto ha sido Scrum.

Scrum es una metodología ágil orientada a solucionar las necesidades del cliente de una forma colaborativa. [13]. La principal característica de Scrum es su sistema iterativo donde cada cierta cantidad de tiempo (máximo 4 semanas) se inicia un ciclo llamado Sprint. Al final de cada ciclo se ha de entregar una parte del producto donde se pueda ver empíricamente lo realizado, por lo que Scrum se considera una metodología ágil de prototipado rápido. Cada Sprint comienza tras haber acabado el anterior, en este momento se mira con retrospectiva lo realizado y se muestra un prototipo al cliente, o su representante, (Product Owner). El Product Owner aporta el valor de negocio expresando la visión final que se tiene del producto, pudiendo cambiar sus necesidades en cada Sprint. Por otro lado, da feedback sobre el prototipo enseñado en cada ciclo. Posteriormente el responsable denominado Scrum Master ayuda al Product Owner a definir sus necesidades en requisitos claros. El Scrum Master, así mismo, es el encargado de trasmitir al equipo de desarrollo (Scrum Team) las nuevas implementaciones y lidera el equipo. En cada Sprint se han de organizar las tareas a realizar de ese Sprint (y de pasado si no se han realizado) según su prioridad.

Cuanto más cortos son los Sprints existe un menor riesgo de que los requisitos cambien y suponga más costes para el proyecto. Es un sistema muy útil en proyectos con necesidades cambiantes o poco definidos en donde se necesita recibir resultados en poco tiempo. Todo esto está sustentado bajo los 3 pilares de Scrum: transparencia, inspección y adaptación.

En este proyecto los ciclos Scrum han tenido una duración de 2 semanas. Al final de cada Sprint se realiza una reunión con el Investigador Principal (IP) del proyecto que ejerce de Product Owner. Los demás miembros del proyecto son el Scrum Master, investigador que está realizando su tesis doctoral, y el Scrum Team que da soporte y desarrolla una parte de la investigación. El Scrum Master y el Scrum Team han realizado reuniones semanales adicionales.

4.2 PLANIFICACIÓN

El proyecto se ha diseñado en función de los objetivos específicos del proyecto:

- **OE1: Diseño de un sistema de comunicación entre los vehículos autónomos.**
  Se ha realizado un estudio del arte sobre los vehículos autónomos para poder realizar un diseño exhaustivo de la problemática a resolver. Así mismo, se han investigado distintas tecnologías y herramientas para diseñar un sistema que solucione la problemática.

- **OE2: Implementación de un entorno complejo de conducción realista en un simulador de conducción.**
  Se ha implementado en un servidor local un simulador de conducción realista adaptado a las necesidades de la problemática a resolver. Posteriormente se han recogido los datos de los vehículos autónomos.

- **OE3 y OE4: Procesamiento de los datos en la nube y Almacenamiento de los datos en la nube.**
  Se ha diseñado e implementado un sistema Big Data en la nube para procesar los datos de los destinos vehículos autónomos y poder llevar a cabo su conexión. Así mismo, se han almacenado los datos en la nube.

- **OE5: Análisis de los resultados obtenidos.**
  Se ha realizado un análisis continuo de los resultados obtenidos. Los datos recogidos del simulador son realistas y correctos. La conexión entre los vehículos y el almacenamiento se realiza de forma correcta, segura y en tiempo real

El diseño de la planificación se ha realizado mediante un Diagrama de Gantt.

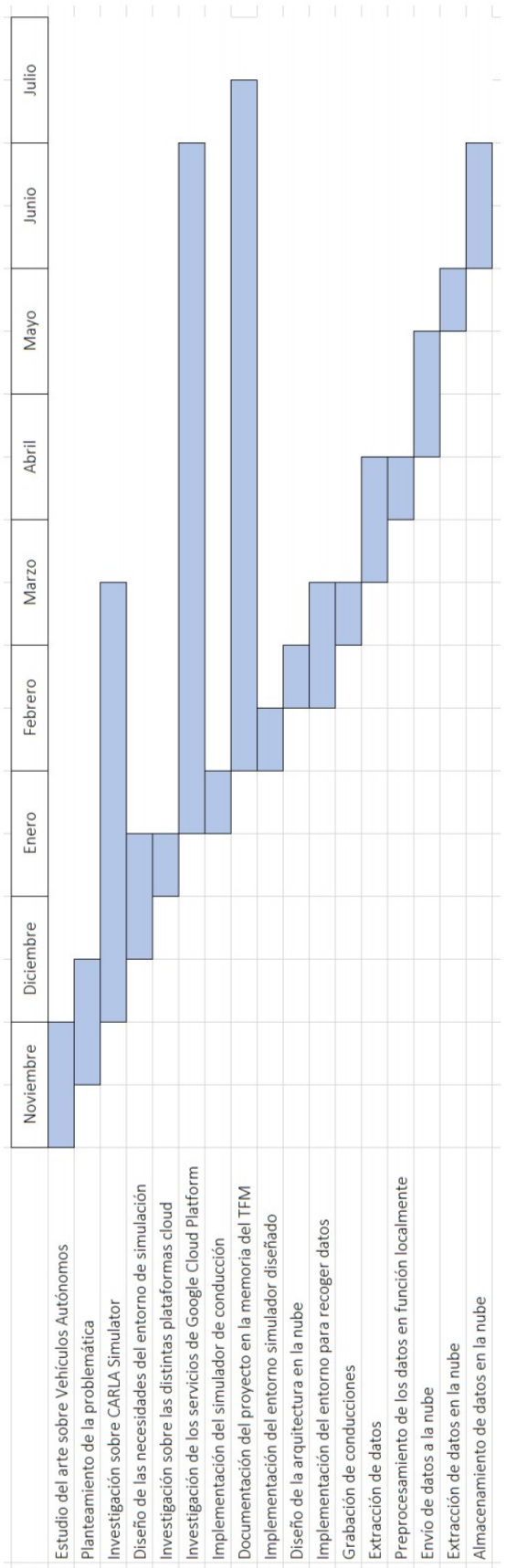


Ilustración 2. Diagrama de Gantt

Capítulo 5. DESCRIPCIÓN DE LA SOLUCIÓN

5.1 ENTORNO OPERACIONAL LOCAL

Actualmente, el proyecto se está desarrollando en un simulador realista. Esto nos permite ahorrar costes ya que no se utilizan vehículos físicos hasta que se llega a fases más avanzadas en la investigación.

Para este proceso se ha elegido el simulador hiperrealista CARLA Simulator (Car Learning to Act). [14] Es un simulador open-souce para la investigación de coches autónomos en entornos urbanos desarrollado por el Centro de Visión de Computador de Barcelona en colaboración con el centro Embodies AI Foundation.

