PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR
FACULTAD DE INGENIERÍA
ESCUELA DE SISTEMAS

IMPLEMENTACIÓN DE UN AGENTE CONVERSACIONAL PARA NEGOCIO DE REPUESTOS AUTOMOTRICES INTEGRADO A PLATAFORMAS DE MENSAJERÍA INSTANTÁNEA

PATRICIO JAVIER PÉREZ VALLEJO

DISERTACIÓN PREVIA LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN SISTEMAS Y COMPUTACIÓN

DIRECTOR: ING. FABIÁN DE LA CRUZ

QUITO, 2019

Copyright © 2019 por Patricio Pérez.
Todos los derechos reservados.

Abstract

In the present work, the components and principles on which Chatbots are built, as well as its history through time, are investigated. The different technological platforms for the implementation of Chatbots are analyzed and compared at present.

Mundirepuestos, being a company that lacks a Chatbot and evidencing the possible benefits and application areas of the use of it, is in the process of implementing a strategic plan, which focuses on the development of this technology in order to increase the company sales and it´s reach to their costumers customers.

It is for this motivation, that through the (Personal Software Process) PSP framework and a cascade methodology, a conversational agent that is meant to be hosted in the cloud programmed in Node.js Express is planned, designed, implemented and tested.

Resumen

En el presente trabajo se investiga los componentes y principios con los que se construyen los chatbot, así como su historia a través de tiempo. Se analizan y comparan las diferentes plataformas tecnológicas para la implementación de Chatbots en la actualidad.

Mundirepuestos, siendo una empresa que carece de un chatbot y evidenciando los posibles beneficios y áreas de aplicación de la utilización del mismo, se encuentra en un proceso de implementación de un plan estratégico, que se enfoca en el desarrollo de esta tecnología con la finalidad de aumentar sus ventas y alcance a sus clientes.

Es por esta motivación, que mediante el marco de trabajo Proceso de Software Personal (PSP) y la metodología en cascada se planifica, diseña, implementa y prueba un agente conversacional que se encuentra alojado en la nube programado en Node.js Express.

Índice

1. Marco Teórico y Conceptual 1
1.1 Antecedentes 1
1.1.1 Mundirepuestos 2
1.1.2 Planteamiento del problema 3
1.1.3 Objetivos 3
1.2 Chatbots 3
1.2.1 Breve historia 4
1.2.2 Recientes desarrollos 7
1.2.3 Plataformas 9
1.2.3.1 *DialogFlow* 9
1.2.3.2 *Microsoft Bot Framework* 12
1.2.3.3 *IBM Watson Conversation* 13
1.2.3.4 *Amazon Lex* 14
1.2.3.5 *Wit.ai* 16
1.2.3.6 *Manychat* 17
1.2.3.7 *Semantic Machines* 17
1.2.3.8 *Botkit.* 18
1.2.3.9 *Rasa Natural Language Understanding* 18
1.2.3.10 *Digital Genius.* 19
1.2.3.11 *Pandorabots* 19
1.2.3.12 *Chatscript* 20
1.2.3.13 *Conversable* 20
1.3 Cuadros comparativos 21
1.4 Procesamiento de lenguaje natural 29
1.4.1 Tokenización 29
1.4.2 Análisis Léxico 30
1.4.3 Análisis Sintáctico 31
1.4.4 Análisis Semántico 32
1.4.5 Integración del discurso 32
1.4.6 Análisis Pragmático 33
1.4.7 Conversaciones basadas en intenciones. 33
1.4.8 Conversaciones basadas en flujo. 34
1.5 Redes Neuronales. 34
1.5.1 Red Neural Recurrente 34
1.5.2 Redes LSTM 38
1.6 Proceso de Software P 39
1.6.1 Principios. 40
1.6.2 Estructura del Proceso 40
1.6.3 Versiones 42
2 Propuesta del agente conversacional 43
2.1 Planteamiento 43
2.2 Planificación 44
2.2.1 Nivel PSP0 44
2.2.2 Nivel PSP0.1 47

2.3 Diseño 48
2.3.1 Nivel PSP0 49
2.3.2 Nivel PSP0.1 57
3 Desarrollo de prototipo 65
3.1 Implementación 65
3.1.1 Nivel PSP0 65
3.1.2 Nivel PSP0.1 84
3.2 Pruebas 94
3.2.1 Nivel PSP0 94
3.2.2 Nivel PSP0.1 100
3.3 Postmortem 108
3.3.1 Nivel PSP0 108
3.3.2 Nivel PSP0.1 111
4 Conclusiones y Recomendaciones 115
4.1 Conclusiones 115
4.2 Recomendaciones 116
5 Lista de referencias 118
6 Glosario de términos 120

Índice de tablas

Tabla 1. Comparación de bots. 21
Tabla 2. Preguntas Frecuentes (FAQ) Mundirepuestos. 55

Índice de figuras

Figura 1. Línea de tiempo de chatbots. 8
Figura 2. DialogFlow SDKs. 10
Figura 3. Proceso de Microsoft Bot Framework. 13
Figura 4. Arquitectura de Watson Chatbot. 14
Figura 5. Cree un bot de Amazon Lex que permita a los pacientes reservar citas. 15
Figura 6. Wit.ai flow. 17
Figura 7. Análisis Léxico 30
Figura 8. Análisis sintáctico 31
Figura 9. Deep learning for chatbots 34
Figura 10. The Unreasonable Effectiveness of Recurrent Neural Networks. 35
Figura 11. Una red neuronal recurrente desenrollada. 36
Figura 12. Módulo de repetición en un LSTM. 39
Figura 13. Flujo de proceso PSP. 42
Figura 14. Versiones de proceso PSP. 42
Figura 15. Flujo conversación de Chatbot. 48
Figura 16. Cabecera Inventario Mundirepuestos SUR. 49
Figura 17. Cabecera Inventario Mundirepuestos Matriz. 50
Figura 18. Estructura de "Firebase RealTime Database". 51
Figura 19. Arquitectura "One Click Integration" de Chatbot. 52
Figura 20. DialogFlow SDKs. 53
Figura 21. Fulfillment en Dialogflow. 54
Figura 22. Fulfillment en Dialogflow. 58
Figura 23. Flujo proceso – "Buscar repuesto". 59
Figura 24. Proyecto Firebase Configuración General. 66
Figura 25. "inventario.json" 66
Figura 26. RealTime Database" Inventario Mundirepuestos. 67
Figura 27. Dialogflow configuración general "Chatbot Prueba". 68
Figura 28. Dialogflow Intención "Default Welcome Intent" 1. 69
Figura 29. Dialogflow Intención "Default Welcome Intent" 1. 70
Figura 30. Dialogflow Intención "askFAQ". 71
Figura 31. Dialogflow Intención "askFAQ" custom payload 1. 72
Figura 32. Dialogflow Intención "askFAQ" custom payload 2. 72
Figura 33. Dialogflow Intención "askContacto" 1. 73
Figura 34. Dialogflow Intención "askContacto" 2. 74
Figura 35. Intención "askUbicacion" custom payload. 75
Figura 36. Facebook Developer App – Configuración básica. 76
Figura 37. Facebook Developer App – Productos. 76
Figura 38. Facebook Developer App – Configuración de Messenger. 77
Figura 39. Dialogflow – Facebook "one-click-integration". 78
Figura 40. Configuración de One Click Integration Webhook. 79
Figura 41. Entidad - Repuesto. 80
Figura 42. Entidad - Filtro. 80
Figura 43. Entidad - Marca. Autor 2018. 81
Figura 44. Entidad - Motor. 82

Figura 45. Entidad – Vehículo. 83ix
Figura 46. Intención – askRepuesto. 84
Figura 47. Intención – askForSearchParams 1. 85
Figura 48. Intención – askForSearchParams 1. 86
Figura 49. Express- “index.js”. 87
Figura 50. Express- “index.js”. 88
Figura 51. Express- Get Endpoint. 88
Figura 52. Dialogflow – Google Project. 89
Figura 53. Google Cloud Platform – Service Keys. 89
Figura 54. Express – Post Endpoint. 90
Figura 55. Express – “message-webhook.js”. 91
Figura 56. Configuración global de Heroku. 92
Figura 57. Heroku-variables de entorno. 92
Figura 58. Heroku despliegue automático. 93
Figura 59. Heroku URL de dominio. 94
Figura 60. Emulador de conversación de Dialogflow. 95
Figura 61. Intención no identificada – Dialogflow. 96
Figura 62. Historial de conversación – Dialogflow 97
Figura 63. UI de Chatbot. 98
Figura 64. Prueba inteciones 1. 99
Figura 65. Intención fallida “askMision”. 100
Figura 66. Consola Node.js Express. 101
Figura 67. Postman – Google Custom Search. 102
Figura 68. Postman – Bad Request Facebook API. 103
Figura 69. Dialogflow. Parámetros y Entidades dentro del historial. 104
Figura 70. Dialogflow - Asignación correcta de una Intención. 105
Figura 71. Búsqueda respuesta flujo parte 1. 106
Figura 72. Búsqueda respuesta flujo parte 2. 107
Figura 73. PSP – Time Log. 109
Figura 74. PSP0 – Resumen del plan de proyecto. 110
Figura 75. PSP0.1 – Resumen del plan de proyecto parte I. 112
Figura 76. PSP0.1 – Resumen del plan de proyecto parte II. 113
Figura 77. Gráfica de interacciones por semana de FB. 114

1. Marco Teórico y Conceptual

En este capítulo se presenta la cronología de los Chatbots y su impacto en la industria de software en la actualidad. A su vez, se estudia los conceptos claves sobre la construcción de estos agentes conversacionales; como lo son: el procesamiento de lenguaje natural y las redes neuronales.

A breves rasgos se menciona los principios y estructura del proceso del marco de trabajo "Proceso de Software Personal" (PSP) [1] que se utiliza en la elaboración del presente trabajo.

1.1 Antecedentes

Miles de chatbots ya ayudan a las empresas a mejorar el servicio al cliente, vender más y aumentar las ganancias. La relevancia de esta investigación queda demostrada por el despliegue masivo de chatbots. De hecho, hoy los chatbots se utilizan para resolver una serie de tareas comerciales en muchas industrias como comercio electrónico, seguros, banca, salud, finanzas, legal, telecomunicaciones, logística, comercio minorista, automóvil, ocio, viajes, deportes, entretenimiento, medios de comunicación y muchos otros.

Gartner Summits proyecta que más del 85% de las interacciones con los clientes se gestionarán sin un ser humano para el año 2020. Se espera que los chatbots sean la aplicación de consumo número uno de Inteligencia Artificial (AI) durante los próximos cinco años, (TechEmergence, 2016).

[1] Conjunto de prácticas disciplinadas para la gestión del tiempo y mejora de la productividad personal de los programadores o ingenieros de software, en tareas de desarrollo y mantenimiento de sistemas, mediante el seguimiento del desempeño predicho frente al desempeño real.

1.1.1 Mundirepuestos

Mundirepuestos, siendo una empresa que carece de un chatbot y evidenciando sus posibles beneficios y áreas de aplicación, se encuentra en un proceso de implementación de un plan estratégico enfocada hacia el desarrollo de esta tecnología con la finalidad de aumentar sus ventas y alcance a sus clientes.

