Transport, Mobility & Society. 2025; 4:210
doi: 10.56294/tms2025210

REVIEW

Integration of Alexa Auto and vehicle diagnostic protocols: towards a smart automotive ecosystem

Integración de Alexa Auto y protocolos de diagnóstico vehicular: hacia un ecosistema automotriz inteligente

Alexis Javier Villacis Ninasunta[1], Mariana Pinargote Basurto[1]

[1]Universidad Técnica Estatal de Quevedo, Facultad de Ingeniería, Departamento de Ingeniería Mecánica. Ecuador.

Cite as: Villacis Ninasunta AJ, Pinargote Basurto M. Integration of Alexa Auto and vehicle diagnostic protocols: towards a smart automotive ecosystem. Transport, Mobility & Society. 2025; 4:210. https://doi.org/10.56294/tms2025210

Submitted: 08-08-2024 **Revised:** 03-11-2024 **Accepted:** 23-07-2025 **Published:** 24-07-2025

Editor: Prof. Emanuel Maldonado

ABSTRACT

Introduction: the study described how the advancement of digital technologies transformed the automotive industry and drove the adoption of in-cabin voice assistants. It presented Alexa and the Echo Auto device as versatile alternatives to factory solutions, thanks to their synchronization with phones and personalization through skills. He argued that, by integrating with diagnostic standards such as OBDII and the CAN network, such assistants offered immediate access to critical vehicle parameters, reduced driver distraction and facilitated more natural interaction for entertainment, communication and control.
Development: the work reviewed the state of the art and reports that associated the assistants with less visual and manual load and better reaction times. It described integrated ecosystems (MBUX, CarPlay, Android Auto) and external alternatives such as Echo Auto and compatible devices. He detailed the architecture of Alexa and its voice processing pipeline, from audio capture to response synthesis, and presented the development of skills by blocks with Voiceflow and by textual programming through Alexa Skills Kit. He explained the role of IoT platforms, with emphasis on Blynk, to orchestrate data and control devices. Characterized OBDII - connector, protocols and interface - and CAN protocol - topology, speeds, frames and termination requirements - as a basis for extracting and transmitting reliable vehicle data to cloud services and assistance applications.
Conclusions: the research argued that linking Alexa Auto to the in-vehicle computer was feasible and strategic. The convergence between voice assistants, IoT, OBDII and CAN enabled timely diagnostics, hands-free interaction and informed decisions. It was concluded that this integration elevated road safety, optimized car lifespan, and paved the way for smarter vehicle ecosystems and, going forward, an orderly transition to autonomous and sustainable driving.

Keywords: Alexa Auto; OBDII; CAN; IoT; Voice Assistants.

RESUMEN

Introducción: el estudio describió cómo el avance de las tecnologías digitales transformó la industria automotriz e impulsó la adopción de asistentes de voz en cabina. Presentó a Alexa y al dispositivo Echo Auto como alternativas versátiles frente a soluciones de fábrica, gracias a su sincronización con teléfonos y a la personalización mediante skills. Argumentó que, al integrarse con estándares de diagnóstico como OBDII y la red CAN, dichos asistentes ofrecieron acceso inmediato a parámetros críticos del vehículo, disminuyeron la distracción del conductor y facilitaron una interacción más natural para entretenimiento, comunicación y control.
Desarrollo: el trabajo revisó el estado del arte y reportes que asociaron los asistentes con menor carga visual y manual y con mejores tiempos de reacción. Describió ecosistemas integrados (MBUX, CarPlay, Android Auto) y alternativas externas como Echo Auto y dispositivos compatibles. Detalló la arquitectura de Alexa y su canal de procesamiento de voz, desde la captura del audio hasta la síntesis de respuesta, y expuso el

desarrollo de skills por bloques con Voiceflow y por programación textual mediante Alexa Skills Kit. Explicó el papel de plataformas IoT, con énfasis en Blynk, para orquestar datos y controlar dispositivos. Caracterizó OBDII —conector, protocolos e interfaz— y el protocolo CAN —topología, velocidades, tramas y requisitos de terminación— como base para extraer y transmitir datos confiables del vehículo hacia servicios en la nube y aplicaciones de asistencia.
Conclusiones: la investigación sostuvo que enlazar Alexa Auto con el computador del vehículo fue factible y estratégico. La convergencia entre asistentes de voz, IoT, OBDII y CAN habilitó diagnósticos oportunos, interacción manos libres y decisiones informadas. Se concluyó que esta integración elevó la seguridad vial, optimizó la vida útil del automóvil y preparó el camino para ecosistemas vehiculares más inteligentes y, a futuro, para una transición ordenada hacia la conducción autónoma y sostenible.

Palabras clave: Alexa Auto; OBDII; CAN; IoT; Asistentes de Voz.

INTRODUCTION

The rapid advancement of digital technologies has significantly transformed the automotive industry, driving the incorporation of intelligent systems focused on both safety and user comfort.[1,2] Among these innovations, voice assistants stand out, whose adoption has grown exponentially in recent years, not only in the home but also in vehicles. These tools have become a key element in the interaction between the driver and the car's information system, reducing visual and manual distraction and allowing more natural and efficient access to entertainment, communication, and vehicle control functions.[3,4,5]

In this context, Amazon has developed the Alexa virtual assistant and the Echo Auto device, specifically designed to offer an optimized in-car experience. These technologies are presented as a versatile alternative to other factory-integrated voice assistants, allowing synchronization with mobile devices and customization through "skills" that expand the range of available services.[6,7] The potential of these innovations is not limited to entertainment or assistance with basic tasks, but also extends to integration with vehicle diagnostic systems, opening up a field of application focused on safety and preventive maintenance.[8,9]

Furthermore, the introduction of the OBDII (On-Board Diagnostics II) system and the CAN (Controller Area Network) protocol have laid the foundation for monitoring and controlling key vehicle parameters.[10,11] These standards allow real-time information to be extracted about engine performance, emissions, and other critical systems, providing essential data to prevent failures, improve efficiency, and comply with environmental regulations.[12,13] The possibility of linking this data with voice assistants represents an opportunity to provide drivers with immediate and accurate information in an accessible and non-invasive way.[14,15,16]

The design of a system to link Alexa Auto and the vehicle's computer is therefore a technological challenge with great potential impact.[17,18] Its implementation requires an understanding of the state of the art in voice assistants, the programming and customization of skills, the use of IoT platforms for device management, and mastery of automotive communication protocols.[19,20,21] By integrating these dimensions, an ecosystem is proposed that can improve the user experience, increase road safety, and optimize the vehicle's useful life through timely diagnostics and personalized assistance.[22,23,24]

In short, this research aims to provide a theoretical basis for the development of such a system, exploring current trends and the technological tools necessary for its implementation, with the goal of offering an innovative solution that combines artificial intelligence, IoT connectivity, and automotive diagnostic standards for the benefit of the modern driver.[25,26,27]