CARLA nos permite adaptar el entorno con las características deseadas ya que garantiza la flexibilidad en la especificación de sensores, actores estáticos o dinámicos (vehículos autónomos en este proyecto) para generar distintas situaciones de tráfico e incluso en condiciones ambientales. [15] Es un simulador tridimensional (3D) con un alto realismo de renderizado y de simulación física (forma en la que se mueven los agentes, datos que se recogen, …) [14]

5.1.1 PREPARACIÓN DEL ENTORNO

CARLA se ha construido en un servidor local con Windows 10 desde su código fuente. Para ello se necesita previamente construir el motor de juego de CARLA: Unreal Engine. [16] Unreal Engine también se ha construido a partir de su código fuente, ya que se necesita instalar una versión específica para que CARLA pueda funcionar. Las tecnologías que se han necesitado para realizar este proceso son:

- **Git**: Software de control de versiones mediante el cual vamos a descargar los códigos fuente de Carla Simulator y de Unreal Editor (la rama correspondiente de CARLA). Estos códigos están alojados en GitHub.

- **Visual Studio 2019**: Se han de descargar todas las herramientas necesarias para poder compilar y ejecutar un programa desarrollado en C++. Se utiliza para compilar todos los códigos fuente de Unreal Engine con la configuración de un "Development Editor".
- **Make**: Es una herramienta que se utiliza para generar los ejecutables de Carla Simulator desde la línea de comandos x64 Native Tools Command Prompt de Visual Studio. Primeramente, se ha de compilar la interfaz de programación de aplicaciones (API). Esta API en Python tiene el rol de cliente de la aplicación, controlando toda la simulación. Posteriormente, mediante Make, se compila y ejecuta el servidor de la aplicación.

Ilustración 3. CARLA Simulator

5.1.2 RECOLECCIÓN DE DATOS DE LOS VEHÍCULOS AUTÓNOMOS

El entorno complejo diseñado está formado por una ciudad simulada que consta de una rotonda de 4 salidas, con un numero cambiante de vehículos que afectan en la misma. La forma de conducción de los vehículos autónomos, y su física, se ha configurado para

que sea una conducción normal. El límite de velocidad dentro de la rotonda se ha determinado en 30 km/h. Los vehículos pueden superar moderadamente de forma autónoma el límite para simular entornos realistas. Se han configurado una serie de sensores en cada vehículo. A continuación, se describen los datos que se recogen en CARLA Simulator:

- Se recogen los datos de la situación de los componentes físicos del vehículo:
    - Pedales de aceleración: float. Valor entre 0.0 (pedal no presionado) y 1.0
    - Pedales de freno: float. Valor entre 0.0 (pedal no presionado) y 1.0
    - Freno de mano: boolean. Determina si se está usando o no (false)
    - Marcha atrás: boolean. Determina si se está usando o no (false)
- Se recoge el límite de velocidad de la carretera mediante información del sistema GPS y cámaras de visión: int. Se indica en kilómetros/hora
- Se recoge la distancia a la rotonda mediante el sistema GPS: float. Se indica en metros
- Se recogen los datos del GPS:
    - Latitud: float
    - Longitud: float
    - Altitud: float
- Se recogen los datos de un sensor de orientación:
    - Rotación roll: float
    - Rotación pitch: float
    - Rotación yaw: float
- Se recoge la información de la velocidad mediante un acelerómetro:
    - Eje x: float. Se indica en metros/segundo
    - Eje y: float. Se indica en metros/segundo
    - Eje z: float. Se indica en metros/segundo
    - Módulo: float. Se indica en kilómetros/hora

Entre los datos extraídos se encuentra un timestamp del instante en el que se recogen los datos junto con un identificador único de cada vehículo. A continuación, se adjuntan los datos extraídos de un vehículo en la rotonda en un instante:


```
"Id": 2684,
"Timestamp": 2022-06-30 20:30:56.277247,
"Pedal acelerador": 0.397482693195343,
"Pedal freno": 0.0,
"Freno de mano": False,
"Marcha atras": False,
"Limite velocidad": 30.0,
"Latitud": -0.0005183581158973993,
"Longitud": -8.971360362742621e-05,
"Altitud": 0.0018423652509227395,
"Rotacion roll": -0.0010681151179596782,
"Rotacion pitch": 0.00027320755179971457,
"Rotacion yaw": 89.52552795410156,
"Distancia rotonda m": 56.90376407816831,
"Velocidad x": 0.06551742553710938,
"Velocidad y": 8.06859302520752,
"Velocidad z": -2.4160059183486737e-05,
"Velocidad total km_h": 29.04789248436416
```


Para poder obtener los datos de los sensores de los vehículos que afectan en la rotonda se han filtrado los vehículos que se encuentran en un radio cercano a la rotonda.

CARLA Simulator es un simulador de conducción realista que consume muchos recursos computacionales. El reto ha sido poder realizar una conducción realista de un vehículo (simulando su conducción autónoma en el entorno) sin que los FPS por segundo en el simulador se vieran ralentizados por el envío de datos a la nube.

La solución implementada ha sido mediante el almacenamiento de todos los datos de la simulación en un fichero cada 0.3 segundos durante la conducción del vehículo. Con esto se ha conseguido poder tener la situación de vehículos en una rotonda de forma realista debido a que la extracción de datos se puede llevar a cabo en tiempo real en el servidor donde se ejecuta CARLA Simulator. Posteriormente se reproduce la simulación con todos los datos de todos los sensores de cada vehículo cada 0.3 segundos y se envían a la nube.

5.2 PROCESAMIENTO DE DATOS EN LA NUBE

Se ha elegido realizar el procesamiento de los datos en la nube debido a la problemática que queremos solucionar. Los coches autónomos conectados están en continuo movimiento y es una necesidad poder tener un acceso fiable y flexible desde cualquier sitio. Por otro lado, es imprescindible la escalabilidad que nos ofrece la nube (cloud) debido a que la situación de conducción en el entorno de conducción de una rotonda es muy variante. Dependiendo de la ocupación de la misma (vehículos autónomos conectados que afectan en una rotonda) el número de mansajes que se procesan es muy variante. El mayor reto Big Data en este proyecto es debido a la gran cantidad de volúmenes de datos que se necesitan procesar, independientemente del poco espacio que ocupa cada mensaje. Procesando los datos en la nube (cloud computing) evitamos situaciones de cuello de botella. Suponiendo que hay una media de 30 VAC que afectan en la rotonda, y teniendo en cuenta que los datos de cada VAC se recogen cada 0.3 segundos, el número de mensajes que se han de publicar diariamente en la nube es de:

(3.600 + 3600/3) x 30 vehiculos= 144.000 mensajes diarios.