Es una empresa que empezó sus operaciones en el año 1992 en Quito - Ecuador. Mundirepuestos se especializa en la venta y distribución de partes, piezas y repuestos automotrices de las líneas Volkswagen, Audi y Skoda.

Son importadores directos de las marcas indicadas ofreciendo a sus clientes una alta gama de variedad en repuestos tanto originales como genéricos, cumpliendo los más altos estándares de calidad y cien por ciento garantizados.

Como empresa, se esfuerza por brindar al usuario la mejor atención y servicio con apoyo en asesoría técnica de primer nivel y trabajando directamente con proveedores para ofrecer repuestos originales de alta calidad a los mejores precios del mercado.

Cuenta con servicio de entrega de sus productos a nivel nacional, dentro y fuera de la ciudad.

En su plan estratégico realizado en febrero del 2018 se reconoció como una oportunidad brindar a sus clientes y a la vez aumentar las ventas de la empresa, la implementación de un chatbot que pueda integrarse con la plataforma de mensajería Facebook Messenger.

1.1.2 Planteamiento del problema

La empresa de retail "Mundirepuestos" – venta y distribución de repuestos automotrices de las líneas Volkswagen, Audi y Skoda no cuenta con un bot inteligente que pueda emular las acciones de un encargado de servicio a clientes, de personal de ventas o postventas, lo que incurre en la pérdida de potenciales clientes y por consiguiente demostrando una pérdida de ventas significativa.

1.1.3 Objetivos

- **Objetivo General:**

Implementar un agente conversacional (chatbot) para negocio de retail de repuestos automotrices "Mundirepuestos" integrado a plataformas de mensajería instantánea.

- **Objetivos Específicos:**
  - Identificar las posibles plataformas tecnologícas de chatbots a ser implementadas.
  - Escoger la plataforma más adecuada para la elaboración del sistema.
  - Analizar el potencial incremento de clientes dentro de la empresa gracias al uso de chatbots.
  - Implementar el sistema en una plataforma Cloud mediante la metodología de PSP.

1.2 Chatbots

Los chatbots son programas, software, que utilizan procesamiento de lenguaje natural (NLP: Natural Language Processing) en un sistema de preguntas y respuestas (QA systems: question answering systems), (Fitrianie, S. 2002). Estos sistemas han sido definidos

también como sistemas expertos que usan razonamiento basado en casos (CBR: case base reasoning), (Wallace, R. 2003).

Los chatbots o agentes conversacionales son programas de software que interpretan y responden a las declaraciones realizadas por los usuarios en lenguaje natural corriente.

El término "agente conversacional" puede ser analizado por las dos palabras que lo componen. La palabra "agente" es un sistema de software, el cual, viene definido por su flexibilidad, entendiendo por flexible que "un agente sea:

- Reactivo, que responda al entorno en que se encuentra.
- Proactivo, que sea capaz de intentar cumplir sus propios objetivos.
- Social, que sea capaz de comunicarse con otros agentes mediante algún tipo de lenguaje", (Wooldridge & Jennings, 1995)

1.2.1 Breve historia

A pesar que los chatbots parecerían ser un término acuñado recientemente, han existido desde que las personas desarrollaron una forma de interactuar con las computadoras. El primer chatbot fue introducido incluso antes que la primera computadora personal sea creada. Se lo denominó Eliza y fue desarrollado en el laboratorio de Inteligencia Artificial[2] (IA) del Instituto de Tecnología de Massachusetts (MIT) por Joshen Weizenbaum en 1966. Usaba reglas simples de conversación y examinaba las palabras claves en la entrada del usuario y activaba las reglas de transformación para la salida, para así simular a un psicoterapeuta. Si bien mostraba que usuarios no tan adeptos podrían

[2] IA es la inteligencia exhibida por máquinas. Agente racional flexible que percibe su entorno y lleva a cabo acciones que maximicen sus posibilidades de éxito en algún objetivo o tarea.

pensar que se comunicaban con un psicoterapeuta real, el sistema no entendía el problema del usuario. Siguiendo estos eventos, en 1991, el premio Loebner fue instituido para alentar a los investigadores de la IA a construir chatbots que puedan superar la prueba de Turing [3] y mejorar el estado de la IA. A pesar que ningún chatbot pasó la prueba hasta 2015, muchos chatbots notables ganaron premios por abarcar otros desafíos importantes. Estos incluyen ALICE, JabberWacky, Rose y Mitsuku. Sin embargo, en 2014, en una competencia de prueba de Turing para conmemorar el 60 aniversario de la muerte de Alan Turing, un chatbot llamado Eugene Goostman, que representa a un niño de 13 años, logró engañar al 33% de los jueces, superando así la prueba, (Janarthanam, 2017)

Según Janarthanam, (2017): El Lenguaje de Marcado de Inteligencia Artificial (AIML) y ChatScript se desarrollaron como una forma de escribir el conocimiento y el contenido conversacional para la mayoría de estos chatbots. Las secuencias de comandos desarrolladas utilizando estos lenguajes de secuencias de comandos pueden incorporarse a los intérpretes para crear un comportamiento conversacional. Los chatbots desarrollados para vencer la prueba de Turing fueron en gran parte conversadores con un solo objetivo: vencer la prueba de Turing. Esto no fue considerado por muchos como un avance para la IA o hacia la construcción de agentes conversacionales útiles.

Por otro lado, investigaciones en la IA, específicamente en Aprendizaje de Máquina (ML) y Procesamiento del Lenguaje Natural (NLP), dio lugar a varias interfaces de conversación, tales como sistemas de respuesta a preguntas, interfaces de lenguaje natural

[3] Una prueba de Turing es una prueba de inteligencia en una computadora en la que un humano (remitente) no debe poder distinguir entre una máquina (receptor) u otro humano (receptor) cuando las respuestas de ambos se presentan al remitente. La prueba de Turing fue diseñada por Alan Turing en 1950.

a bases de datos y sistemas de diálogo hablado. A diferencia de los chatbots creados para superar la prueba de Turing, estos sistemas tenían objetivos muy claros. Los sistemas de respuesta a preguntas procesaron preguntas en lenguaje natural y encontraron respuestas en conjuntos de datos de texto no estructurados. Las Interfaces de Lenguaje Natural para Sistemas de Bases de Datos (NLIDBS) interpretaban las consultas de base de datos planteadas en un lenguaje natural como el inglés, las convertían en SQL y devolvían los resultados como respuesta. Los Sistemas de Diálogo Hablado (SDS) podían mantener conversaciones contextuales con los usuarios para manejar tareas conversacionales, como reservar boletos, controlar otros sistemas y dar clases particulares a los estudiantes. Estos fueron los precursores de los chatbots modernos y las interfaces de conversación, (Janarthanam, 2017)

En la primera década del siglo XXI, SmarterChild fue construido por ActiveBuddy. Fue el primer intento de crear un chatbot que no solo ofreciera entretenimiento, sino que también proporcionara al usuario información más útil, como información sobre acciones, resultados deportivos, citas de películas y mucho más. Vivía dentro de AOL y Windows Live Messenger, con más de 30 millones de personas usándolo. Más tarde fue adquirida por Microsoft en 2007 por $ 46 millones. SmarterChild es el precursor de Siri de Apple y S Voice de Samsung. Siri es un asistente personal inteligente que fue desarrollado como un proyecto paralelo por SRI International y luego fue adoptado por Apple en su iOS 5 para iPhone. Ha sido una parte integral del ecosistema iOS. Siri permite a los usuarios entablar conversaciones aleatorias a la vez que proporciona información útil sobre el clima, las acciones y las entradas de películas. Gigantes tecnológicos como Samsung y Google

también han seguido los pasos de Apple desarrollando sus propios asistentes de inteligencia artificial, S Voice y Google Allo, respectivamente. También hay asistentes domésticos con voz como Amazon Alexa y Google Home, que son otra representación de los chatbots, (Khan y Das, 2018).

1.2.2 Recientes desarrollos

Cuando se recuerda el pasado, las empresas han construido sus propios chatbots individuales basados en IA para servir a los usuarios finales. En la actualidad, esta tendencia está cambiado, desde Telegram que abrió su plataforma bot para el público en junio del 2015, permitiendo así que los desarrolladores puedan crear chatbots que sirvan para numerosos servicios. Así también, Slack, anunció a sus usuarios un bot en diciembre del 2015. (Khan y Das, 2018).

Segun Khan (2018), en abril de 2016, Facebook anunció su popular plataforma Messenger para chatbots. Este fue un enfoque radicalmente diferente para las interfaces de conversación en comparación con Siri, Alexa y Cortana. A diferencia de estos asistentes personales, el anuncio de Facebook llevó a la creación de chatbots personalizados y de marca. Estos bots son muy parecidos a Siri, Cortana y Alexa, pero pueden adaptarse a los requisitos del negocio que los construye. Los chatbots ahora están listos para irrumpir varios mercados, incluido el servicio al cliente, ventas, marketing, soporte técnico, etc. Muchas plataformas de mensajería, como Skype, Telegram y otras, también se abrieron a los chatbots casi al mismo tiempo.

Actualmente en el 2018 existen una gran variedad de plataformas APIs [4] que desarrolladores pueden utilizar para desarrollar su propio chatbot, como IBM Watson Conversation, Microsoft Bot Framework, Dialogflow, Amazon Lex, Chatfuel, Wit.ai, etc. En el siguiente capítulo se realizará un estudio comparativo sobre las principales características que brindan dichas herramientas.

En la Figura 1, se presenta la línea de tiempo de los chatbots de acuerdo a los autores Rashid Khan y Anik Das publicado en su obra "Build better Chatbots".

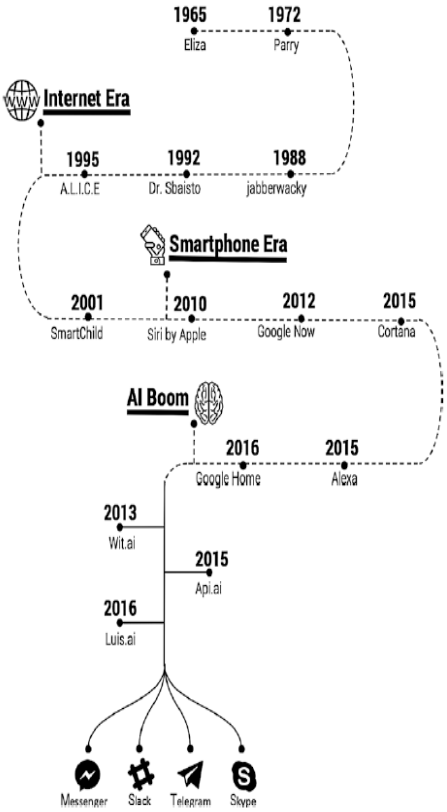


Figura 1. Línea de tiempo de chatbots.
Fuente: Rashid Khan & Anik Das (2018).

[4] La interfaz de programación de aplicaciones (API), es un conjunto de subrutinas, funciones y procedimientos que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

1.2.3 Plataformas

A continuación, se presenta y compara tecnologías, plataformas, frameworks, SDKs[5], y otras herramientas que existen en el mercado para implementar chatbots.