DEVELOPMENT

Introduction

This chapter details the theoretical basis for the design and implementation of the link system between the vehicle's computer and Alexa Auto, highlighting first the state of the art in voice assistants and diagnostic systems in the automotive industry.[28,29,30] It also covers topics related to the Echo Auto voice assistant, the development of skills, and platforms for managing IoT devices.[31,32] These are the tools necessary for the development of the system.[33]

It also addresses topics related to the OBDII system, such as interfaces and protocols for communication, and the controller area network system known as CAN, which is the basis for achieving proper communication with the data bus under the defined protocol, and thus extracting or sending the necessary information.[34]

State of the Art

The potential of the automotive industry is currently the main factor driving the growth of artificial intelligence applied in this field, mainly due to customer preference for new and advanced features such as driver assistance, autonomous driving, personalized entertainment, etc. This is why voice assistants are becoming a growing trend in vehicles. In addition to minimizing driver distraction during manual driving, they offer an experience close to the transition to autonomous driving. The current landscape of commercial voice assistants for vehicles includes Google Assistant via Android Auto, Apple's Siri via CarPlay, Mercedes-Benz's MBUX, Amazon Alexa's Echo Auto, and many more.[35]

According to Smith S.[1], as voice assistant technology becomes more popular in automotive systems, the need to optimize these systems becomes crucial for a sophisticated in-vehicle driving experience, as research indicates that voice assistant usage will triple by 2023. Within the automotive field, 57,6 % of respondents in the 2019 Voicebot report on the adoption of voice assistants in vehicles stated that within a year they would use voice assistants while in a vehicle more frequently than they had up to that point. In addition, 19,1 % of consumers already say that integrated voice assistance is an important consideration or even a requirement when purchasing a vehicle. Similarly, studies have shown that replacing traditional screens with smart voice assistants results in improved driving performance, less time off the road, less lane deviation, and better reaction times to accidents.[36]

Below are several applications in the automotive field related to voice assistants and diagnostic systems. Monitoring vehicle systems seeks to improve the vehicle's useful life by alerting the user to possible faults, while voice assistants focus on providing greater comfort to users.[37]

Voice assistants and diagnostic systems in vehicles

In the context of improving the user experience[2] uses information provided by IoT platforms to offer timely solutions in the search for parking spaces for vehicles using an intelligent voice assistant. The author has developed a skill for Alexa that connects to the FIWARE platform, which collects environmental data provided by users, such as sensors and even mobile applications. This allows users to make a request to the virtual assistant, which generates a response that guides them to an available parking space.[38]

The work carried out by Caicedo et al.[3] seeks to identify faults or errors that could jeopardize the vehicle's operation in a timely manner. The aim is to preserve the vehicle's useful life and keep the driver informed of the vehicle's status in real time.[39] This information is presented in web interfaces, where the user can review it in greater detail. In terms of the hardware used, this system requires the ELM327 diagnostic module to be connected to the vehicle's OBDII port, and the collected data is sent via Bluetooth to an ESP32 microcontroller, which is responsible for decoding it. In turn, the ESP32 is connected to a computer (LattePanda), where the data is sent to the cloud so that it can be monitored in a more detailed interface.[40]

Voice assistants for vehicles

There are two main alternatives for voice assistants in vehicles: voice assistants integrated into the vehicle during manufacture and voice assistants with independent devices. Depending on which one is available in the vehicle, the user can access different features related to entertainment and multimedia or even vehicle control.[41]

The factory-integrated options may be the vehicle brands' own systems, such as Mercedes-Benz with its MBUX assistant, BMW with BMW Intelligent Personal Assistant, or Audi with its MMI Touch Response operating system. Another integration option available is due to the fact that a wide variety of manufacturers have entered into agreements with Amazon, Google, or Apple to install Android or iOS-compatible systems and use a smartphone to sync with their infotainment center. The main advantage of these systems is that they allow access to additional vehicle control functions. These include engine start, temperature control, door locks, and others that seek to make the driver's stay more comfortable.[42]

Apple's CarPlay operating system allows the iPhone screen to be displayed on the vehicle's digital dashboard screen. This allows the user to access the phone's functions much more safely and without having to take their hands off the wheel, and even interact with the Siri voice assistant. Currently, a wide variety of vehicles have included or are compatible with this system. The models and brands can be checked on Apple's official website.[4]

On the other hand, Google has Android Auto, a mobile application that allows an Android phone to be connected to the vehicle's screen. This allows the user to access their smartphone applications via voice commands. The latest models from major manufacturers, including Ford, Nissan, Toyota, Mazda, Hyundai, etc., are compatible with Android Auto.[5]

Amazon has entered into various agreements with several brands to include Alexa in car manufacturing, allowing users to control their vehicles from anywhere they want or from any Echo device. Currently, a wide variety of brands and models feature this technology, which can be verified on their official website, where a specific skill is even available for each one.[6]

For vehicles that do not have this technology integrated, there are mobile app options such as AutoMate that allow interaction with the smartphone via voice commands. However, their functions are limited to navigation, music, phone, and messaging apps. Other options that require physical devices such as JBL Link Drive or Anker Roav Bolt allow users to connect to Google Assistant and its features. Alternatively, they can use an Echo Auto device to interact with Amazon's Alexa voice assistant.[7]

https://doi.org/10.56294/tms2025210 ISSN: 3072-8398

Alexa voice assistant and Echo Auto

Alexa is a virtual assistant developed by Amazon, which is used in devices such as Echo smart speakers. Its main function is to provide a voice-based user interface so that people can interact with other devices or services. Among the most common functions for this assistant are controlling smart devices, accessing training services, general information through Internet searches, creating lists or reminders, calling and messaging services, as well as personalized skills.[43]

Each time the user makes a request, a series of processes are carried out that allow the voice assistant to interpret the message given by the user and provide an appropriate response. Figure 1 shows a 7-step process that describes the processing flow that takes place:

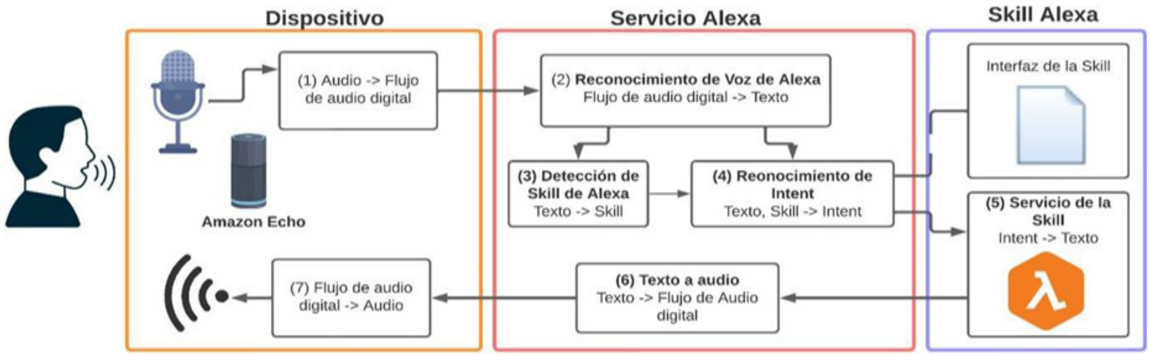

Figure 1. Alexa processing channel

- Audio conversion: the physical device captures the audio, digitizes it, encrypts it, and sends it as a digital stream to the Alexa processing cloud.
- Voice recognition: this allows the digital audio stream to be transformed into a text string, using Speech to Text services.
- Skill detection: the recognized text allows Alexa to determine the corresponding application to activate.
- Intent recognition: Alexa uses the intent recognition model of the respective skill and extracts the user's request along with other important parameters from the text.
- Skill services: to generate an appropriate response, Alexa accesses services that receive the information previously extracted from the text to generate textual responses to the user's request.
- Text-to-audio conversion: Alexa's Text-to-Speech services are used to generate a digital audio stream of the textual response and send it to the Echo device.
- Audio stream conversion: the Echo device plays the generated audio to provide an audible response to the user's request.

Alexa's Echo devices are a variety of voice-controlled smart speakers. These may include the Alexa voice assistant built in or may require a smartphone with the Amazon Alexa mobile app to access the assistant.[44] A specific example of these devices is the Echo Auto, which is designed specifically for installation inside a vehicle. Therefore, its physical characteristics are built to function in these environments prone to constant noise and with a compact size so as not to obstruct other parts of the vehicle. This allows the use of Alexa's hands-free features, such as playing music, making calls, accessing internet searches, and many other functions to interact with the user without them taking their hands off the wheel.[8]

Alexa Skills

Alexa is Amazon's cloud-based voice service available on the brand's own devices and third-party devices with Alexa built in. It also has a number of features, which Amazon calls "Skills," that allow consumers to create a more personalized experience.[9]

To access the content of a skill, a request is made to the Alexa device or platform to invoke it. Invoking refers to the act of initiating an interaction with a particular Alexa skill. Figure 2 shows the basic structure of a request to interact with the voice assistant.[45]

Figure 2. Basic structure of a request

As detailed by Domínguez[10], the first step involves saying the wake word, which is usually "Alexa," although it can be customized with options such as "Echo," "Amazon," or another option. Next, the launch word is used, which tells Alexa the action the user wants; alternatives include start, open, begin, ask, or question.[46]

Next, the invocation name is mentioned, which users will use to refer to the specific skill. It is important to choose the name carefully to avoid misunderstandings by the voice assistant. To complement the information requested by the user, it is essential to add additional details known as utterances, which serve to specifically define the skill's course of action.[10]

Skill Programming Platforms for Alexa

According to a study, with the growing popularity of voice assistants, there are platforms provided by the manufacturer of Alexa devices, as well as various platforms from independent companies that allow skills to be developed through code programming (JavaScript or Python), blocks, or templates. Below are some examples of such programs:

- Platforms provided by Amazon Alexa or Alexa Skills Kit (ASK): this is a set of tools developed by Amazon for the design of voice assistants, specifically for the creation of Alexa skills, which includes official documentation, code examples, and an application programming interface (API).[11]
    1. Blueprints: this platform, developed by Amazon, allows the creation of custom skills without the need for coding, as it offers predefined templates that can be customized with actions, questions and answers, task scheduling, and tracking.[12]
- Independent company platforms
    2. Voiceflow: this is a tool for creating and designing voice skill conversations and interactions using block programming. By creating conversation flows, it provides an interface that is compatible with various voice assistants, including Alexa, Google Assistant, and others.[13]
    3. Dialogflow: this is a platform that incorporates natural language understanding technology, either through text or audio from people, facilitating the creation of conversational user interfaces in various applications, such as mobile devices, web applications, bots, and voice assistants with Alexa.[14]

Types of programming for skills

For the creation of Alexa skills, two types of programming can be broadly classified: block programming and textual programming. In this context, the development approaches for both modalities will be defined on particular platforms; for example, Voiceflow for block programming and Alexa Skill Kit for textual programming.

Block-based programming in Voiceflow

Block-based programming, also known as visual programming, provides an understanding of the flow of voice conversations, allowing for rapid iteration and creativity throughout the design of voice assistants.[15] The Voiceflow platform allows the creation of Alexa skills through block-based programming. Here, you can drag and drop function blocks to define conversation flows and create rapid skill prototypes.[16] Some of the features of this tool include integration with information from other IoT platforms, an interface for verifying skill functionality, and the ability to publish directly to the Alexa Skills Kit development console (Alexa Developer Console).[13] The Voiceflow interface contains multiple blocks for developing skill programming, as shown in figure 3.

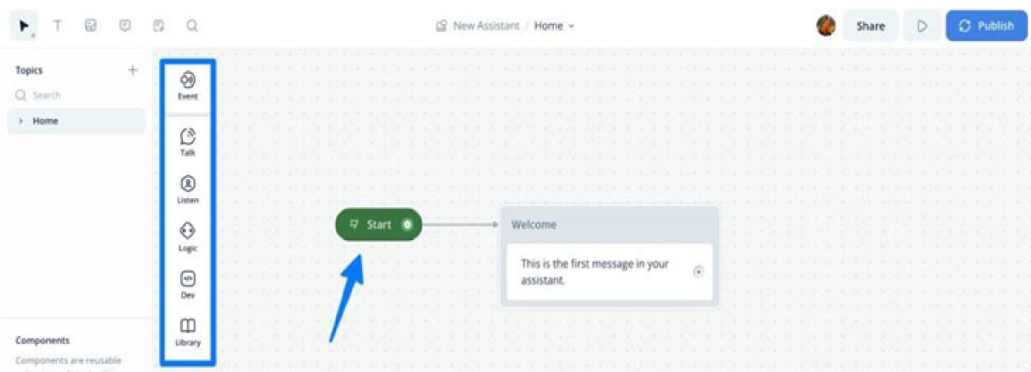

Figure 3. Voiceflow interface and its basic components[17]

As explained Tahsim[18], the blocks represent the different functionalities that can be incorporated into the conversation structure and are classified as follows:

- Event Block: represents the start of the skill's conversation flow, where you can configure the initial actions and the main intent of the voice assistant.