Por otro lado, los datos se guardan en etapas tempranas de desarrollo del proyecto total de investigación para su posterior uso en caso de ser requerido. Todo esto lo facilitan los sistemas cloud ya que ofrecen una capacidad de almacenamiento ilimitada. El sistema de cloud computing que se utiliza en este proyecto es Google Cloud Platform (GCP)

5.2.1 ARQUITECTURA DEL SISTEMA BIG DATA

Se ha diseñado una arquitectura Big Data con el Sistema distribuido de publicación/suscripción Cloud Pub/Sub de mensajes en tiempo cuasi-real (latencias de 100 milisegundos). [17] Es una herramienta utilizada para realizar ingestas y distribución de datos streaming, alojada en Google Cloud. A continuación, se muestra un esquema de la arquitectura realizada en draw.io:

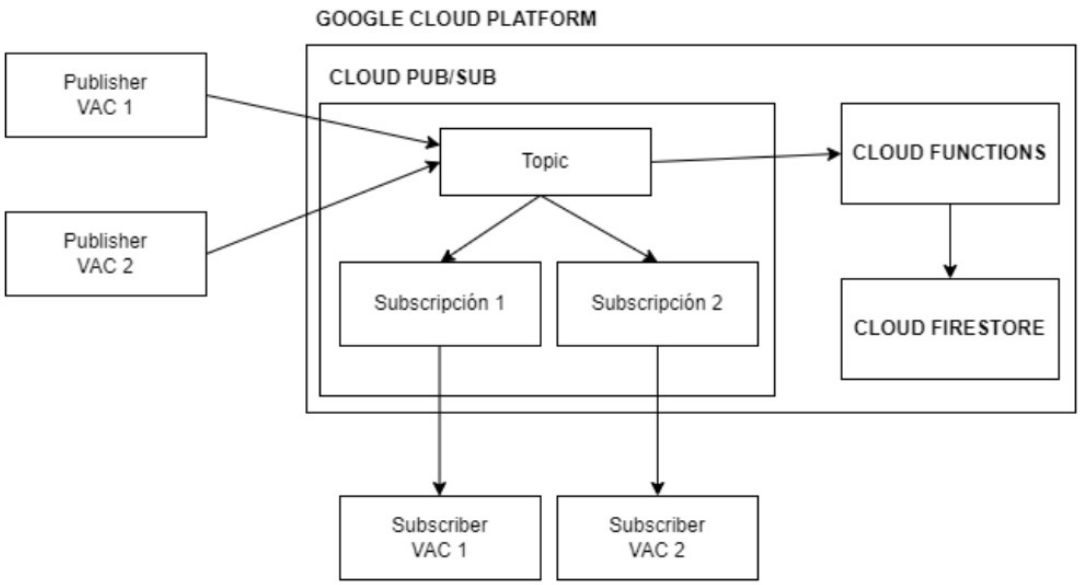


Ilustración 4. Diagrama de la arquitectura Big Data en la nube

La conexión entre vehículos autónomos se describe en los siguientes pasos:

1. Todos los VAC envían datos de sus sensores en tiempo real a la nube ejerciendo la función de Publisher en Cloud Pub/Sub. Todos los mensajes son enviados a un topic concreto creado en Cloud Pub/Sub.
2. Para que un VAC pueda recibir toda la información que se está procesando en la nube se tiene que suscribir al topic especifico que está realizando la ingesta de datos. Para que no exista un rebalanceo de información y lleguen todos los

mensajes cada VAC debe estar suscrito al topic con una suscripción propia. De esta forma cada vehículo autónomo recibe información de todos los demás en tiempo real.

Todos los datos se guardan en etapas tempranas de desarrollo del proyecto total de investigación para su posterior uso en caso de ser requerido. Para ello se ha creado un trigger que se ejecuta cada vez que llegan datos al topic deseado en Cloud Pub/Sub. Una vez el trigger indica que han llegado datos se ejecuta una función creada en Cloud Functions que envía la información a la base de datos noSQL Cloud Firestore. Esta base de datos almacena los datos con formato clave-valor en documentos schemaless que se almacenan en colecciones. Cloud Firestone se adecúa a este problema debido a su baja latencia y su alta escalabilidad. [18]

5.2.2 IMPLEMENTACIÓN DE LA ARQUITECTURA EN LA NUBE

Lo primero que se ha de hacer para poder diseñar una arquitectura cloud en GCP es especificar el nombre del proyecto donde se quiere trabajar. Un proyecto es un contenedor que contine (o te permite habilitar) aplicaciones o servicios disponibles de Google Cloud. [19] Dentro de cada proyecto se administran los recursos que se requieren y pueden tener especificaciones distintas a los demás proyectos de un usuario (servicios habilitados, forma de facturación…). El proyecto que se ha creado para este proyecto se llama “IsabelSutilTFM”. Posteriormente se ha de habilitar la facturación, lo que te da acceso a la plataforma cloud.

5.2.2.1 Conexión entre los vehículos autónomos

Una vez se tiene acceso a los servicios cloud ya se puede implementar la arquitectura previamente diseñada. El servicio más importante en esta arquitectura es Pub/Sub. Es la herramienta que se utiliza para que cada vehículo autónomo publique los mensajes. Todos estos mensajes de todos los vehículos que afectan en una rotonda se recogen en un topic. Cuando cada vehículo autónomo se subscribe al topic donde se están enviando todos los mensajes obtiene todos los datos de todos los coches autónomos en real time consiguiendo una correcta comunicación.

Dentro del proyecto cloud “IsabelSutilTFM” se ha creado un topic en el servicio Pub/Sub llamado “datosVAC” en GCP (https://console.cloud.google.com). Dentro de este topic se identifica cada vehículo autónomo conectado mediante un id que envía junto con los datos de sus sensores. Este topic se ha definido con una encriptación administrada por Google. Dentro de ese topic se ha creado una subscripción al mismo, simulando que es la de un vehículo autónomo que se quiere conectar con los demás. Cada vehículo autónomo necesita su propia subscripción al topic.

La subscripción se ha definido como “datosVAC-sub”, suscrita al topic “datosVAC” con tipo de envío pull. Cuando el subscritor recibe cada dato confirma su llegada a Pub/Sub, al contrario que con el tipo de envío push. El plazo de confirmación desde que los datos se envían y el subscriptor confirma que los ha recibido es de 10 segundos. Una vez pasado este tiempo Pub/Sub no realiza otro intento de envío del mensaje, “Dead lettering”. Esto se debe a que no se quiere que el VAC reciba información obsoleta de la situación de los demás vehículos. En esta subscripción no se ha habilitado que el envío de los datos solo se envíe una vez al subscriptor antes de que el plazo de los 10 segundo ha transcurrido “Exactly once delivery”. Esto se debe a que, en esta problemática, es crucial que se reciban todos los datos real time correctamente. Si un dato se recibe más de una vez (se ha realizado un envío desde Pub/Sub al subscriptor antes de que la confirmación del primer mensaje haya llegado) el subscriptor filtra los datos mediante el timestamp (dato generado por cada publicador que registra el instante de recogida de datos de los sensores).