Tomando en consideración que Mundirepuestos es una empresa mediana y su presupuesto destinado para ejecutar su plan estratégico que consta del desarrollo de un chatbot integrado a la plataforma Facebook Messenger; se ha limitado el alance de esta sección. Como se mencionó en el capítulo anterior los chatbots usan NLP y existen dos formas para calcularlo, las conversaciones basadas en intenciones y las conversaciones basadas en flujo.

Debido a que las conversaciones basadas en flujo usan RNN y/o LTSM la implementación desde cero de las mismas, así como entrenamiento, representan un gasto considerable de recursos como tiempo, dinero y personal capacitado, por lo que se ha optado por hacer más énfasis en aquellos bots o servicios que utilizan las nuevas tendencias como lo son los basados en intenciones.

1.2.3.1 *DialogFlow*

Previamente conocido como Api.ai, fue adquirido por Google en 2016. Esta plataforma se conecta con usuarios de Google Assistant y muchas otras plataformas de mensajería instantánea a muchos dispositivos.

[5] **SDK:** Conjunto de herramientas de desarrollo de software que le permite al programador crear una aplicación informática para un sistema concreto, por ejemplo, ciertos paquetes de software, entornos de trabajo, plataformas de hardware, computadoras, videoconsolas, sistemas operativos, etcétera.

DialogFlow usa intenciones, entidades, acciones con parámetros, contextos, voz a texto y texto a voz, junto con el aprendizaje automático que funciona de manera silenciosa y entrena a su modelo. DialogFlow tiene conocimiento incorporado sobre temas como charlas informales, clima y sabiduría. Esto significa que no se tiene que entrenar al agente para estos propósitos. DialogFlow devuelve la salida como datos JSON.

En la Figura 2, se representa la arquitectura de Dialogflow y su librería de "Fullfilment" y como interactúa con la integración de plataformas y su API de agente de conversación.

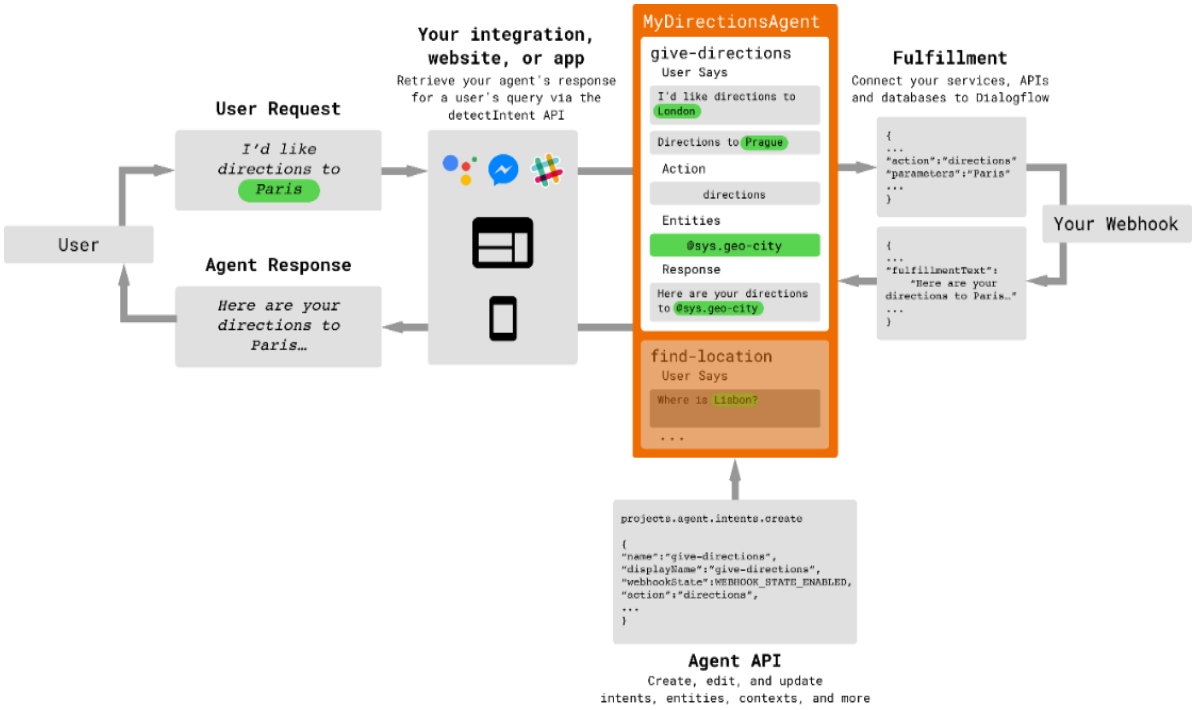


Figura 2. DialogFlow SDKs.
Fuente: Dialogflow, 2018.

Codificación rápida. Con Dialogflow, las tareas relacionadas con el código se pueden completar de manera expeditiva y de una forma rápida , ya que la plataforma tiene un editor

de código en línea. Esto proporciona a los desarrolladores herramientas para vincular a sus agentes con aplicaciones a través de Cloud Functions para Firebase o si lo prefieren, pueden crear su webhook[6] personalizado y hospedarlo en la nube.

***Aprendizaje automático accionado*.** Dialogflow ahora es compatible con las tecnologías de aprendizaje automático de Google. Esto proporciona a los desarrolladores los medios para capacitar a sus agentes para comprender la intención del usuario mediante la extracción de datos de la conversación. La plataforma también incluye más de 30 plantillas predefinidas que los desarrolladores pueden utilizar como bases.

Conversaciones naturales. Si bien los usuarios básicamente hablarán con una máquina y un conjunto de algoritmos, aún es crítico proporcionarles conversaciones naturales para que sientan que el soporte es personal y personalizado. Los desarrolladores pueden hacer esto con la ayuda de Dialogflow, ya que la plataforma admite la creación de chatbots que pueden llevar a cabo conversaciones naturales.

Charla. Con Dialogflow, los desarrolladores pueden crear algoritmos para que sus agentes conversen con los usuarios. La plataforma les permite definir frases definidas para diferentes temas o líneas de conversación, como emociones, confirmaciones; lo cual permite a los desarrolladores crear conversaciones interactivas con usuarios que no están relacionados con solicitudes de servicio y consultas.

---

[6] Un webhook (también llamado devolución de llamada web o API push HTTP) es una forma en que una aplicación puede proporcionar otras aplicaciones con información en tiempo real. Un webhook entrega datos a otras aplicaciones a medida que ocurren, lo que significa que obtiene datos inmediatamente.

Los precios de Dialogflow son los siguientes: La versión gratuita (Dialogflow Standard Edition) incluye interacciones ilimitadas de texto, pero limitadas a 3 peticiones por segundo, mientras que, la versión de pago (Dialogflow Enterprise Edition) tiene un costo de 0,002$/llamada y permite realizar hasta 10 peticiones por segundo.

1.2.3.2 ***Microsoft Bot Framework***

Microsoft Bot Framework tiene dos componentes principales: Bot Builder SDK y Microsoft Language Understanding Intelligent Service (LUIS). Botbuilder es un SDK de desarrollo que admite .NET y Node.js. Si bien LUIS es el motor NLP / NLU que contextualiza las entradas de los usuarios. Actualmente, soporta 30 idiomas y cuenta con traducción automática. El Botbuilder es de código abierto y está disponible en GitHub.

El SDK de Bot Builder proporciona la API de REST de línea directa, que permite alojar el chatbot en cualquier aplicación o sitio web. Por lo tanto, los desarrolladores se benefician de escribir una vez ejecutar cualquier metodología. También es posible incorporar Cortana para las API de voz y Bing para la búsqueda junto con LUIS en el Generador de Bot. El LUIS también se puede usar como un servicio independiente con su C # SDK, Python SDK, Node JS SDK y Android SDK. Estos beneficios hacen que Microsoft Bot Framework sea una solución ideal para las necesidades de la plataforma.

En la Figura 3, se muestra las capas de arquitectura de Microsoft Bot Framework y su relación con Microsoft Cognitive Services y su despliegue en diferentes canales de conversación

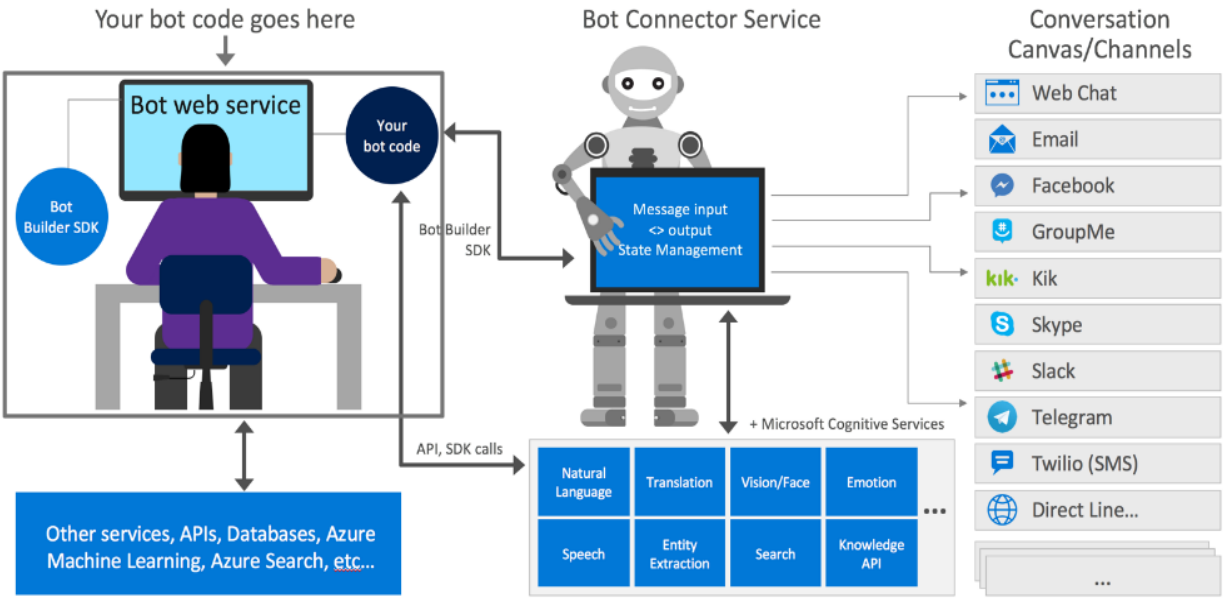


Figura 3. Proceso de Microsoft Bot Framework.
Fuente: Mukherjee, 2017.

1.2.3.3 ***IBM Watson Conversation***

Watson es una plataforma de IA proporcionada por IBM que puede comprender todas las formas de datos, interactuar con las personas y aprender de esa interacción. IBM ha trasladado la tecnología de Watson a la nube y había lanzado su API, lo que permite al usuario crear robots de conversación. Está construido sobre redes neuronales y sus componentes principales son intenciones, entidades y diálogo.

El Asistente virtual de Watson que viene con la plataforma se puede usar para responder a las consultas espontáneas de los usuarios y se puede personalizar para las necesidades de un negocio individual.