- Talk Block: allows voice responses to be sent to the user. Among the response options in the block, you can transform text into audio, transmit audio, display an image or video, and finally project multimedia files.
- Listen Block: captures user input, whether voice or text, to guide them through the established conversation flow. This is achieved by identifying commands stored in the platform's database, where two paths are established for the conversation flow: the alternative choice block to direct to a predetermined result and the capture block, where information provided by the user is recorded.
- Logic Block: controls decision-making in the assistant's programming and stores relevant data such as conditions, variables, loops, code termination, etc.
- Developer Block (Dev): enables code snippets that allow the skill to be executed and simulated based on the established conditions, facilitating error verification.
- Library Block: facilitates access to block templates so that they can be edited and customized to the user's needs.

It should be noted that this platform can connect to external services through API blocks, which enable access to variables from external devices or websites. This block consists of several parts, as shown in figure 4:

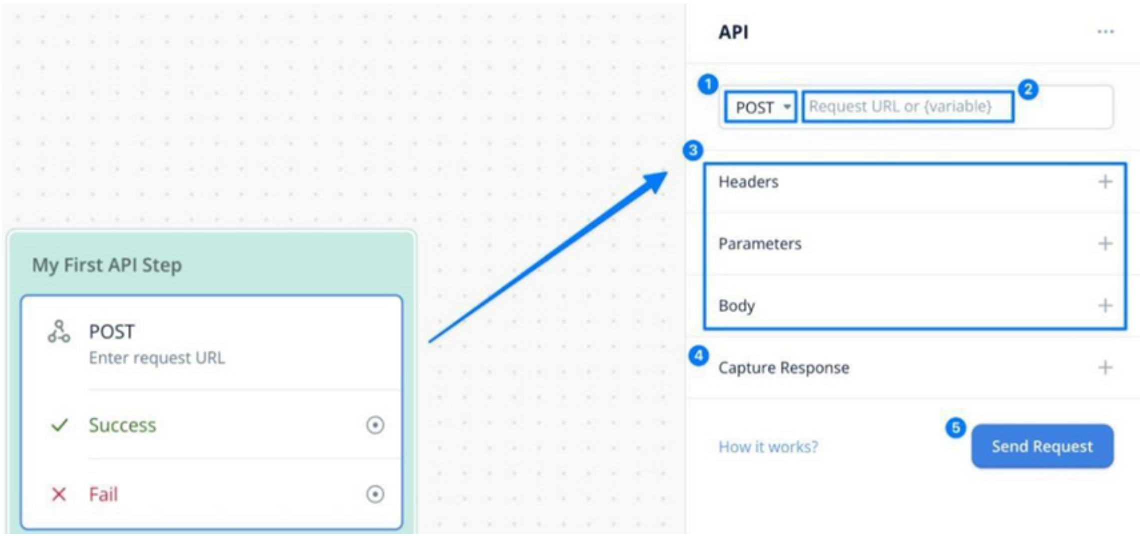

Figure 4. Parts of the API Block[17]

1) Request type: these are requests that involve retrieving data from an external database or sending data to another source, among other options. There are five requests available:
   - GET: Retrieves information.
   - POST: Creates information.
   - PUT: Completely updates information.
   - DELETE: Deletes information.
   - PATCH: Partially updates information.

2) Endpoint: refers to the URL code or variable in which the API data is set or requested.
3) Header, body, and parameter: this is additional information for accessing or directing the API, such as usernames, passwords, authentication tokens, etc.
4) Response capture/Save variables: saves the information obtained from the API, allowing it to be set as a response to continue the conversation flow.
5) Request submission: allows you to verify the API request to the required external site.

Textual programming in Alexa Skills Kit

This type of programming is fundamental to the advancement of voice assistants, giving developers the ability to devise custom commands and responses. Textual programming involves the use of programming languages such as JavaScript or Python to create custom skills for Alexa[15]. In this context, Amazon offers the Alexa Skills Kit (ASK) platform, which integrates tools, documentation, source code examples, and APIs designed for the creation of specific skills for Alexa. As shown in figure 5, the flow for developing Alexa skills has the following structure:

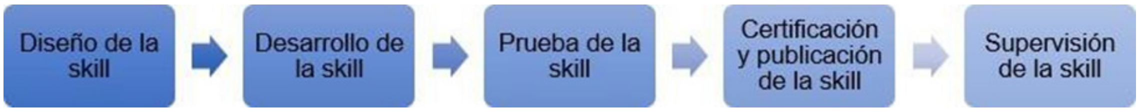

Figure 5. Alexa skill development workflow[19]

Below are details of certain aspects of each phase of the skill development process:

- Design: in this phase, the voice interaction model is selected, the words and phrases that users can use to activate the skill are defined, visual aspects and the possibility of incorporating external applications are included, as well as other elements.[20]
- Development: at this stage, it is possible to develop a skill in two different ways. In the first option, a template or pre-designed models are used in ASK, where the interaction model, intents, and example expressions are configured to customize the skill.[20] The second alternative is also developed on the same ASK platform, but it can be customized through textual programming using languages such as JavaScript, Python, C, Go, Ruby, among others.[20] To handle user requests, it is based on the programming logic of two fundamental functions: canHandle and handle, which are found in the controllers for each skill intent.

The canHandle function establishes the conditions or criteria that must be met for the controller to take responsibility for handling a particular request.

The handle function is responsible for generating a response to the user. When the skill receives a request, the canHandle function within each controller evaluates whether that specific controller is suitable for handling the request based on the previously established conditions. If the condition is met, the controller

takes over handling the request and the handle function is activated to generate the corresponding response that will be communicated to the user.

As mentioned above, there are controllers that manage the flow of the skill. These controllers enable the definition of intents such as start commands, variable requests, obtaining necessary information, requesting help or specific details, as well as ending the interaction with the skill. Figure 6 presents an example of code that configures the controller for the launch request. This controller has the function of starting the skill when the skill receives the activation invocation from the user and the LaunchRequest command is detected, which triggers the execution of the skill.

```
const LaunchRequestHandler = {
  canHandle(handlerInput) {
    return Alexa.getRequestType(handlerInput.requestEnvelope) === 'LaunchRequest';
  },
  handle(handlerInput) {
    const speechText = 'Welcome to your SDK weather skill. Ask me the weather!';

    return handlerInput.responseBuilder
      .speak(speechText)
      .reprompt(speechText)
      .withSimpleCard('Welcome to your SDK weather skill. Ask me the weather!', speechText)
      .getResponse();
  }
};
```

Figure 6. Example of the controller for a skill launch request

- Testing: this section provides a testing environment for error verification that incorporates an Alexa simulator with text and audio output capabilities.
- Certification and publication: once the skill's correct performance has been confirmed, it is certified on the Amazon platform. Here, the application is verified to comply with the established requirements in terms of policies, security, functionality, as well as the voice interface and user experience. The skill is then published in the official Amazon store.
- Monitoring: finally, in this phase, it is possible to monitor usage, perform analyses, view the revenue generated by the skill, and other relevant variables.