Para poder llevar a cabo la carga de trabajo de procesamiento se tiene que crear una cuenta de servicio para los VAC dentro de la herramienta de Identity and Access Management (IAM). Con una cuenta de servicio, al contrario que con las cuentas de usuario, los VAC solo tienen acceso a los recursos asignados de Pub/Sub, sin tener la capacidad de ser propietario o administrar los mismos. [20] Las cuentas de usuario están asociadas con un cifrado RSA para poder realizar una correcta autentificación. El cifrado RSA tiene un mecanismo de encriptación asimétrica: dos claves forman parte del proceso de encriptación. Una clave publica que pueden conocer varias cuentas y una clave privada que solo conoce una cuenta específica y no puede compartir. En esta

problemática se encriptan los datos mediante una clave privada que solo conoce cada VAC y se desencripta mediante una clave publica en GCP. De esta forma se garantiza que los datos se han enviado por un VAC, se autentifica el origen. Todos los datos publicados en Pub/Sub son correctos. Al realizar la comunicación entre el topic y el subscriptor se realiza otra vez mediante un cifrado RSA. Los datos se cifran con la clave publica en el topic y se descifran con la clave privada en cada VAC, autentificando el destino. De esta forma se garantiza que solo tienen acceso a los datos de los demás vehículos autónomos otros VAC y que no se va a poder realizar un ataque para conocer la situación de un VAC de forma indeseada. Es una forma de proteger la conducción autónoma de ataques en el proceso de la comunicación, evitando posibles accidentes. Cada cuenta de servicio se crea un el rol de Pub/Sub Administrador ya que cada VAC asume ambos roles de publicador/subscriptor. Una vez creada una cuenta de servicio se ha de crear y descargar una clave privada para poder realizar todo el proceso de autentificación, la key type utilizada en este proyecto tiene la estructura de un JSON (pares clave-valor).

En este proyecto se recogen los datos mediante un simulador (CARLA Simulator) gestionado por una API en Python. Es ahí donde se ha configurado la conexión entre el servidor local en el que se está ejecutando CARLA y la arquitectura Big Data en Google Cloud Platform. Para realizar esta conexión se ha utilizado la librería pubsub_v1 de la biblioteca google.cloud.

Primeramente, se define donde se aloja la clave privada con la que se lleva el proceso de autentificación y se asocia a que es una credencial para poder utilizar los servicios cloud de Google en ambos procesos (publicador y subscriptor).

```
credentials_path = '.\\isabelsutiltfm-privatekey.json'
os.environ['GOOGLE_APPLICATION_CREDENTIALS'] = credentials_path
```

Para publicar mensajes en el topic se ha de definir en el proceso de recolección de datos del VAC un Publisher Client (objeto con la capacidad de publicar en Pub/Sub) y la ruta en cloud del topic a donde se han de enviar los datos de los sensores.

```
publisher = pubsub_v1.PublisherClient()
topic_path = 'projects/isabelsutiltfm/topics/datosVAC'
```

Tras 0.3 segundos recoge información de los sensores del coche, se organiza en formato JSON (entre los que se encuentra el dato timestamp obtenido mediante la librería Datetime) y se publican los datos en el topic realizando la autentificación del origen. Todos los valores del JSON se han definido como string ya que es el único tipo de dato que acepta Pub/Sub. Una vez realizado este proceso se imprime por pantalla un mensaje indicando el id del mensaje publicado. Para publicar mensajes se han de definir en el objeto publicador el topic destino previamente definido, los datos del mensaje y los atributos del mensaje. Como los datos de un mensaje solo pueden ser string se define como datos el identificador de cada coche debido a su importancia y como atributos el JSON que contiene todos los datos recogidos de los sensores.

```
mensaje_id = publisher.publish(topic_path, data, **dict_actor)
print(f'id del mensaje publicado {mensaje_id.result()}')
```

Cuando se ejecuta el proceso se puede ver cómo se van enviando los mensajes.

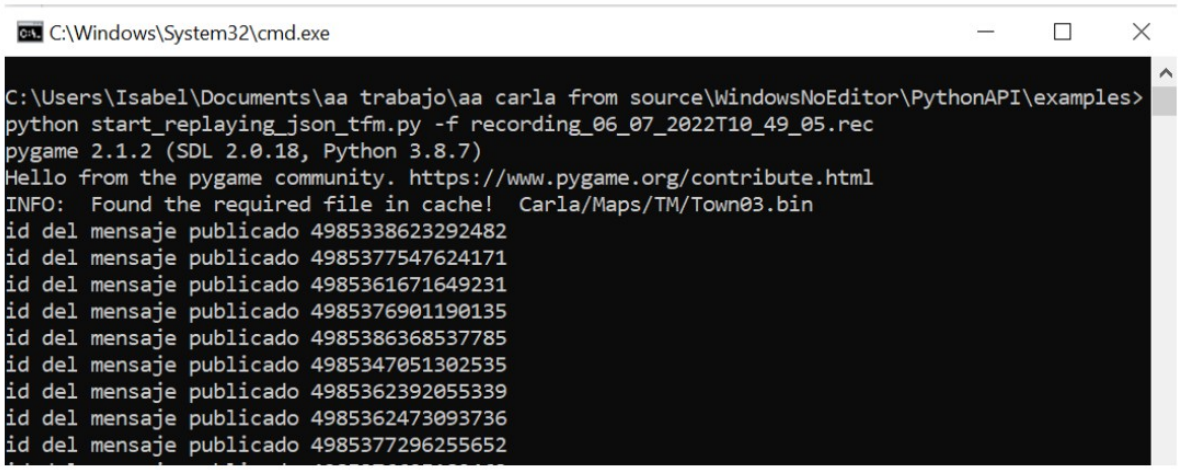


Ilustración 5. Envío de datos a Cloud Pub/Sub

Los datos enviados se pueden visualizar en la interfaz gráfica de Google Cloud desde la herramienta de Pub/Sub.