Watson proporciona SDK (Kits de desarrollo de software) para Node.JS, Java, Python, iOS y Unity; el mismo que permite implementar bots en plataformas de mensajería, dispositivos móviles e incluso robots. Los datos en el bot también son seguros ya que la plataforma permite a los usuarios optar por no compartir datos. También ofrece una integración rápida con una amplia gama de redes, canales y entornos.

En la Figura 4, la plataforma del servicio de conversación de Watson (1), se enlaza con cualquier plataforma de mensajería como Slack, Messenger, entre otras. Posteriormente se puede desplegar el chatbot en una aplicación propia o de terceros (3) que finalmente se comunica con bases de datos y otras fuentes externas como repositorios (4).

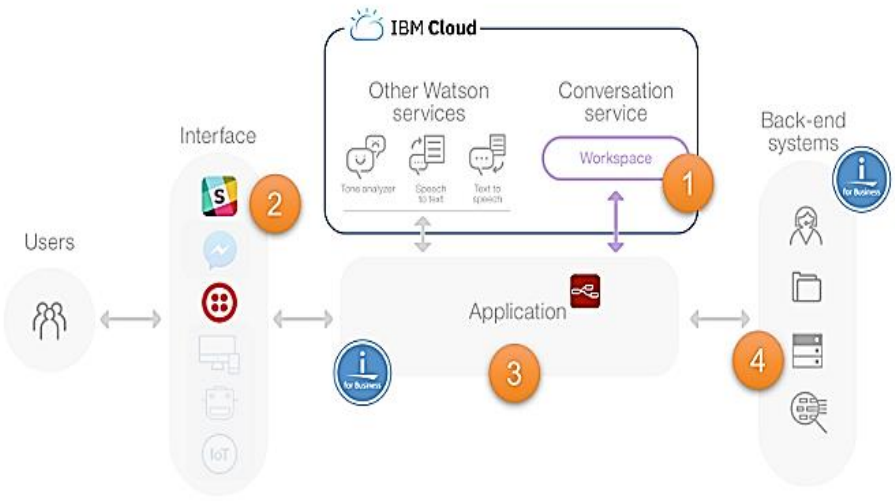


Figura 4. Arquitectura de Watson Chatbot.
Fuente: IBM 2018.

1.2.3.4 ***Amazon Lex***

Amazon Lex es la tecnología de fondo utilizada como motor NLP / NLU para Amazon Alexa. Los desarrolladores pueden usarlo para crear sus propios asistentes

personales que tienen poderosas capacidades de aprendizaje automático. Junto con Lex, Amazon también ofrece Polly, un convertidor de texto a voz de última generación y "Rekognition", el cual es un sistema de reconocimiento de imágenes de aprendizaje profundo. Lex usa "Lambda" para el cumplimiento de la intenciones y "Cognito" para la autenticación del usuario. Todas estas soluciones se pueden combinar para abordar las complejas necesidades de los bots. Con la ayuda del servicio Lambda, el bot puede integrarse con muchas ERP y CRM Suites como Zendesk, QuickBooks, Salesforce, Microsoft Dynamics, Marketo y HubSpot.

Los bots se pueden implementar en plataformas móviles, web, de escritorio e incluso de IoT. Es posible agregar programáticamente expresiones de ejemplo, crear ranuras, agregar valores de ranura, etc.

En la Figura 5, se muestra la arquitectura de la plataforma Amazon Lex y como interactúa con los diferentes APIS que ofrece AWS.

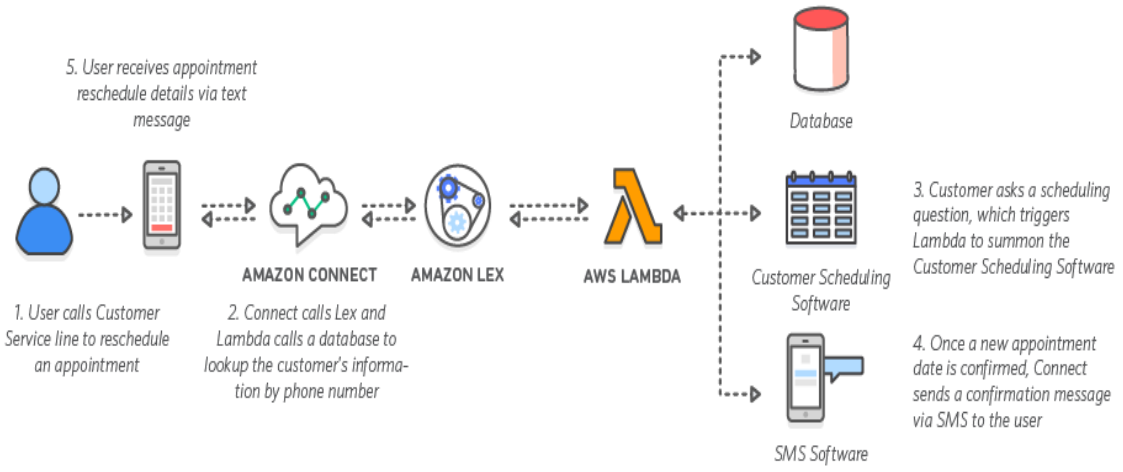


Figura 5. Cree un bot de Amazon Lex que permita a los pacientes reservar citas.
Fuente: *AWS*, 2018.

1.2.3.5 ***Wit.ai***

Es una plataforma Software como Servicio (SaaS) adquirida por Facebook, que facilita a los desarrolladores crear un chatbot para su aplicación o dispositivo. Se puede usar en plataformas que aprenden semánticamente nuevos comandos a aquellos que el desarrollador aporta. Los bots se pueden implementar en plataformas móviles, web, de escritorio e incluso de IoT. Es posible agregar programáticamente expresiones de ejemplo, crear ranuras, agregar valores de ranura, etc.

El concepto clave para definir comportamientos en Wit.ai es lo que se denomina "Historias". Representan esqueleto básico del posible diálogo por un ejemplo de ello. Una "historia" es una agrupación de intentos relacionados, mientras que la intención en sí misma es una entidad definida por el usuario del tipo de rasgo que no define todo el flujo.

El modelo de aprendizaje automático de la NLP está formado en base a ejemplos, lo que lo hace realmente genial, entrenando a Wit.ai con dichos ejemplos de respuesta cuando el usuario envíe solicitudes similares, responderá en consecuencia.

Wit.ai tiene un poderoso mecanismo para entender las entidades del lenguaje. Otra gran característica es asignar un rol a las entidades, lo que ayuda enormemente con el procesamiento en el lado del servidor.

En cuanto a la interacción con el lado del servidor, el wit.ai implementa los comandos "Bot envía" y ofrece integración de webhooks para acciones de bot personalizados, como llamadas a API.

En la Figura 6, se representa la intención de “Hacer una reserva en un restaurante” en la plataforma Wit.ai



Figura 6. Wit.ai flow.
Fuente: Techcrunch. 2015.

1.2.3.6 ***Manychat***

ManyChat es la plataforma de desarrollo de bots sin codificación más popular que hace que sea muy fácil crear un Bot de Messenger de Facebook. Desarrollar un bot ManyChat es como crear un flujo de trabajo o secuencia de correo electrónico. Puede usarlo para transmitir contenido o crear una comunicación basada en flujo. Es gratis para hasta 500 suscriptores y luego comienza su plan pagado desde $ 10 / mes.

1.2.3.7 ***Semantic Machines***

Una IA conversacional propietaria. Las características del sistema incluyen un motor de conversación, síntesis de voz, aprendizaje profundo, aprendizaje por refuerzo,

reconocimiento de voz, extracción de intención semántica y tecnología de generación de lenguaje (NLG).

1.2.3.8 ***Botkit.***

Está diseñado para ayudar a las personas ocupadas a construir bots para sus necesidades de forma rápida y fácil, sin tener que entender los conceptos complejos detrás de la herramienta. Proporciona una interfaz unificada para enviar y recibir mensajes. Inicialmente destinado a Slack, ahora tiene una funcionalidad ampliada para admitir la conexión a varias plataformas de mensajería. El marco tiene flujos de trabajo intuitivos organizados de manera clara y concisa, está bien documentado y ofrece una gran cantidad de ejemplos de chatbot en vivo para que los explore, por lo que es muy fácil comenzar y usarlos aún más.

Es importante tener en cuenta que el framework no tiene capacidades de NLP, pero se puede resolver conectando los servicios existentes o personalizados de NLP a través de middlewares.

1.2.3.9 ***Rasa Natural Language Understanding***

Aunque no es un marco designado específicamente para la creación de chatbots, Rasa NLU es una de las soluciones que facilitan su back-end. Mientras Botkit y Microsoft Bot se conectan a los mensajeros, Rasa NLU es similar a los servicios de NLP, que proporciona poder de procesamiento en las instalaciones. Además, “Rasa tiene una interfaz Python para integrar los extractores de entidades directamente en las aplicaciones escritas

en Python". También puede ejecutarse como un servicio dentro de otros marcos, exponiendo los puntos finales de la API REST.

1.2.3.10 ***Digital Genius.***

No es exactamente una plataforma de chatbot, sino una herramienta de eficiencia de agente de aprendizaje profundo (Deep Learning) que funciona en cualquier idioma. De acuerdo a Digital Genius "La Automatización de Procesos Conversacionales (CPA) pone su soporte al cliente en piloto automático utilizando AI para conectar conversaciones con procesos." El CPA permite que las consultas repetitivas de los clientes, como las solicitudes de reembolso, las solicitudes de estado del pedido, las cancelaciones y más, se resuelvan por completo, sin la participación del agente, incluso cuando los sistemas de terceros adicionales, como la facturación o el procesamiento de pagos, forman parte del proceso de resolución. Se instala como una capa en el software de servicio al cliente existente como Salesforce, Zendesk, etc.

1.2.3.11 ***Pandorabots***

La API de Pandorabots le permite integrar un servicio de alojamiento de bot y un motor de procesamiento de lenguaje natural en su propia aplicación. Los SDK desarrollados son Java, Node.js, Python, Ruby, PHP y Go. Pandorabots usa AIML (lenguaje de marcado de inteligencia artificial) e incluye A.L.I.C.E. (La Entidad Informática de Internet Lingüístico Artificial) - un lenguaje natural que procesa chatterbot. Los casos de uso comunes incluyen publicidad, asistencia virtual, aprendizaje electrónico,

entretenimiento y educación y debido a esto académicos y universidades utilizan la plataforma para la enseñanza y la investigación.

1.2.3.12 ***Chatscript***

ChatScript es el motor de chatbot de próxima generación que ha ganado 4 veces el premio Loebner y es la base de la compañía de lenguaje natural para una variedad de nuevas empresas de tecnología. ChatScript es un motor basado en reglas, las cuales se crean en los scripts de programa a través de un proceso llamado scripting de flujo de diálogo. Estos utilizan un metalenguaje de scripting (un script) como su código fuente. El motor de ChatScript tiene muchas características, como una potente combinación de patrones dirigida a la detección de significados, un diseño de regla simple combinado con scripts generales de estilo C, un diccionario de WordNet incorporado, una extensa ontología extensiva, control de máquinas locales a través de popen / tcpopen / jsonopen, lectura estructurada de datos Json, sitios web, y otros.