Platforms for IoT device management

IoT platforms allow you to manage information from devices connected to the internet so that they can be controlled and managed remotely. Due to the complexity and variety of devices, there are different categories of IoT platforms depending on the applications for which they are intended. Among the most common are platforms for enabling IoT applications, connectivity, devices, data analysis, and others.

Due to the wide variety, the selection of a suitable platform will depend on various factors such as the field of application, technical and operational needs, and the scope to be achieved. Some of the best-known examples currently available include Google Cloud Platform, Particle, Amazon AWA IoTCore, Azure IoT Hub, NodeRed, etc. Blynk is another example of an IoT platform, which allows users to connect hardware to the cloud and create applications for Android, iOS, and the web. Among its features are the ability to control and manage electronic devices, as well as real-time analysis of historical data.[21] It has an important component called Blynk.Edgent to facilitate the connection of devices compatible with the platform, including microcontrollers such as ESP32, ESP8266, various Arduino models with Internet connection, Raspberry, and others. This is achieved thanks to the Blynk library, which allows the device to communicate with the platform in a bidirectional manner. As shown in figure 7, this platform allows for easy connection to devices and their monitoring or control from either mobile or web applications.

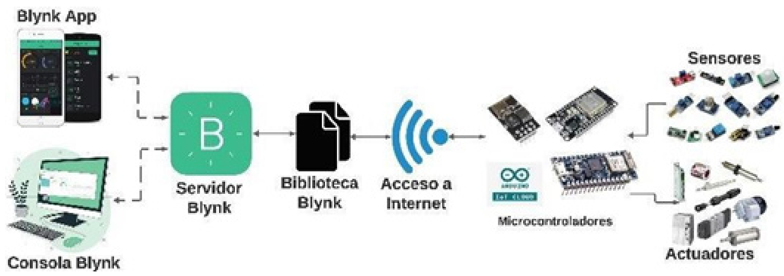

Figure 7. Integration of the Blynk platform with external devices

OBDII diagnostic standard

OBD is an on-board diagnostic system for vehicles. General Motors was the first manufacturer to incorporate a diagnostic system into its models in the 1980s, mainly to monitor injection systems and make simple adjustments to comply with Environmental Protection Agency (EPA) emissions standards. This first system was called ALDL (Assembly Line Diagnostic Link).

By 1988, the California Air Resources Board determined that all gasoline-powered vehicles must be equipped with some basic OBD capability, known as OBD-I, which uses electronic devices to monitor and enforce vehicle emission limits. With the improvement of on-board diagnostic systems and technological advances, a new generation of the OBD system was developed, which became mandatory for vehicles sold in the United States starting in 1996. This was known as OBDII, which was installed in most cars and industrial vehicles to monitor faults related to vehicle exhaust emissions .

Due to differences in regulations, emission requirements, and regional standards around the world, the OBD standard has several variations and extensions that seek to comply with specific requirements at either the local or global level. For example, the European On-Board Diagnostics (EOBD) variant adapts the OBDII standard so that vehicles manufactured and marketed in Europe comply with the region's regulations. The SAE J1979 standard defines a series of variants of the OBD standard, with which the Engine Control Unit (ECU) can be designed to meet the specifications for emissions permitted in a specific region.

Vehicle diagnostics are performed based on the main parameters related to proper engine operation, such as speed, load, engine temperature, fuel consumption, ambient temperature, air flow, and gas emissions. All related sensors are connected to the ECU, which is responsible for controlling all functions so that the engine operates properly based on the values recorded by its sensors. If any of the main parameters are outside the established ranges, the OBDII system stores and processes this information to alert the driver of a malfunction.

An improved version of this diagnostic system, called OBDIII, is currently under development, with the aim of modernizing vehicle diagnostics. It will be able to detect any type of fault and send all this information to a regulatory agency for monitoring. The remote communication of any vehicle data is intended to be carried out via satellite, so that as soon as the vehicle's malfunction light comes on, it can be detected and reported via satellite anywhere on the planet.[22]

The OBDII protocol is developed under a set of standards to facilitate primarily the diagnosis of vehicle emissions. ISO 15031-1 provides an introduction to the international standards that are part of the specifications for diagnostic systems and assigns them according to the Open System Interconnection (OSI) model. Table 1 details each of the ISO standards in accordance with the 7-layer OSI model. The use of these standards is based on SAE J1962, which specifies the physical characteristics of the diagnostic connector used.

https://doi.org/10.56294/tms2025210 ISSN: 3072-8398

Table 1. Diagnostic specifications related to emissions applied to the OSI model

OSI Model - 7 Layers	Applied standards				
Layer 7 - Application	ISO 15031-5				
Layer 6 - Presentation	ISO 15031-2, 5, 6 SAE J1930-DA / SAE J1979-DA SAE J2012-DA (OBD)				
Layer 5 - Session	Not applicable			ISO 14229-2	
Layer 4 - Transport	ISO 15031-5		ISO 14230-4	ISO 15765-2	ISO 15765-4
Layer 3 - Network					
Layer 2 - Data link Data	SAE J1850	ISO 9141-2	ISO 14230-2	ISO 11898-1	
Layer 1 - Physical			ISO 14230-1	ISO 11898 – 1, 2	

Interfaces for OBDII communication

The standard OBDII connector is known as the Diagnostic Link Connector (DLC). It is usually located under or next to the dashboard, although its location may vary depending on the model or make of the vehicle. According to SAE J1962 specifications, there are two standard 16-pin hardware interfaces. These correspond to types A and B, as shown in figure 8. Type A has a continuous slot in the middle of the two rows of pins, while type B has an interrupted slot in the middle. In addition, type A is used for vehicles with a 12V power supply, and type B is used for 24V power supplies with a distinctive blue color on the connector.

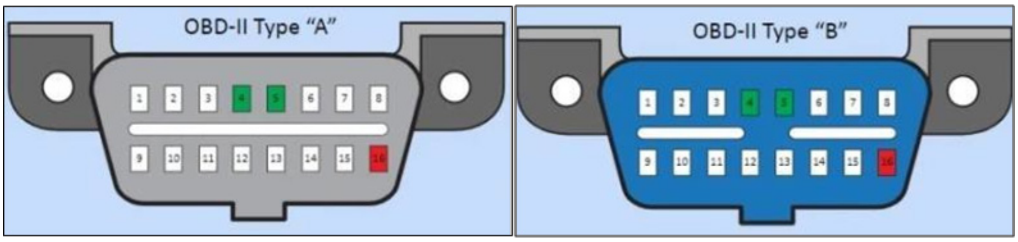

Figure 8. Interfaces for OBDII specified by SAE J1962

Details each of the pins on the OBDII connector, for both type A and type B. The use of these pins will depend on the manufacturer and the communication needs of the vehicle. OBDII systems have five communication protocols, and each manufacturer selects only one to implement in the vehicle.