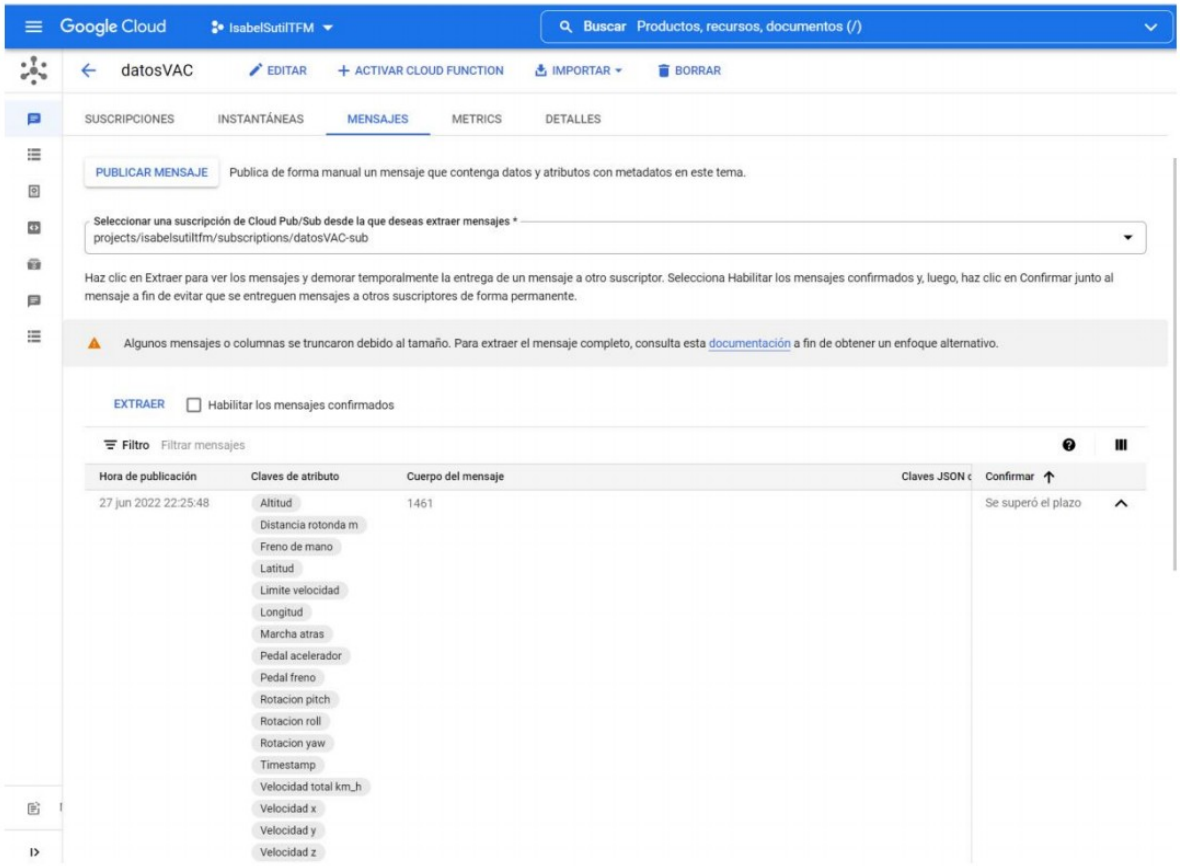


Ilustración 6. Datos publicados en Cloud Pub/Sub

Para recibir mensajes en el topic, ejerciendo de subscriptor se ha de definir otro proceso dentro del VAC. En este proceso, al igual que en el publicador, se definen la clave privada del VAC para poder realizar la autentificación y se define el path del topic en la nube. A diferencia se ha de declarar un objeto de tipo subscriptor para poder extraer los mensajes con los datos de todos los VAC.

```
subscriber = pubsub_v1.SubscriberClient()
```

Se define una función llamada callback que imprime por pantalla el mensaje recibido (id del VAC que ha publicado esos datos) junto con sus atributos (datos de los sensores del VAC en un instante).

```
def callback(message):
    print(f'Mensaje recibido: {message}')
    print(f'Datos recibidos: {message.data}')

    if message.attributes:
        print('Attributes: ')
        for key in message.attributes:
            value = message.attributes.get(key)
            print(f'{key} : {value}')
```

Para ejecutar el proceso se utiliza el objeto suscriptor previamente definido para que reciba datos del topic “DatosVAC”, cada vez que se reciben datos se llama a la función callback para que se impriman por pantalla.

```
streaming_pull_mensaje_id = subscriber.subscribe(subscription_path,
callback=callback)
print(f'Escuchando del path: {subscription_path}')

with subscriber:
    try:
        streaming_pull_mensaje_id.result()
```

Al ejecutar el proceso obtenemos la información de todos los VAC.

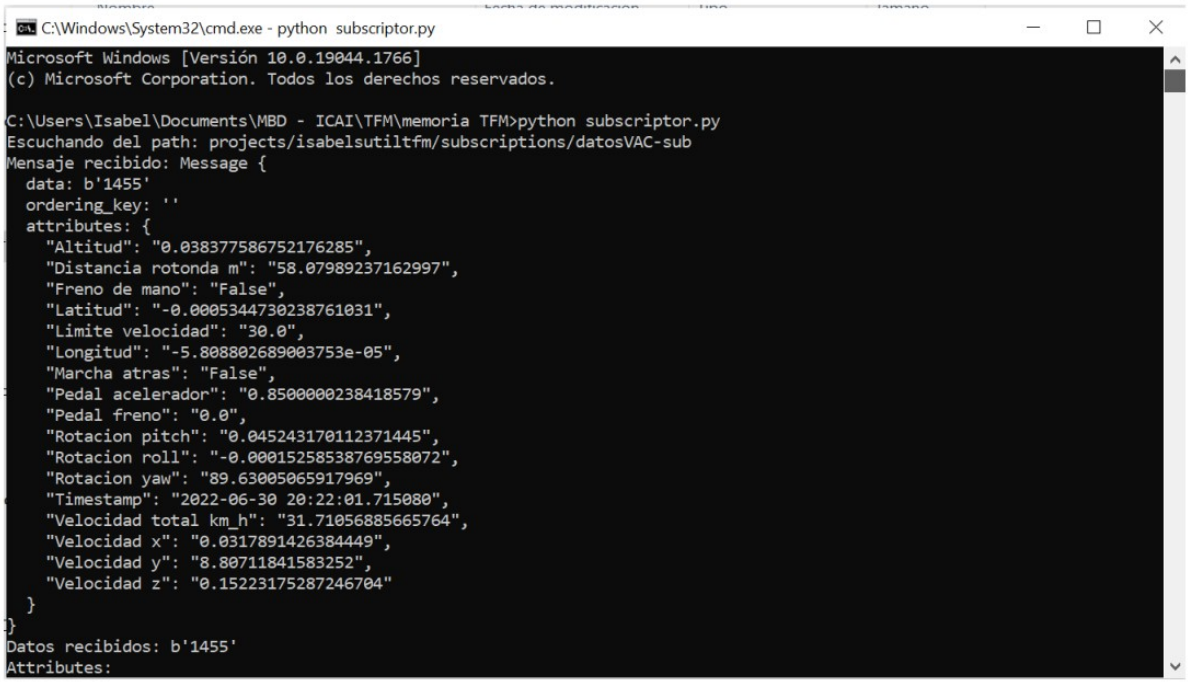


Ilustración 7. Datos recogidos de Cloud Pub/Sub

5.2.2.2 Almacenamiento de los datos

Para almacenar los datos se ha implementado una función en Cloud Functions que se ejecuta por un trigger cada vez que llega un mensaje al topic DatosVAC con los datos de los sensores y los envía y guarda a la base de datos Firestore.

Sin embargo, antes de implementar la función, se ha de habilitar la base de datos Firestore en GCP para que se puedan almacenar los datos y no existen errores de ejecución por este motivo. Esta base de datos se utiliza en modo nativo para poder utilizar todas las características real time de la herramienta de desarrollo de aplicaciones de la que forma parte: Firebase Realtime Database. [21] Posteriormente se ha de habilitar el proyecto que estamos utilizando en Google Cloud Platform en la consola de Firebase (https://console.firebase.google.com/u/0/).