ChatScript se ejecuta en Windows, Linux, Mac, iOS o Android y cuenta con herramientas integradas para soportar el mantenimiento y prueba de grandes sistemas. El soporte de UTF8 permite scripts escritos en cualquier idioma y se encuentra bajo la licencia MIT.

1.2.3.13 ***Conversable***

Adquirido por LivePerson.” Conversable es la plataforma de software como servicio (SaaS) de clase empresarial para diseñar, crear y distribuir mensajes de AI y

experiencias de voz en múltiples plataformas, como Facebook Messenger, Twitter, SMS, Amazon Echo, Google Home, y muchos otros."

La empresa con sede en Austin Texas se está desarrollando principalmente para las compañías Fortune 500.

1.3 Cuadros comparativos

En la Tabla 1 se detalla los diferentes bots a disposición en el presente mercado con sus características, lenguajes de programación y SDKs soportados, idiomas a su disposición, planes con sus respectivos precios, así como sus áreas de aplicación o uso general. Gran parte de la información recopilada en esta sección es gracias a la investigación hecha por DataMonsters (2017).

Tabla 1. Comparación de bots.
Fuente: Autor. (2018)

Bot	**Proveedor**	**Características**	**Lenguajes de programación/ Apps/ Integración**	**Planes /Precio**	**Canales**	**Idiomas**	**Clientes/ Áreas de uso frecuente**
Dialogflow	Google	Creación y gestión de agentes Intenciones Entidades Formación Integraciones Analítica Cumplimiento Editor de código en línea Soporte multiplataforma Soporte de agente multilingüe Charla	SDKs: Android iOS Cordova HTML JavaScript Node.js .NET Unity Xamarin C++ Python Ruby PHP Epson Moverio Botkit Java	Estándar/Gratis Versión Enterprise/ $0.002 por solicitud	Agent Demo Page Actions on Google Facebook Slack Twilio IP Messaging Twilio Skype Tropo Telegram Kik LINE Spark Alexa Cortana Twitter	Portugués brasileño, chino (cantonés), chino (simplificado), chino (tradicional), inglés, holandés, francés, alemán, italiano, japonés, coreano, portugués, ruso, español, ucraniano.	Bots de nivel medio B2C (Business to Consumer), asistentes virtuales, MVPs (Minimum Viable Product)

Bot	Proveedor	Características	Lenguajes de programación/ Apps/ Integración	Planes /Precio	Canales	Idiomas	Clientes/ Áreas de uso frecuente
Microsoft Language Understanding Intelligent Service (LUIS)	Microsoft	Utiliza intenciones y entidades. Todas las aplicaciones LUIS se centran alrededor de un tema específico de dominio o contenido relacionado. Aprendizaje activo. Puede usar modelos preexistentes, de clase mundial y pre-construidos de Bing y Cortana. Implemente modelos en un punto final HTTP con un solo clic. LUIS devuelve JSON fácil de usar	C# SDK Python SDK Node JS SDK Android SDK	5 TPS (Transactions per second) / Gratis Standard 50 TPS ($ 1.50 por 1,000 transacciones)	Cortana Facebook Messenger Kik Slack Skype Web Chat	Inglés, francés, italiano, alemán, español, portugués brasileño, japonés, coreano y chino.	Está diseñado para permitir a los desarrolladores crear aplicaciones inteligentes que puedan entender el lenguaje humano y, en consecuencia, reaccionar a las solicitudes de los usuarios.
IBM Watson Conversation Service	IBM	Puede comprender todas las formas de datos, interactuar con las personas y aprender de esa interacción. Está construido sobre una red neuronal. Los componentes principales son intenciones, entidades y diálogo.	Node SDK Java SDK Python SDK iOS SDK Unity SDK	Lite - 1,000 consultas por mes Estándar - $ 0.0025 por llamada a la API; Premium - Precio según la solicitud.	Cualquier plataforma de mensajería compatible con voz, imágenes y texto.	Ingles, Japonés	Adecuado para asistentes virtuales y bot que se integran con los servicios de IBM.

Bot	Proveedor	Características	Lenguajes de programación/ Apps/ Integración	Planes /Precio	Canales	Idiomas	Clientes/ Áreas de uso frecuente
Lex	Amazon	Usando Amazon lex cualquier desarrollador puede construir chatbots conversacionales instantáneamente. Es el servicio que presta Amazon. Amazon Lex proporciona las funcionalidades avanzadas de aprendizaje profundo del reconocimiento automático de voz (ASR) para convertir el habla en texto y el lenguaje natural (NLU) para reconocer la intención del texto.	IoS y Android SDKs, Java, JavaScript, Python, CLI, Net, Ruby, PHP, Go, C++	4000 solicitudes de voz/ 0,004 USD 1000 solicitudes de texto/ 0,00075 USD	Facebook Slack Twilio Amazon Lex API	Inglés (US),	Modo de pre visualización, solo para opinar.
Wit.ai	Facebook	Se puede usar en plataformas que aprenden semánticamente nuevos comandos a aquellos que el desarrollador aporta. Los bots se pueden implementar en plataformas móviles, web, de escritorio e incluso de IoT. Es posible agregar programáticamente expresiones de ejemplo, crear ranuras, agregar valores de ranura, etc.	Node.js client Python client Ruby client Otras plataformas: HTTP API"	Gratis	Voz Texto	Albanés, árabe, azerbaiyano, bengalí, bosnio, búlgaro, birmano, catalán, chino, croata, checo, danés, holandés, inglés, estonio, finlandés, francés, georgiano, alemán, griego, hebreo, hindi, húngaro, islandés, indonesio,	Utilizado por más de 65,000 desarrolladores para crear aplicaciones y dispositivos con los que puede hablar o enviar mensajes de texto.

Bot	Proveedor	Características	Lenguajes de programación/ Apps/ Integración	Planes /Precio	Canales	Idiomas	Clientes/ Áreas de uso frecuente
						Italiano, japonés, coreano, latín, lituano, macedonio, malayo, noruego, persa, polaco, portugués, rumano, ruso, serbio, eslovaco, esloveno, sueco, sueco, tagalo, tamil, tailandés, turco, ucraniano y vietnamita.	
Manychat	Manychat	La funcionalidad básica le permite recibir nuevos usuarios, enviarles contenido, programar publicaciones, configurar respuestas automáticas de palabras clave (texto, imágenes, menús), transmitir automáticamente su fuente RSS y mucho más.	No requiere código	Gratis	Facebook	Ingles	Chatbots
Semantic Machines	Microsoft	Motor de conversación Síntesis del habla Aprendizaje profundo Aprendizaje reforzado Reconocimiento de voz	N/A	N/A	Voz Texto	Independiente del idioma	E-Commerce; Viajar; Calendario; Negocios; Buscar; Productividad; y automotriz.

Bot	Proveedor	Características	Lenguajes de programación/ Apps/ Integración	Planes /Precio	Canales	Idiomas	Clientes/ Áreas de uso frecuente
		Extractos de intención semántica. Utiliza la tecnología de generación de lenguaje natural (NLG) del motor de conversación.					
RASA NLU	Rasa Technologies GmbH	Clasificación de intención extracción de entidad	HTTP api Python	Gratis	Corre localmente	Ingles, Alemán	Aseguradoras Banca Salud Telecomunicaciones Viajes

Bot	Proveedor	Características	Lenguajes de programación/ Apps/ Integración	Planes /Precio	Canales	Idiomas	Clientes/ Áreas de uso frecuente
Chatfuel	Chatfuel	Construye un bot de Facebook, sin necesidad de codificación o experiencia	Complementos : búsqueda de Google Búsqueda de Bing API JSON Importación RSS Suscribir plugin Digerir IFTTT Zapier Entrada del usuario Chat en vivo	Gratis Chatfuel PRO $15/month.	Albanés, árabe, azerbaiyano, bengalí, bosnio, búlgaro, birmano, catalán, chino, croata, checo, danés, holandés, inglés, estonio, finlandés, francés, georgiano, alemán, griego, hebreo, hindi, húngaro, islandés, indonesio, Italiano, japonés, coreano, latín, lituano, macedonio, malayo, noruego, persa, polaco, portugués, rumano, ruso, serbio, eslovaco, esloveno, sueco, sueco, tagalo, tamil, tailandés, turco, ucraniano, vietnamita	Puede enviar mensajes con tarjetas de texto, tarjetas de imagen, galerías y complementos Facebook Messenger y Telegram	Chatbots dirigidos únicamente a la integración de Facebook
Chatscript		Las reglas se crean en los scripts de programa a través de un proceso llamado scripting de flujo de diálogo. Estos utilizan un metalenguaje de scripting (simplemente llamado "script")	ChatScript se ejecuta en Windows, Linux, Mac, iOS o Android Cuenta con herramientas integradas para soportar el mantenimiento y prueba de grandes sistemas ".	Licencia MIT	El soporte UTF8 permite scripts escritos en cualquier idioma.	ChatScript se ejecuta en Windows, Linux, Mac, iOS o Android Cuenta con herramientas integradas para soportar el mantenimiento y prueba de	ChatScript es la base de la compañía de lenguaje natural para una variedad de nuevas empresas de tecnología.

Bot	Proveedor	Características	Lenguajes de programación/ Apps/ Integración	Planes /Precio	Canales	Idiomas	Clientes/ Áreas de uso frecuente
		como su código fuente.				grandes sistemas ".	
Pandorabots	Pandorabots	AIML (lenguaje de marcado de inteligencia artificial) Incluye A.L.I.C.E.	"SDKs: Java Node.js Python Ruby PHP Go"	Gratis Hasta 1.000 mensajes / mes. Premium $ 0.0025 (USD) por mensaje	Multilenguaje	Los chatbots creados y alojados con Pandorabots aparecen en aplicaciones de mensajería y aplicaciones nativas, la web, juegos, redes sociales y dispositivos conectados.	Los casos de uso común incluyen publicidad, asistencia virtual, aprendizaje electrónico, entretenimiento y educación. Los académicos y las universidades utilizan la plataforma para la enseñanza y la investigación. Chats de texto o de voz con los consumidores

Bot	Proveedor	Características	Lenguajes de programación/ Apps/ Integración	Planes /Precio	Canales	Idiomas	Clientes/ Áreas de uso frecuente
DigitalGenius	DigitalGenius	AI predice metadatos de casos y sugiere las respuestas correctas a sus agentes. La IA aprende de la interacción de cada agente. Utiliza un modelo de red neuronal profunda, vectores de palabras, operaciones estadísticas. Algoritmos de aprendizaje profundo.	Human + AI Customer Service se instala como una capa en su software de servicio al cliente existente (Salesforce, Zendesk, etc.).	Profesional: $ 2,000 / mes	Ingles	Email Social Media Mensajería móvil Chat en vivo	Servicio al cliente

1.4 Procesamiento de lenguaje natural

Según Icapps (2017), Natural Language Processing (NLP), “es una forma en que las computadoras pueden analizar, comprender y derivar el significado del lenguaje humano”. Esencialmente, el término NLP se usa para describir la capacidad de una máquina para inferir lo que se le dice, desglosarlo, comprender su significado, determinar la acción apropiada y responder en un lenguaje que el usuario entienda.