The type of protocol used by the vehicle can be identified by the pins that have copper or conductive material in the specific pin. Figure 9 shows the pins that the port should have at a minimum according to the protocol, since in addition to those highlighted, the manufacturer may opt for additional pins, either for power or for purposes unrelated to the protocol, as may be the case for brand-specific diagnostic devices:

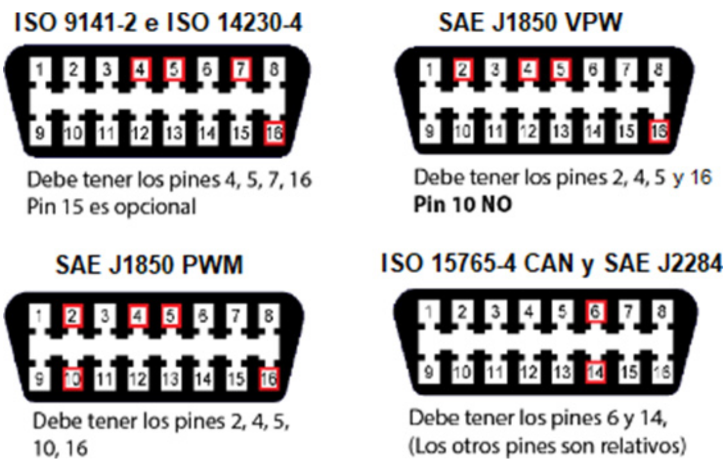

Figure 9. Connectors for OBDII protocols[23]

Communication protocols

As of 2008, the CAN communication protocol became mandatory for all vehicles manufactured in the United States. However, it is common to find models manufactured prior to 2008 or from another country of origin in circulation, with one of the other protocols supported by OBDII.[24] These five protocols are described below:

- ISO 9141-2: protocol used in European vehicles, the Chrysler brand, and some Asian vehicles between 2000 and 2004. It consists of asynchronous serial communication at a speed of 10,4 kbps. It requires the use of pins 4, 5, and 16 of the OBDII connector, while pins 7 and 15 are optional.
- ISO 14230-4 (KWP2000): this refers to the Keyword Protocol, which is very common in European and Chrysler vehicles. It requires pins 4, 5, 7, and 16, while pin 15 is optional. It supports various data transmission speeds, the most common being 5 kbps and 10,4 kbps.
- SAE J1850 PWM: pulse Width Modulation Protocol, allows communication of up to 41,6 kbps in differential mode over two wires. Requires pins 2 and 10 for communication, and pins 4, 5, and 16 for power. It is mainly used in older models of Ford and other American brands.
- SAE J1850 VPW: variable Pulse Width Modulation protocol, used mainly in General Motors vehicles. It operates at a transmission speed of 10,4 kbps over a single ground-referenced cable. It requires pins 2, 4, 5, and 16.
- ISO 15765-4 CAN (SAE J2480 CAN) and SAE J2284: CAN BUS protocol, used since 2003 and mandatory in US vehicles since 2008. Pins 6 and 14 are mandatory for the protocol, while pins 4, 5, and 16 are used for power. It allows different data transmission speeds up to 1 Mbps, in differential mode over two wires.

Controller Area Network (CAN)

The CAN communication protocol was designed by Robert Boch’s company in 1986 for implementation in the automotive industry. Years later, it was standardized due to its advantages, such as system robustness, high transmission speeds, reduced wiring, and flexibility in bus configuration, among others. This serial communication system requires two cables (CAN-H and CAN-L) to achieve a differential signal with a transmission speed of up to 1Mbps.

Each device connected to the network is called a “node.” These cannot send messages directly to other nodes, as they send their messages to the CAN bus network, where they will be available to any other node to which the message is addressed. As described, each node consists of the following components:

- Control unit: responsible for deciding what information to transmit or receive, as well as interpreting each message.
- CAN controller: it communicates with the control unit’s microprocessor to condition the data and send it to the CAN transceiver. It also receives data from the transceiver, conditions it, and sends it to the control unit’s microprocessor.
- CAN transceiver: transmits and receives data directly from the CAN bus. It is responsible for conditioning the information so that it can be used by the controllers, placing it at the appropriate voltage levels.
- Termination resistors: these are connected at the ends of the bus cable to prevent reflection phenomena. They are connected at each end, in parallel with a value of 120**Ω**, so if the system is working correctly, the resistance of the CAN line should be 60**Ω**.
- Bus cables: these consist of two twisted cables to connect all the control units in the system. They can reach a length of 500 m with a maximum transmission speed of 125 kbps, up to 100 m with a speed of 500 kbps, and 40 m at 1 Mbps. In all these cases, it is recommended not to exceed 50 cm between the bus and the connected device.

For data transmission, the controller issues a transmission start request, along with all the message data. When the bus is free, the message is transmitted to all control units, which verify the identifier and determine whether the message is required; if not, they ignore it. This message is a sequence of bits with different parameters to identify them from start to finish and allow communication between different units. Messages can be in a basic format with a 10-bit identifier or in an extended format with a 29-bit identifier, which is defined in ISO 15765-4.[11] Figure 10 shows the structure of the messages.

The message structure fields specified in figure 11 are described in greater detail in Appendix B, specifically for the standard CAN message format. For the extended format, the same fields are used as in the standard format, an additional field called Substitute Remote Request (SRR) is added, consisting of a recessive bit (logical “1”) to ensure that a frame with an 11-bit identifier can access the bus at the same time as a remote frame with a 29-bit ID extension. An additional reserved bit (R1) is also included.

When implementing this communication network in a vehicle, a connection scheme such as the one shown in figure 11 can be used. In this case, there are two CAN buses designed to connect the various modules for vehicle control. Unlike a conventional system where point-to-point connection requires a large amount of wiring, in this network the connection of each module is greatly simplified.