Para crear una función con Cloud Functions se han de tener habilitadas las API de: Cloud Build API, Cloud Functions Api, Cloud Logging API y Cloud Pub/Sub API.

La función implementada se llama “dataVAC-fiestore”, está ubicada en la región “europee-west1” y se activa cuando llegan mensajes al topic “datosVAC”. El entorno de ejecución en el que se ha implementado es Python 3.8. Los requerimientos necesarios para que pueda funcionar se han definido en el archivo requirements.txt

```
# Function dependencies
google-cloud-logging
grpcio
google-cloud-firestore
google-cloud
```

La función implementada se ha definido en el archivo main.py. Cada vez que llega un mensaje al topic se lanza un trigger que activa la función envio_a_firestore(event, context), con los parámetros del mensaje (event) y los metadatos del mensaje (context). Primeramente, se recoge el id del VAC (cuerpo del mensaje) en la variable pubsub_message_id y los atributos que contienen la información de los sensores.

Se define la estructura de la base de datos: Dentro de la colección llamada dispositivos se almacena un documento por cada VAC. Dentro del documento de cada VAC se almacenan los datos en documentos en la colección “datosDelVAC”. No hace falta definir la estructura de la base de datos previamente, se va generando real time al llegar los datos.

```
import base64
import json
import logging

from google.cloud import firestore
db = firestore.Client()

def envio_a_firestore(event, context):

    logging.info('Mensaje con ID: ' + str(event))

    pubsub_message_id = base64.b64decode(event['data']).decode('utf-8')

    atributos = event['attributes']

    _, doc_ref = db.collection('dispositivos/{}/datosDelVAC'.format(pubsub_messag
e_id)).add({
      'timestamp' : atributos['Timestamp'],
      'pedal_acelerador' : atributos['Pedal acelerador'],
      'pedal_freno' : atributos['Pedal freno'],
      'freno_de_mano' : atributos['Freno de mano'],
      'marcha_atras' : atributos['Marcha atras'],
      'limite_velocidad' : atributos['Limite velocidad'],
      'latitud' : atributos['Latitud'],
      'longitud' : atributos['Longitud'],
      'altitud' : atributos['Altitud'],
      'rotacion_roll' : atributos['Rotacion roll'],
      'rotacion_pitch' : atributos['Rotacion pitch'],
      'rotacion_yaw' : atributos['Rotacion yaw'],
      'distancia_rotonda_m' : atributos['Distancia rotonda m'],
      'velocidad_x' : atributos['Velocidad x'],
      'velocidad_y' : atributos['Velocidad y'],
      'velocidad_z' : atributos['Velocidad z'],
      'velocidad_total_km_h' : atributos['Velocidad total km_h']
    })

    logging.info('Mensaje con ID: ' + doc_ref.id)
```

Los datos insertados se pueden visualizar en la interfaz gráfica de Firestore dentro de GCP. Cada documento (datos cada 0.3 segundos de cada vehículo) ocupa 0.007 GB.

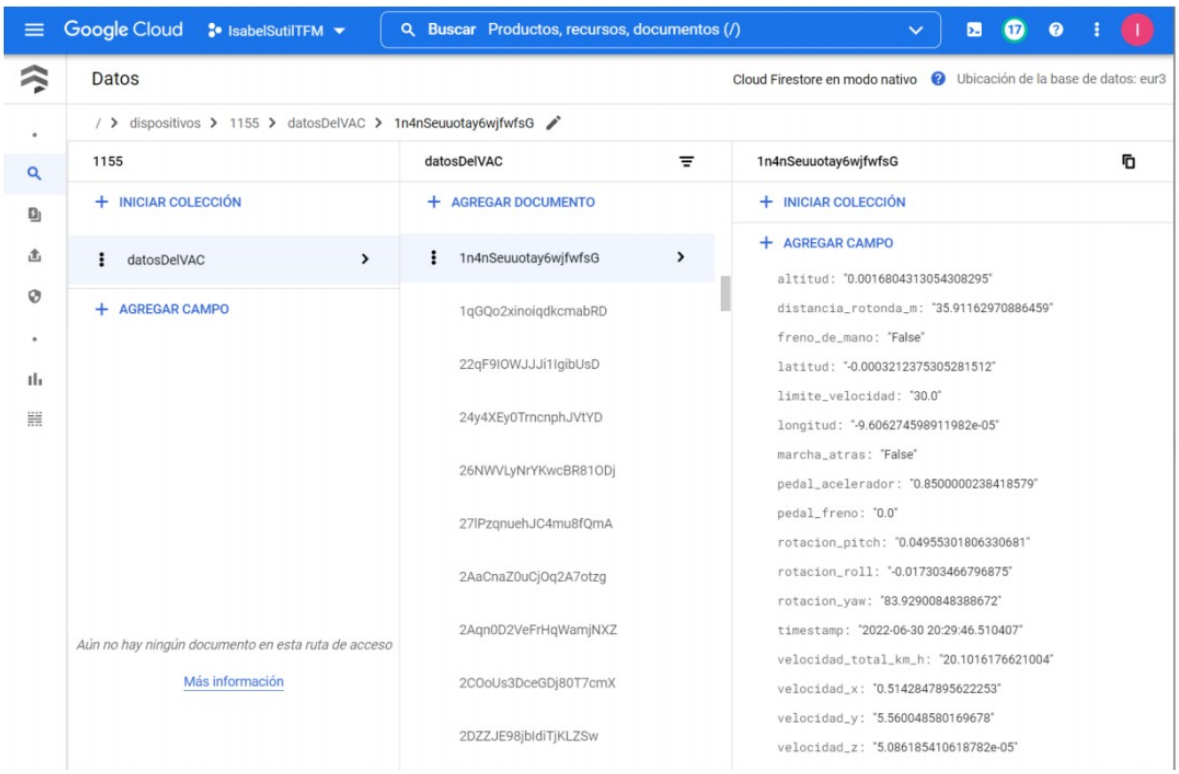


Ilustración 8. Datos almacenados en Cloud Firestore

Capítulo 6. PRESUPUESTO

6.1 CALCULO ANUAL DE LOS DÍAS EFECTIVOS DE TRABAJO

A continuación, se adjunta una tabla que describe el número de días efectivos utilizados para la realización del proyecto y el número de horas empleadas.

Duración del proyecto: 9 meses (noviembre 2021 – julio 2022)	273
Sábados y Domingos: (273 x 2/7 = 68.25)	- 62
Días efectivos de vacaciones	- 14
Días festivos reconocidos	- 5
Varios	-5
Total estimado de días efectivos	187
Total horas/proyecto efectivas (Calculadas para 4 horas de trabajo/día)	748

Tabla 1. días y horas efectivas de trabajo en el proyecto

El proyecto se ha desarrollado por un equipo conformado por un Data Architect y por un Data Engineer. A continuación, se detalla el computo proporcional de horas efectivas de cada empleado.