Para hacerlo, el NLP, se basa en siete pasos claves que a continuación, se detallarán qué hay detrás de cada uno de ellos para establecer un vínculo entre una solicitud escrita y un análisis de expresión en lenguaje natural.

1.4.1 Tokenización

Antes de procesar cualquier mensaje escrito, el texto debe dividirse en palabras y oraciones para facilitar el análisis. En ese sentido, la tokenización es un procesamiento previo: identifica las unidades básicas (palabras y oraciones) que se procesarán durante el análisis. Si bien este paso puede parecer básico, la tokenización debe ser precisa para que el resto del análisis sea relevante. De hecho, dado que la tokenización es el primer paso del NLP, los errores que se cometan aquí se propagarán y causarán problemas de interpretación más adelante.

Las técnicas del NLP se pueden aplicar a "textos sucios" (mensajes que contienen errores de ortografía, espaciado incorrecto o uso incorrecto de la puntuación), lo que lo hace aún más complicado.

1.4.2 Análisis Léxico

Con un proceso de tokenización exitoso, las palabras se separan adecuadamente entre sí, siendo el siguiente paso el categorizar los tokens para facilitar su procesamiento. Una forma sencilla de clasificar los tokens es usar una categorización ya existente del lenguaje, la naturaleza gramatical de las palabras.

En la Figura 7, se presenta la tokenización de una oración, en la cual se puede ver como la misma se desambigua en artículos, adjetivos, sustantivos, adverbios, verbos y preposiciones respectivamente.

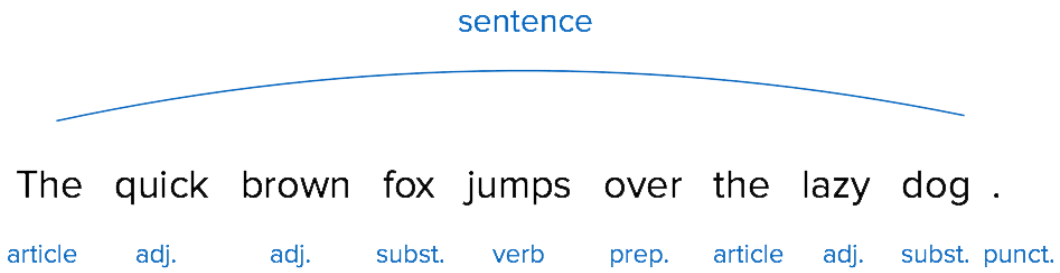


Figura 7. Análisis Léxico
Fuente: La lingüística detrás de los chatbots. ICapps (2017).

Primero, es conveniente porque la coincidencia entre una palabra y su naturaleza gramatical se realiza fácilmente mediante un diccionario. Además de eso, debido a que la gramática es el conjunto de reglas que rigen la composición de las cláusulas y frases en el lenguaje, la clasificación de tokens con los criterios gramaticales facilita los siguientes pasos, especialmente el siguiente: el análisis sintáctico.

1.4.3 Análisis Sintáctico

Mientras que la tokenización y el análisis léxico ocurren a nivel de palabra, el análisis sintáctico salta al nivel de la oración para identificar la relación entre cada palabra. Esencialmente, te lleva de regreso a la escuela: artículo + adjetivo (s) + sustantivo = grupo de sujetos y así sucesivamente.

En la Figura 8, se representa la descomposición de una oración desde el punto de vista sintáctico.

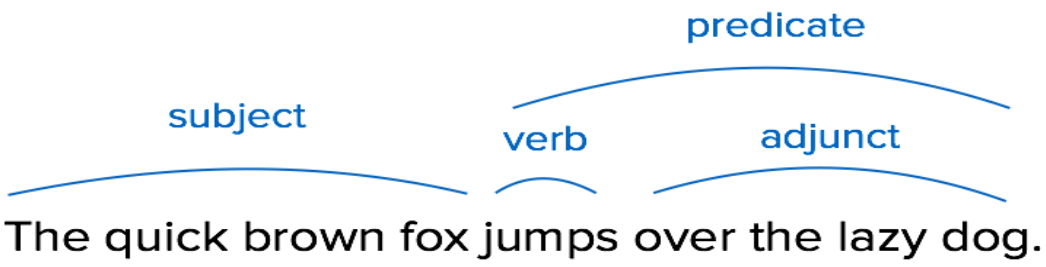


Figura 8. Análisis sintáctico
Fuente: La lingüística detrás de los chatbots. ICapps, 2017.

El análisis sintáctico proporciona el orden y la estructura de cada oración en el texto. La identificación de los sujetos, por ejemplo, es particularmente importante para uno de los siguientes pasos, la integración del discurso, que analiza el contexto alrededor de cada oración.

1.4.4 Análisis Semántico

En este paso, la computadora busca el significado de cada palabra. Lo que una vez más puede parecer un paso simple para un humano es más complicado para una computadora.

Algunas palabras son sencillas y, por lo tanto, fáciles de interpretar: "monosemio", es el ejemplo perfecto de cuerpo y alma: es un sustantivo que se refiere exclusivamente a "la propiedad de tener un solo significado".

El significado pretendido puede ser mucho más difícil de descifrar para palabras polisémicas. Según la Real Academia de la Lengua “pasar” tiene 64 significados diferentes como estos: *Llevar, conducir de un lugar a otro, Dicho de la lumbre de carbón: Encenderse bien; Rezar sin devoción o sin atención; Desecar algo al sol, o al aire o con lejía.*

1.4.5 Integración del discurso

La integración del discurso analiza el significado de las oraciones en comparación con las oraciones anteriores. Se asume la cohesión entre las siguientes oraciones del texto.

Para lograr esto, la clave está en los pronombres que deben reconocerse correctamente como tales y luego vincularse al antecedente relevante. Por ejemplo, en la siguiente declaración, la computadora debe poder reconocerla correctamente y vincularla a "Google":

Google es un motor de búsqueda. Ayuda a las personas a encontrar la información que están buscando en línea.

Si bien este ejemplo es realmente sencillo, sin embargo, una oración real y natural rara vez lo es. Las referencias anafóricas pueden ser difíciles de interpretar, especialmente en el lenguaje natural donde los antecedentes de los pronombres pueden ser poco claros o faltantes.

1.4.6 Análisis Pragmático

La pragmática estudia las formas en que el contexto contribuye al significado. Es el paso que va de lo que se dice a lo que se quiere decir, que es el más difícil de los seis pasos. La ambigüedad es difícil de manejar con las computadoras, sin embargo, esto es todo lo que un humano hace mientras habla. Dependiendo de los idiomas y situaciones, el contexto puede ser más o menos importante.

Hay dos tipos diferentes de conversaciones que los algoritmos de Procesamiento de Lenguaje Natural (NLP) pueden calcular.

1.4.7 Conversaciones basadas en intenciones.

Esto es cuando los algoritmos de la NLP usan intenciones y entidades para llevar a cabo una conversación, que al reconocer los nombres y los verbos en la declaración de un usuario y luego hacer una referencia cruzada a su diccionario, el bot puede realizar una acción válida.

1.4.8 Conversaciones basadas en flujo.

Las conversaciones basadas en flujo son el siguiente nivel de comunicación inteligente, y en la cual se alimenta una Red Neural Recurrente (RNN) de muchas muestras diferentes de conversaciones entre dos personas.

Según Shirdar sobre Kyunghyun Cho et al en 2014, los modelos basados en secuencia-secuencia se introdujeron por primera vez en el documento: "Aprendizaje de Representaciones de Frases", utilizando el codificador-decodificador RNN para la traducción de máquina estadística.

En la Figura 9, se muestra una red neural implementando un modelo de secuencia a secuencia.

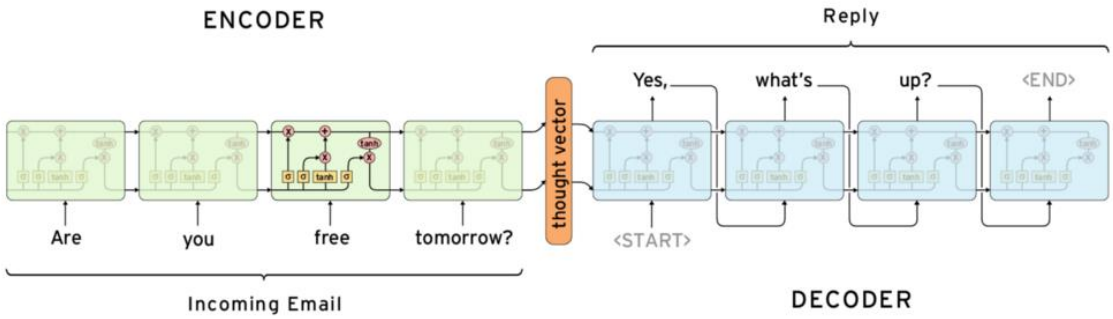


Figura 9. Deep learning for chatbots.
Fuente: Denny Britz, 2016.

1.5 Redes Neuronales.

1.5.1 Red Neural Recurrente

Es una red neuronal donde la salida no solo depende de la entrada actual, sino de una serie de entradas proporcionadas en el pasado. Dado que la salida está influenciada por

una serie de entradas anteriores, hace que la Red Neural Recurrente (RNN) sea muy efectiva en el Procesamiento del Lenguaje Natural, ya que los contextos de la siguiente palabra no necesariamente dependen solo de la palabra anterior sino de una serie de palabras antes de eso.

Dependiendo del escenario, las RNN se pueden usar para hacer frente a una variedad de tareas. Algunos de ellos se enumeran a continuación.

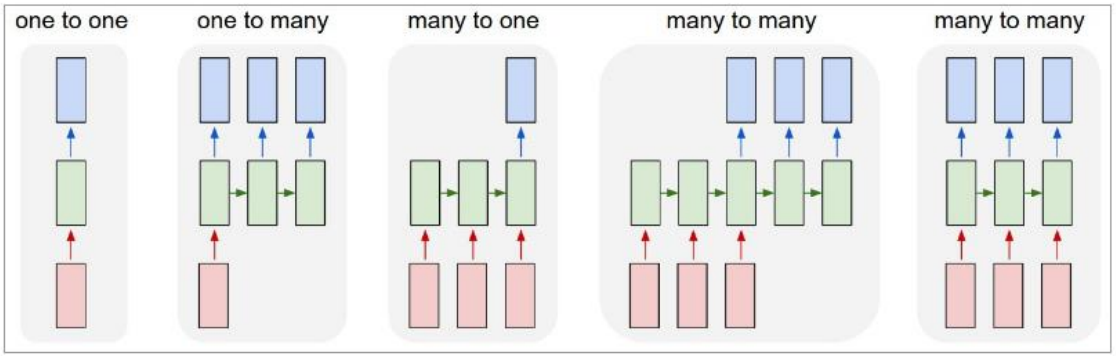


Figura 10. The Unreasonable Effectiveness of Recurrent Neural Networks.
Fuente: Denny Britz, 2016.