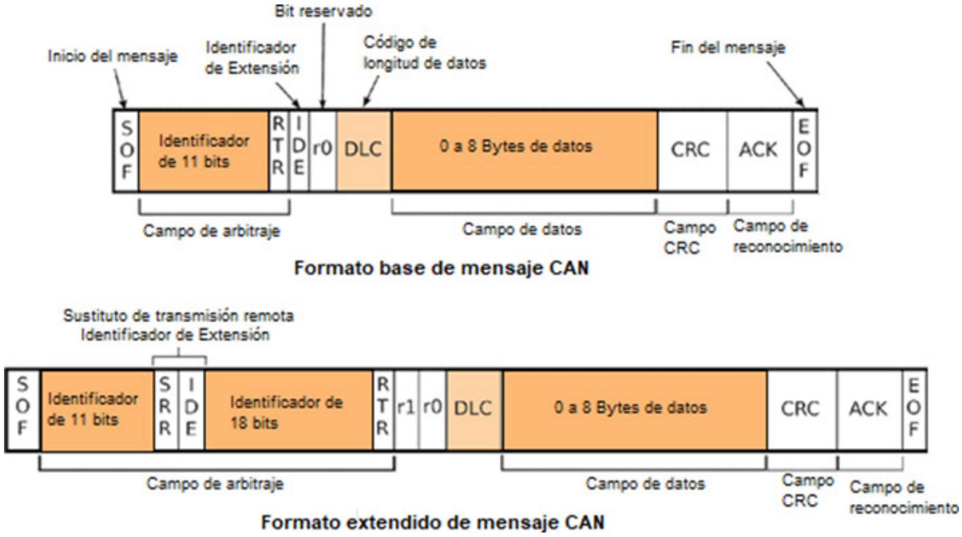

Figure 10. Message structure of the CAN communication protocol[1]

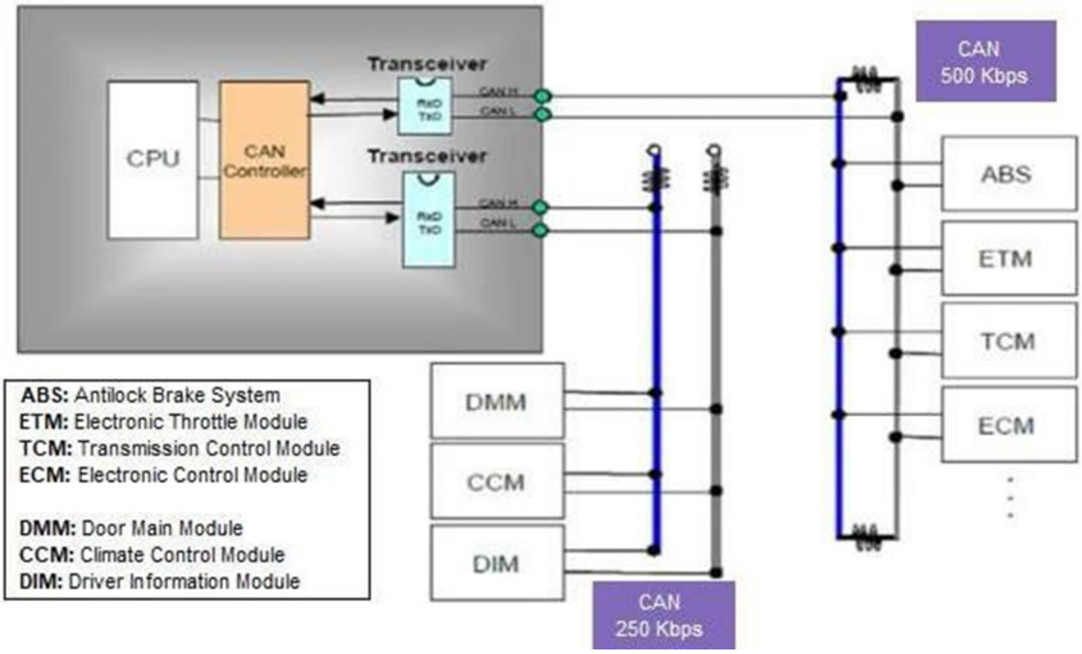

Figure 11. Connection diagram of a CAN bus in a vehicle

It should be noted that, as there are two CAN communication buses, the main modules of the vehicle are connected to the high-speed CAN bus (500 kbps), as a rapid response is required from these modules because they are essential for the correct functioning of the vehicle. These include the transmission control modules (TCM), electronic control module (ECM), electronic throttle module (ETM), among others. Other modules, such as climate control or doors, have a lower priority and are therefore connected to the lower speed CAN bus (250 kbps).

As detailed for a CAN bus with a speed between 40 kbps and 1 Mbps, the ideal voltage levels are:

- Dominant Voltage Level: 1,5 V on CAN-L and 3,5 V on CAN-H. The CAN-H voltage must be at least 0,9 V higher than CAN-L.
- Recessive Voltage Level: 2,5 V on CAN-L and 2,5 V on CAN-H. The voltage on CAN-H must not be higher than the voltage on CAN-L plus 0,5 V.

Figure 12 illustrates the voltage levels during data transmission on the CAN bus and their relationship to the voltage levels interpreted by the vehicle's microcontroller. However, these voltage levels are only reference values, as in practice they may vary due to several factors, such as the condition of the vehicle and the number of control units, among others.

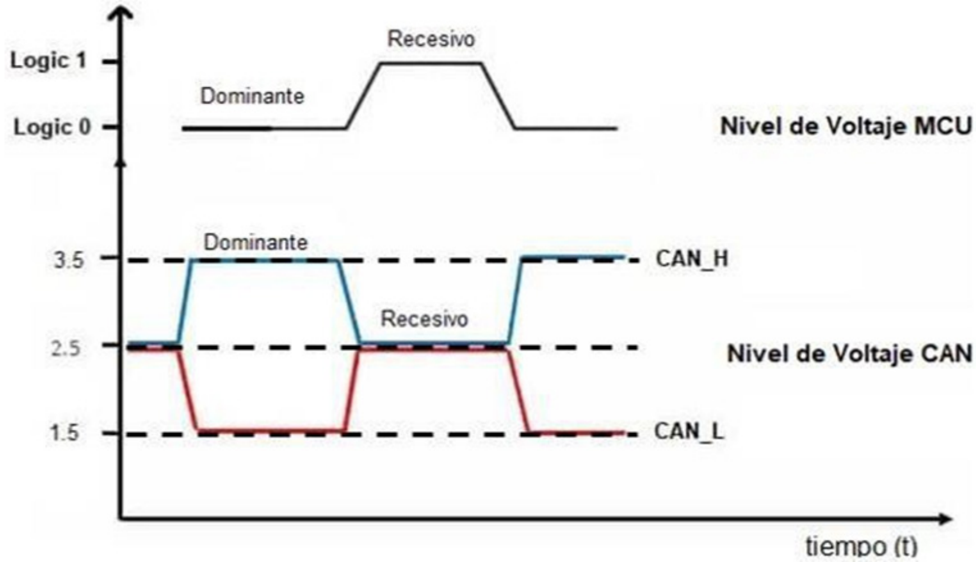

Figure 12. Voltage levels of the CAN communication protocol in a vehicle

CONCLUSIONS

The analysis of the design and implementation of a link system between Alexa Auto and the vehicle's computer highlights the impact that digital technologies have on the contemporary automotive industry. The integration of voice assistants, particularly Alexa, with diagnostic standards such as OBDII and the CAN protocol, opens up a horizon of applications that transcends simple entertainment and delves into the optimization of vehicle safety, efficiency, and maintenance. This advance not only improves the user experience but also contributes to reducing driving risks by minimizing distractions and providing real-time information in an accessible and natural way.