Data Architect	4/9 Total horas/proyecto efectivas	332
Data Engineer	5/9 Total horas/proyecto efectivas	416

Tabla 2. Horas efectivas por perfil técnico

6.2 COSTO DEL PERSONAL

A continuación, se adjunta una tabla que describe el costo total del personal, teniendo en cuenta los dos perfiles que han participado en las horas empleadas para este proyecto.

	Data Architect	Data Engineer
Sueldo medio anual bruto	57.478 €	35.467 €
Prestaciones seguridad social (33,95% del sueldo bruto)	19.513,78 €	12.041,04 €
Coste total anual	76.991,78 €	47.508,04 €
Coste de una hora (jornada laboral anual = 1.776 horas)	43.35 €	26,75 €
Coste proporcional horas/proyecto	14.392,2 €	11.128 €
Coste total del personal		25.520,2 €

Tabla 3. Costo del personal

6.3 COSTO DEL MATERIAL

A continuación, se adjunta una tabla que describe el costo del material utilizado en el proyecto en el servidor local donde se ejecuta el simulador de conducción.

Procesador	AMD Ryzen ThreadRipper Pro 3975WX 3.5 GHz	2.679,25 €
Placa Base	Asus PRO WS WRX80E SAGE SE WIFI	865,95 €
Memoria RAM	8 x Kingston HyperX Fury Black 16GB DDR4 3200Mhz PC 25600 CL16	720 €
Disipador	iCUE H150i RGB PRO XT Liquid CPU Cooler	134,88 €
Disco duro	Gigabyte AORUS NVMe Gen4 SSD 1TB M.2 3D TLC	201,6 €
Caja	Fractal Design Define 7 XL USB 3.1 Negra	385,03 €
Fuente de alimentación	Gigabyte GP P850GM 850W 80 Plus Gold Modular	129,99 €
Tarjeta gráfica	Gigabyte AORUS GeForce RTX 3080 Ti MASTER 12GB GDDR6X	1.699,89 €
Tarjeta de red	Intel X550-T2. Tarjeta de red 2x RJ45 10Gb PCIe	292,99 €
Volante de conducción	Logitech G29 Driving Force Xbox-PC	399 €
Total		7.508,58 €

Tabla 4. Costo del material

6.4 *COSTO DE LOS SERVICIOS EN LA NUBE*

A continuación, se describe el número de operaciones de escritura en Cloud Firestore [22] suponiendo que hay una media de 30 VAC que afectan a la rotonda. Número de operaciones de lectura/escritura de documentos durante las 4 semanas en las que se ha estado utilizando Cloud Firestore, teniendo en cuenta que los datos se recogen cada 0.3 segundos. La región elegida de alojamiento de Cloud Firestore es Europa (varias regiones)

Documentos totales al día = (3.600 + 3600/3) x 30 vehículos= 144.000 documentos diarios.

Teniendo en cuenta que los primeros 50.000 documentos de lectura diarios son gratuitos junto con los 20.000 primeros de escritura el número de documentos diarios que necesitan un coste son:

- 94.000 documentos de lectura
- 124.000 documentos de escritura

Estos datos se han almacenado en Cloud Firestore durante las 4 semanas. Cada documento ocupa 0.007 GB.

3.225.600 documentos x 0.007 GB = 22.579,2 GB

Si tenemos en cuenta que el primer GB de almacenamiento es gratuito el costo del almacenamiento es lo que corresponda a 21.579,2 GB.

Coste total de los servicios contratados en Google Cloud Platform:

Operaciones de lectura (0.057 €/ documento)	535,8 €
Operaciones de escritura (0.17 €/ documento)	21.080 €
Almacenamiento (0.17 €/ GB)	3.668,46 €
Total	25.284,26 €

Tabla 5. Costo de los servicios en la nube

6.5 COSTO TOTAL

Coste total del personal	25.520,2 €
Coste total del material local	7.508,58 €
Coste total de los servicios en la nube	25.284,26 €
Suma de los costes	58.313,04 €
Beneficio del proyecto (15%)	8.742,96 €
TOTAL	67.056 €

Tabla 6. Costo total del proyecto

Capítulo 7. RESULTADOS DEL PROYECTO

En este proyecto se ha conseguido diseñar e implementar correctamente un sistema Big Data que permite la conexión entre vehículos autónomos en un entorno simulado.

Se recogen datos de los sensores de cada vehículo autónomo conectado en el simulador de conducción CARLA Simulator cada 0.3 segundos. Todo ello se ha realizado en un entorno de conducción complejo de forma simulada: las rotondas.

Cada vehículo recoge y publica sus datos en un topic dentro del servicio de Pub/Sub de Google Cloud Platform de forma segura mediante una autentificación con cifrado RSA (clave publica/privada) por lo que solo se almacenan datos reales y confiables. La conexión se realiza mediante subscripciones al topic por parte de cada vehículo autónomo, el cual recibe datos relevante a la rotonda de todos los vehículos que afectan en la misma de forma autenticada. Esto es imprescindible para el vehículo autónomo pueda tomar decisiones de conducción correctas confiando que los datos son reales y fiables. Mediante este sistema se ha conseguido un procesamiento de los datos en la nube en tiempo real de forma automática.

Los datos de todos los vehículos autónomos conectados se almacenan correctamente y en tiempo real en una base de datos NoSQL orientada a documentos en la nube: Cloud Firestore.

Capítulo 8. CONCLUSIONES

Mediante el sistema Big Data diseñado e implementado se consigue una conectividad segura y en tiempo real entre los vehículos autónomos.

Mediante las tecnologías y servicios que se ofrecen en Google Cloud Platform podemos procesar ingestas de grandes volúmenes de datos que no se podrían realizar con las tecnologías tradicionales (144.000 mensajes diarios publicados suponiendo que afectan 30 vehículos de media en una rotonda).

El valor que este proyecto aporta es la posibilidad de llegar a conocer mejor el entorno de conducción mediante los datos de los sensores de otros vehículos. Esto aporta una visión más global del sistema debido a que los datos de los sensores están limitados a su entorno cercano.

Esto es imprescindible para que los coches lleguen a tener una autonomía completa que aún no se ha podido alcanzar, especialmente en los entornos complejos de conducción.

Gracias a los avances en el área de investigación de los vehículos autónomos conectados se aspira a:

- Tener una mayor seguridad en el vehículo.
- Reducir el número de accidentes en rotondas.
- Reducir el número de heridos o muertos en las carreteras.

Capítulo 9. FUTURAS LÍNEAS DE TRABAJO

La principal futura línea de trabajo es la implementación en un entorno real el sistema de comunicación entre vehículos autónomos diseñado e implementado en un entorno virtual. Se ha de comprobar que la conexión se realiza correctamente en todo momento con la nube. La comunicación entre los vehículos Autónomos Conectados se puede hacer de manera correcta y segura. Los datos llegan íntegros a la nube y a los vehículos subscritos que quieren recibir la información. No existen cuellos de botella en la nube debido a la gran cantidad de datos que se publican y se recogen en las subscripciones.