En la Figura 10, se representa a cada rectángulo como un vector y las flechas representan funciones (Ej. Multiplicación de matrices). Los vectores de entrada son rojos, los vectores de salida se encuentran en azul y finalmente los vectores verdes mantienen el estado de la RNN. De la izquierda a la derecha:

1. Modo básico o vainilla de procesar sin RNN, de un tamaño fijo de entrada a un tamaño fijo de salida. (Ej. Clasificación de imágenes).
2. Secuencia de salida (Ej. El subtítulo de imágenes toma una imagen y da como salida una oración de palabras).

3. Secuencia de entrada (Ej. Análisis de sentimiento donde dado una oración se la clasifica como una expresión con sentimiento positivo o negativo. También podría aplicarse en el reconocimiento de voz).
4. Secuencia de entrada y de salida (Ej. Traducción de máquina: una RNN lee una oración en inglés y la traduce a español).
5. Secuencias de entradas y salidas sincronizadas. (Ej. La clasificación de videos en donde se requiere etiquetar cada fotograma de un video. Cabe recalcar que en todos los casos no hay restricciones pre especificadas en las secuencias de longitudes porque la transformación recurrente (verde) es fija y se puede aplicar tantas veces como se quiera.

En la Figura 11, se visualiza una RNN desenrollada.

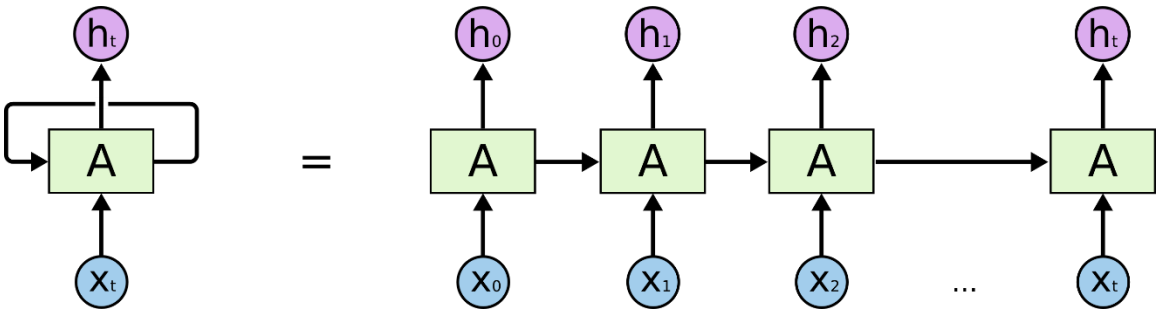


Figura 11. Una red neuronal recurrente desenrollada.
Fuente: Christopher Olah, 2015.

El campo de las redes neuronales recurrentes está bien establecido con métodos populares. Para que las técnicas sean efectivas en problemas reales, dos problemas principales deben resolverse para que la red sea útil.

Cómo entrenar la red con Backpropagation: La técnica básica para entrenar a las redes neuronales avanzadas es propagar el error y actualizar los pesos de la red.

La propagación hacia atrás se descompone en una red neuronal recurrente, debido a las conexiones recurrentes o de bucle. Esto se abordó con una modificación de la técnica de Backpropagation llamada Backpropagation Through Time o BPTT.

En lugar de realizar una propagación hacia atrás en la red recurrente como se indica, la estructura de la red se desenrolla, donde se crean las copias de las neuronas que tienen conexiones recurrentes.

Por ejemplo, una sola neurona con una conexión a sí misma (A-> A) podría representarse como dos neuronas con los mismos valores de peso (A-> B).

Esto permite que el gráfico cíclico de una red neuronal recurrente se convierta en un gráfico acíclico como una red neuronal de avance avanzado, y se pueda aplicar la propagación hacia atrás.

Cómo evitar que los gradientes se desvanezcan o exploten durante el entrenamiento. Cuando se utiliza Backpropagation en redes neuronales muy profundas y en redes neuronales recurrentes desenrolladas, los gradientes que se calculan para actualizar los pesos pueden volverse inestables.

Pueden convertirse en números muy grandes llamados gradientes de explosión o números muy pequeños llamados el problema de degradación de fuga. Estos grandes números, a su vez, se utilizan para actualizar los pesos en la red, lo que hace que la capacitación sea inestable y que la red no sea confiable.

Este problema se resuelve en las redes de Perceptron multicapas profundas mediante el uso de la función de transferencia del Rectificador, e incluso en los enfoques más exóticos, pero ahora menos populares del uso de la capacitación previa de capas sin supervisión.

En las arquitecturas de redes neuronales recurrentes, este problema se ha solucionado utilizando un nuevo tipo de arquitectura denominada Redes de memoria a corto plazo que permite capacitar a las redes recurrentes profundas.

1.5.2 Redes LSTM

Como lo señala Olah (2015), “las redes de memoria a corto plazo a largo plazo, generalmente llamadas "LSTM", son un tipo especial de RNN, capaz de aprender dependencias a largo plazo. Fueron introducidos por Hochreiter y Schmidhuber (1997). Trabajan tremendamente bien en una gran variedad de problemas, y ahora son ampliamente utilizados.”

La memoria en los LSTM se llama celdas y puede considerarlas como cajas negras que toman como entrada el estado anterior X_{t-1} y la entrada actual X_t. Internamente, estas celdas deciden qué guardar en (y qué borrar) de la memoria. Luego combinan el estado anterior, la memoria actual y la entrada. Resulta que este tipo de unidades son muy eficientes para capturar dependencias a largo plazo.

En la Figura 12, se representa como los módulos de repetición almacenan las memorias en una red LSTM.

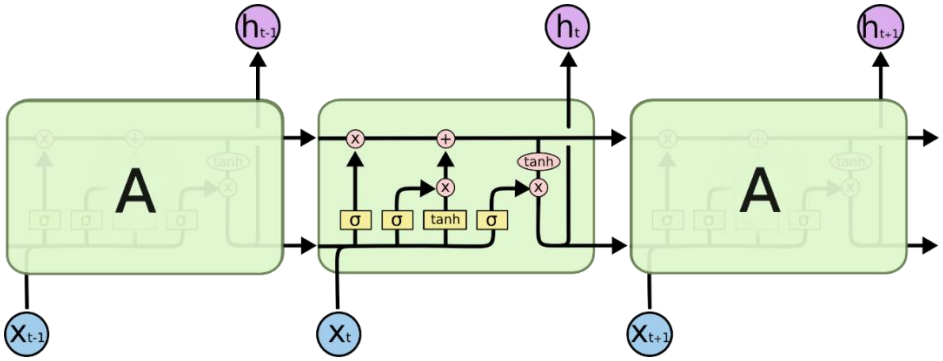


Figura 12. Módulo de repetición en un LSTM.
Fuente: Christopher Olah, 2015.

1.6 Proceso de Software Personal

Según Humphrey (2001), el Proceso de software personal (PSP) está destinado a ayudar a los ingenieros de software a comprender y mejorar su rendimiento, mediante el uso de un "procedimiento disciplinado basado en datos". El PSP fue creado por Watts Humphrey para aplicar los principios subyacentes del Modelo de Madurez de la Capacidad (CMM) del Software Engineering Institute (Instituto de Ingeniería de Software) a las prácticas de desarrollo de software de un solo desarrollador. El PSP es similar al Modelo de Madurez de la Capacidad (CMM), excepto que se enfoca en el proceso personal y se concentra en las prácticas de trabajo de los ingenieros individuales, con el objetivo de producir sistemas de software de calidad. Cuando los ingenieros utilizan el enfoque disciplinado para el desarrollo de software incluido por PSP, aprenden a ser ingenieros más competentes que producen productos de software de calidad a tiempo.

La PSP está diseñada para ayudar a los profesionales de software a usar de manera consistente las mejores prácticas de ingeniería. Les muestra cómo planificar y realizar un seguimiento de su trabajo, utilizar un proceso definido y medido, establecer objetivos

medibles y realizar un seguimiento del rendimiento en relación con estos. El PSP muestra a los ingenieros cómo administrar la calidad desde el inicio del trabajo, cómo analizar los resultados de cada trabajo y cómo utilizar los resultados para mejorar el proceso para el siguiente proyecto.

1.6.1 Principios.

Se debe seguir la manera y el enfoque correcto para realizar un trabajo de ingeniería de software, los ingenieros deben planificar su trabajo antes de comprometerse o comenzar un trabajo, y deben usar un proceso definido para planificar el trabajo. Para comprender su desempeño personal, deben medir el tiempo que pasan en cada paso del trabajo, los defectos que inyectan y eliminan, y el tamaño de los productos que producen. Para producir constantemente productos de calidad, los ingenieros deben planificar, medir y realizar un seguimiento de la calidad del producto, y deben centrarse en la calidad desde el principio de un trabajo. Finalmente, deben analizar los resultados de cada trabajo y utilizar estos hallazgos para mejorar sus procesos personales que mediante un diseño robusto que se caracteriza por un conjunto de prácticas y técnicas de implementación que permiten la construcción de sistemas de software comercializables que proporcionan una forma y función satisfactorias para los usuarios.

1.6.2 Estructura del Proceso.

De acuerdo a Humphrey (2001), PSP es un proceso personal que puede adaptarse a las necesidades del desarrollador individual y que no es específico a ninguna metodología de programación o diseño. Se puede considerar que los métodos de ingeniería de software varían de predictivos a adaptativos y en el caso de PSP es una metodología predictiva. El

entrenamiento de PSP sigue un enfoque de mejora evolutiva: un ingeniero que aprende a integrar el PSP en su proceso comienza en el primer nivel: PSP0 (presenta la disciplina y medición de procesos, estimación y planificación) y avanza en la madurez del proceso. hasta el nivel final - PSP2.1. (presenta la gestión de la calidad y el diseño) La estructura del proceso de PSP a partir de una declaración de requisitos, el primer paso en el proceso de PSP es la planificación. Hay un script de planificación que guía este trabajo y un resumen del plan para registrar los datos de planificación. Mientras los ingenieros siguen el guion para hacer el trabajo, registran su tiempo y los datos de los defectos en los registros de tiempo y defectos. Al final del trabajo, durante la fase postmortem (PM), resumen los datos de tiempo y defectos de los registros, miden el tamaño del programa e ingresan estos datos en el formulario de resumen del plan. Una vez hecho esto, entregan el producto terminado junto con el formulario de resumen del plan completado.

En la Figura 13, se visualiza el flujo de proceso que sigue PSP según Humprey. En el mismo se evidencia los guiones y data que se recolecta en cada fase dentro de una metodología en cascada.

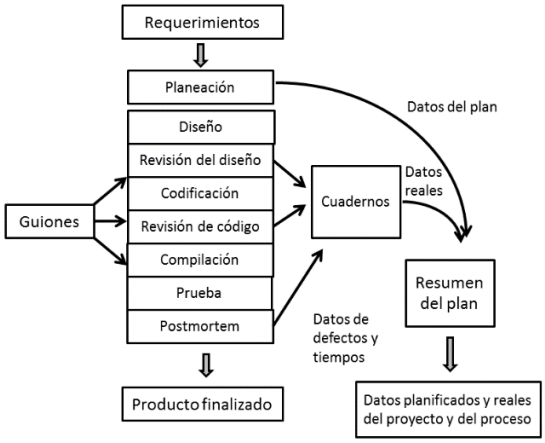


Figura 13. Flujo de proceso PSP.
Fuente: Humphey, 2001.