The studies reviewed confirm that the adoption of voice assistants in vehicles is a growing trend, as drivers value the ability to interact with the system without taking their eyes off the road or their hands off the wheel. Likewise, the incorporation of personalized skills and IoT platforms reinforces the versatility of Alexa Auto, allowing the assistant to respond not only to comfort needs, but also to critical vehicle status monitoring requirements. This approach puts the user at the center, giving them more intuitive and complete control over their driving environment.

Similarly, the study of the CAN protocol and the OBDII standard shows that the robustness and standardization of these systems enable reliable and agile communication between vehicle modules, ensuring more accurate diagnostics and early fault detection. Linking this capability to a voice assistant such as Alexa means moving towards a smart automotive ecosystem, where connectivity, artificial intelligence, and usability converge to create preventive and personalized solutions.

In summary, the research shows that the development of an integration system between Alexa Auto and the vehicle's computer is not only a technological innovation, but also a strategic proposal for the mobility of the future. Its implementation will contribute to optimizing the interaction between driver and vehicle, improving road safety, extending the useful life of cars, and promoting a more efficient use of resources. This reaffirms that the synergy between voice assistants, IoT, and diagnostic protocols not only redefines the present of the automotive industry but also lays the foundation for an orderly transition to autonomous and sustainable driving.

BIBLIOGRAPHIC REFERENCES

1. Smith S. Digital voice assistants in use to triple to 8 billion by 2023. Juniper Research; 2019.

2. Calleja J. On the use of virtual assistant for smart cities parking service deployment. Santander: Universidad de Cantabria; 2021.

3. Caicedo K, Bodero L. Diseño e implementación de un sistema de conectividad para el monitoreo en tiempo real de los sistemas funcionales de un automóvil. Guayaquil: Universidad Politécnica Salesiana; 2021.

4. Apple. iOS - CarPlay. España: Apple; 2021.

5. Android Auto. Compatibilidad con Android Auto. 2017.

6. Amazon. ¿Qué es el Alexa Skills Kit? 2024.

7. Amazon. ¿Qué es una Skill de Alexa? 2024.

8. Amazon. Certify and Publish Your Skill. 2024.

9. Amazon. Design Your Skill. 2024.

10. Domínguez J. Desarrollo de una skill educativa sobre el asistente virtual Alexa: estudia conmigo. Elche: Universidad Miguel Hernández; 2022.

11. Amazon. Echo Auto - Pon Alexa en tu coche. 2024.

12. Muñoz C. Cómo crear tus skills de Alexa, aunque no sepas de programación. Tuexperto.com; 2022.

13. Voiceflow. Quick start guide. 2024.

14. Google Cloud. Cómo implementar un chatbot de IA con Dialogflow. 2020.

15. IONOS. Programación visual: la entrada más sencilla al mundo digital. 2020.

16. Edu J, Ferrer X, Such J, Suarez G. SkillVet: Automated Traceability Analysis of Amazon Alexa Skills. IEEE Trans Depend Secure Comput. 2021;20:161-75.

17. Tahsim A. Una descripción general de los conceptos básicos del diseño en Voiceflow. Voiceflow; 2023.

18. Tahsim A. Paso API: ¿Cómo hago llamadas API? ¿Cómo conecto mi Voiceflow a las API? Voiceflow; 2023.

19. Amazon. Lleva a Alexa en tu vehículo: Todo lo demás. 2024.

20. Amazon. Test and Debug Your Skill. 2024.

21. Blynk.io. Introduction - Blynk Documentation. 2024.

22. Villamar I. Estudio y análisis de los sistemas de diagnóstico en los automóviles modernos sistemas OBD. Cuenca: Universidad de Azuay; 2018.

23. ElectrOnline. MINI ELM327 V2.1 Bluetooth HH OBD OBDII avanzado calidad. 2024.

24. Falch M. OBD2 explained - a simple intro. 2023.

25. AutomotrizEscaner. Escáner Automotriz ELM327. 2024.

26. AV Electronics. Módulo WiFi ESP8266EX. 2024.

27. AV Electronics. Arduino Nano. 2024.

28. Electronilab. Módulo CAN bus MCP2515. 2024.

29. ENDADO. Sensor temperatura del refrigerante HELLA 89422-35010. 2024.

30. Fernández F. La sonda lambda o sensor de oxígeno. 2012.

31. Fernández R. Amazon Alexa Auto Tech: The ultimate guide to Alexa in your car. MUO. 2022.

32. Gonzales P. Tecnología HC. Los coches que mejor te hablan. 2022.

33. HELLA. Sensor de temperatura del refrigerante. 2024.

34. Litnevskyi S. Manual de usuario de Excel Mazda SkyActiv OBD-II calc (FORScan). 2021.

35. MACTRONICA. Convertidor DC-DC 12V a 5V 1A reductor. 2024.

36. Mazda CX-5 Service & Repair Manual: Controller Area Network (CAN) System. 2016.

37. Motores Auto. Sistema de inyección: partes, funcionamiento, tipos. 2023.

38. Naylamp Mechatronics. Módulo CAN MCP2515. 2023.

39. Opinautos. ¿Qué es el sensor de oxígeno (o sonda lambda) en un Hilux? 2024.

40. Ortiz A, Dávila R. Implementación de un asistente virtual para los estudiantes de pregrado de una universidad peruana. Rev Conrado. 2023;19(92):121-8.

41. Rudrawar K, Choudhar N, Meshram A. Voice assisted bots for automobile applications. In: Proceedings of the 3rd International Conference on Advanced Technologies for Societal Applications. Springer; 2021. p. 489-97.

42. STP. Sistema de combustible y aditivos de combustible. STP; 2023.

43. Toyota Venza. Toyota Venza: system diagram. Toyota Venza; 2016.

44. Transductor.net. Apertura eléctrica de puertas actuadores. 2023.

45. Varga A. CANDRIVE. GitHub; 2023.

46. Villén J. Simulador de la ECU de un vehículo con protocolo ISO 9141-2. Sevilla: Universidad de Sevilla; 2016.

FINANCING

None.

CONFLICT OF INTEREST

Authors declare that there is no conflict of interest.

AUTHORSHIP CONTRIBUTION

Conceptualization: Alexis Javier Villacis Ninasunta, Mariana Pinargote Basurto.
Data curation: Alexis Javier Villacis Ninasunta, Mariana Pinargote Basurto.
Formal analysis: Alexis Javier Villacis Ninasunta, Mariana Pinargote Basurto.
Drafting - original draft: Alexis Javier Villacis Ninasunta, Mariana Pinargote Basurto.
Writing - proofreading and editing: Alexis Javier Villacis Ninasunta, Mariana Pinargote Basurto.