En el caso de que el proceso de comunicación entre los vehículos autónomos no funcione correctamente o tenga latencias (no ocurra en tiempo real) el diseño se ha de mejorar y volver a implementar.

Actualmente existen entidades que están invirtiendo en área del análisis de datos para la aplicación en vehículos autónomos no conectados. Otra línea de trabajo futura es el uso de técnicas de Machine Learning o de Deep Learning para reconocer patrones que puedan ayudar a establecer reglas de conducción autónoma con la información obtenida de los demás Vehículos Autónomos Conectados. El valor que aportan los datos de los sensores de todos los vehículos que afectan en una situación compleja de conducción puede suponer para un algoritmo la toma correcta de la decisión.

Capítulo 10. BIBLIOGRAFÍA

[1] **Agencia Estatal de Investigación. 2021**. "Proyecto I+D+i «Prueba de Concepto» 2021: Sistema para la automatización de vehículos de transporte público y compartido destinados a entornos semiestructurados." [En línea]. Disponible en: https://www.aei.gob.es/ayudas-concedidas/ayudas-destacadas/proyecto-idi-prueba-concepto-2021-sistema-automatizacion [consulta: Junio de 2022]

[2] **Agencia Estatal Boletín Oficial del Estado. 2022.** "Real Decreto Legislativo 8/2004, de 29 de octubre, por el que se aprueba el texto refundido de la Ley sobre responsabilidad civil y seguro en la circulación de vehículos a motor." [En línea]. Disponible en: https://www.boe.es/buscar/act.php?id=BOE-A-2004-18911 [consulta: Junio de 2022]

[3] **Sociedad de Ingenieros de Automoción. 2021.** "Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles" [En línea]. Disponible en: https://www.sae.org/standards/content/j3016_202104/ [consulta: Junio de 2022]

[4] **Sociedad de Ingenieros de Automoción. 2021.** "SAE Levels of Driving Automation™ Refined for Clarity and International Audience" [En línea]. Disponible en: https://www.sae.org/blog/sae-j3016-update [consulta: Junio de 2022]

[5] **European Commission. 2019.** "Road safety: Commission welcomes agreement on new EU rules to help save lives" [En línea]. Disponible en: https://ec.europa.eu/commission/presscorner/detail/en/IP_19_1793 [consulta: Junio de 2022]

[6] **Defense Advanced Research Projects Agency. 2004.** "The Grand Challenge" [En línea]. Disponible en: https://www.darpa.mil/about-us/timeline/-grand-challenge-for-autonomous-vehicles [consulta: Junio de 2022]

[7] **Instituto Universitario de Investigación del Automovil. 2020.** "«Sistema de arbitraje distribuido para conducción cooperativa conectada y autónoma en entornos complejos. Servicios cooperativos y conciencia situacional» (CCAD)" [En línea]. Disponible en: https://insia-upm.es/project/ccad/ [consulta: Junio de 2022]

[8] **SEGVAUTO 4.0-CM. 2020.** "Objetivos y Líneas de Investigación" [En línea]. Disponible en: https://segvauto.es/ [consulta: Junio de 2022]

[9] **Comisión Económica de la Naciones Unidas para Europa. 2022.** "UN Regulation extends automated driving up to 130 km/h in certain conditions" [En línea]. Disponible en: https://unece.org/sustainable-development/press/un-regulation-extend-automated-driving-130-kmh-certain-conditions [consulta: Junio de 2022]

[10] **Comisión Económica de la Naciones Unidas para Europa. 2022.** "Proposal for the 01 series of amendments to UN Regulation No. 157 (Automated Lane Keeping Systems)" [En línea]. Disponible en: https://unece.org/sites/default/files/2022-05/ECE-TRANS-WP.29-2022-59r1e.pdf [consulta: Junio de 2022]

[11] **Dirección General de Tráfico. 2022.** "Glorietas: Las imprudencias más peligrosas" [En línea]. Disponible en: https://www.dgt.es/comunicacion/noticias/glorietas-las-imprudencias-mas-peligrosas/ [consulta: Junio de 2022]

[12] **Kent Beck; James Grenning; Robert C. Martin; Mike Beedle; Jim Highsmith; Steve Mellor; Arie van Bennekum; Andrew Hunt; Ken Schwaber; Alistair Cockburn; Ron Jeffries; Jeff Sutherland; Ward Cunningham; Jon Kern; Dave Thomas; Martin Fowler; Brian Marick. 2001.** "The Agile Manifesto" [En línea] Disponible en: http://agilemanifesto.org/ [consulta: Junio de 2022]

[13] **Ken Schwaber; Jeff Sutherland. 2020.** "La Guía Definitiva de Scrum: Las Reglas del Juego" [En línea] Disponible en: https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-Spanish-European.pdf [consulta: Junio de 2022]

[14] **Alexey Dosovitskiy; German Ros; Felipe Codevilla; Antonio López; Vladlen Koltun; Intel Labs; Toyota Research Institute; Computer Vision Center, Barcelona. 2017.** "CARLA: An Open Urban Driving Simulator" [En línea] Disponible en: http://proceedings.mlr.press/v78/dosovitskiy17a/dosovitskiy17a.pdf [consulta: Junio de 2022]

[15] **CARLA Simulator. 2017** "Open-source simulator for autonomous driving research" [En línea] Disponible en: https://carla.org/ [consulta: Junio de 2022]

[16] **CARLA Simulator. 2017** "Windows build" [En línea] Disponible en: https://carla.readthedocs.io/en/latest/build_windows/ [consulta: Junio de 2022]

[17] **Google Cloud Documentation. 2022.** "What is Pub/Sub?" [En línea] Disponible en: https://cloud.google.com/pubsub/docs/overview?hl=en [consulta: Junio de 2022]

[18] **Firebase Documentation. 2022.** "Cloud Firestore" [En línea] Disponible en: https://firebase.google.com/docs/firestore [consulta: Junio de 2022]

[19] **Google Cloud Documentation. 2022.** "Managing Google Cloud Projects, App Engine Applications, and Billing" [En línea] Disponible en: https://cloud.google.com/appengine/docs/flexible/dotnet/managing-projects-apps-billing?hl=es-419 [consulta: Junio de 2022]

[20] **Google Cloud Documentation. 2022.** "Service Accounts" [En línea] Disponible en: https://cloud.google.com/iam/docs/service-accounts [consulta: Junio de 2022]

[21] **Google Cloud Documentation. 2022.** "Choosing between Native mode and Datastore mode" [En línea] Disponible en: https://cloud.google.com/datastore/docs/firestore-or-datastore [consulta: Junio de 2022]

[22] **Google Cloud Documentation. 2022.** "Firestore Pricing" [En línea] Disponible en: https://cloud.google.com/firestore/pricing [consulta: Junio de 2022]