1.6.3 Versiones

En la Figura 14, la progresión de distintas versiones de PSP se evidencia en conjunto con la iteración de cada versión y sus requerimientos para cumplir con cada una de ellas.

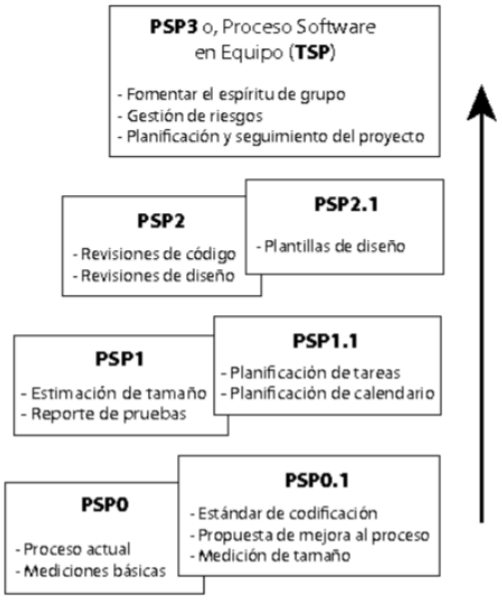


Figura 14. Versiones de proceso PSP.
Fuente: Humphey, 2001.

2 Propuesta del agente conversacional

En el presente capítulo se presenta el planteamiento de los requerimientos funcionales y no funcionales del chatbot. Se hace una propuesta formal del diseño de alto nivel de la arquitectura de la aplicación, así como diagramas de flujos de procesos y eventos dentro de la misma. También, se describen a detalle los casos de uso de cada requerimiento funcional.

2.1 Planteamiento

Mundirepuestos, siendo una empresa que carece de un chatbot y evidenciando sus posibles beneficios y áreas de aplicación, se encuentra en un proceso de implementación de un plan estratégico enfocada hacia el desarrollo de esta tecnología con la finalidad de aumentar sus ventas y alcance a sus clientes.

Se ha optado por escoger a Dialogflow como herramienta principal para el desarrollo del presente chatbot. Según lo que se expuso en el capítulo anterior, sus características y beneficios son los más idóneas para la realización del presente trabajo.
Esta sección se realizará con la versión PSP0 y la siguiente versión será en PSP0.1.

Previa solicitud explicita de la empresa se ha llegado al acuerdo que el sistema será capaz de:

Requerimientos no funcionales:

F0. Consultar el inventario del negocio almacenado en una base de datos no relacional alojada en Firebase.

F3. Desplegarse en la nube.

Requerimientos funcionales:

F1. Responder a preguntas frecuentes (FAQ).

F2. Buscar repuestos automotrices del inventario de la empresa.

2.2 Planificación

Se ha decidido distribuir las funcionalidades dependiendo del impacto y complejidad de las mismas. Las funcionalidades F0 y F1 se realizarán en la versión PSP0 y las funcionalidades F2 y F3 en la versión PSP0.1

2.2.1 Nivel PSP0

Se ha desambiguado las siguientes tareas para esta versión y se ha estimado 32 horas para el cumplimiento de dichas tareas, para posteriormente ser agregada en el resumen del plan de proyecto de PSP.

Cada tiempo será registrado en la herramienta llamada “Process Dashboard”, la misma que cuenta con plantillas para PSP y sus respectivas versiones con su estructura predefinida.

- Se normalizará la información de los inventarios de Mundirepuestos que se encuentra en libros Excel.
- Se creará un proyecto de Firebase para hacer uso del servicio de base de datos en tiempo real (Realtime Database).
- Se procederá a cargar la data que se normalizó previamente que debe encontrarse en formato JSON para poder importarse.

- Se configurará la autentificación, reglas y backups de la base de datos.
- Se creará un Agente en Dialogflow.
- Se creará la Intención por defecto de “Bienvenida” o “Default Welcome”.
- Se creará la Intención de “askFAQ” que servirá para desplegar las preguntas frecuentes.
- En esta fase también se entrenará todas las Entidades del sistema dentro de Dialogflow como “Filtros”, “Marca”, “Motor”, “Repuesto” y “Vehículo”.
- Cada pregunta frecuente tendrá una Intención a su vez con su respuesta customizada.
- Se habilitará la integración de “one-click-integration” para Facebook Messenger, la misma que proporcionará una URL de webhook.
- Se creará y configurará una aplicación de desarrollador de Facebook, que se conectará al servidor de webhook y a la página pública de Facebook de Mundirepuestos, y funcionará como un middleware entre ellos.

Para futuras referencias dentro de Dialogflow se encuentran los siguientes términos:

- **Agente:** Los agentes se describen mejor como módulos de Comprensión del Lenguaje Natural (NLU), los cuales pueden incluirse en su aplicación, sitio web, producto o servicio, y traducir las solicitudes de texto o de usuarios hablados en datos procesables. Esta traducción se produce cuando la expresión de un usuario coincide con una intención dentro de su agente.

- **Intención**: Soporte o el servicio que el usuario desea del agente que normalmente asigna la entrada de usuario a sus respuestas. En cada intención, se puede definir ejemplos de expresiones de usuario que pueden desencadenar una intención, también se puede determinar qué extraer de la expresión y cómo responder.
- **Fulfillment o Cumplimiento:** Esta parte de la conversación le permite pasar la solicitud del bot a una fuente externa y obtener respuesta y devolvérsela al usuario a través de Webhook.. La configuración de un webhook le permite pasar información de una intención coincidente a un servicio web y obtener un resultado de ella.
- **Entidades**: Se utilizan para extraer datos importantes de la entrada del usuario y en Dialogflow existen tres tipos de entidades: sistema, desarrollador y usuario.
- **Contextos**: Representan el estado actual de la solicitud de un usuario y le permiten a su agente llevar información de una intención a otro. Puede usar combinaciones de contextos de entrada y salida para controlar la ruta de conversación que el usuario toma a través de su diálogo.
- **Eventos:** Los eventos le permiten invocar intenciones basándose en algo que ha sucedido en lugar de lo que un usuario comunica. Dialogflow admite eventos de varias plataformas (como Google Assistant, Slack y más) basadas en las acciones que los usuarios realizan en esas plataformas. También puede crear sus propios eventos personalizados que pueden activarse a través del Fullfilment o la API detectIntent.
- **Frases de entrenamiento:** Ejemplos de lo que los usuarios pueden decir para que coincida con una intención en particular.

- **Acción y parámetros:** Define cómo la información relevante (parámetros) se extrae de las declaraciones del usuario. Los ejemplos de este tipo de información incluyen fechas, horas, nombres, lugares y más. Puede usar parámetros como entrada en otra lógica, como buscar información, realizar una tarea o devolver una respuesta.
- **Respuesta:** Una expresión que se habla o se muestra al usuario.

2.2.2 Nivel PSP0.1

Después de la culminación de la primera iteración de PSP0 se ha estimado 15 días laborables para esta versión. A continuación, se ha desglosado las siguientes tareas para esta versión.

- Se creará un ambiente de desarrollo de NodeJS.
- Se creará un Webserver[i7] simple con un webhook en el framework NodeJS Express.
- Se creará un endpoint[8] para la verificación de Facebook.
- Se creará un segundo endpoint para el manejo de los mensajes.
- Se conectará a la aplicación de desarrollador de Facebook que se creó en la versión PSP0, que se conectará al servidor de webhook y a la página pública, y funcionará como un middleware entre ellos.
- Se configurará la autentificación de Dialogflow en el webserver.

[7] Un webserver es un programa que utiliza HTTP para servir los archivos que forman las páginas web a los usuarios, en respuesta a sus solicitudes, que son reenviadas por los clientes HTTP de sus computadoras.

[8] Un endpoint de un canal de comunicación. Para las API, un endpoint puede incluir una URL de un servidor o servicio. El lugar donde las API envían solicitudes y donde vive el recurso, se denomina endpoint.

- Se creará la intención para “búsqueda de repuestos automotrices” de la empresa.
- Se entrenará las entidades de repuestos y filtros de búsqueda en DialogFlow.
- Se desplegará el sistema en Heroku. (Plataform as a Service).
- Se habilitará al público la aplicación de desarrollador en Facebook tras la aprobación de la misma.

2.3 Diseño

En la Figura 15, se observa el flujo de las conversaciones que puede tener el agente conversacional, tanto las entradas del usuario (inputs), las repuestas del bot y llamadas a recursos externos.

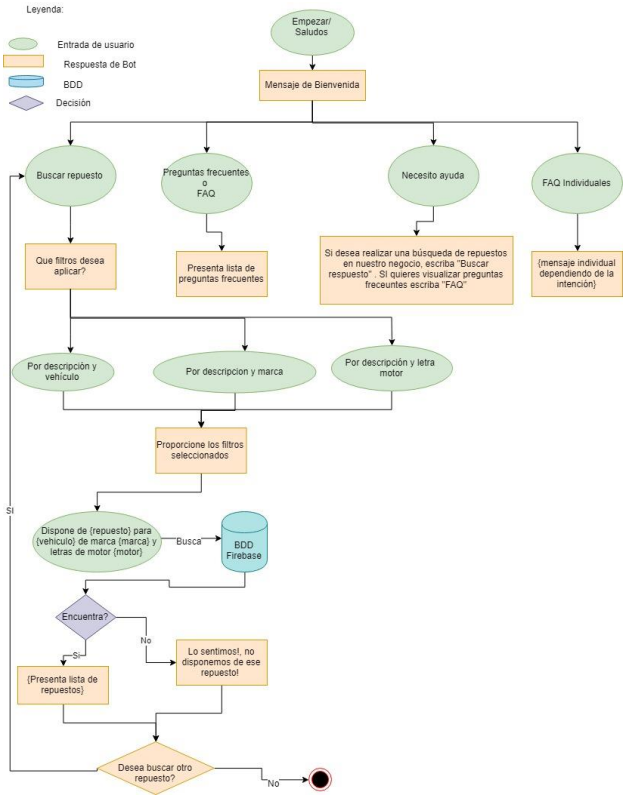


Figura 15. Flujo conversación de Chatbot.
Fuente: Autor. 2018.

2.3.1 Nivel PSP0

2.3.1.1 Normalización de datos y estructura de la base de datos Firebase.

Debido a que el inventario que manejaba Mundirepuestos se encontraba elaborado en dos libros de Excel; que corresponden a sus dos sucursales respectivamente, dicha información es mandatorio normalizar y estructurarla.

En las Figura 16 y 17, se muestra las cabeceras de los documentos de "Inventario" que se obtuvieron de ambas sucursales. Como se puede observar las cabeceras son distintas, pero se estandarizó a que todas cumplan con el mismo orden y nomenclatura como: NO, CODIGO, DESCRIPCION, VEHICULO, LETRA_MOTOR, MARCA, CANTIDAD, UBICACIÓN, PVP, PROVEEDOR. Se agregará dos columnas más SUCURSAL Y URL, que servían para identificar de que local proviene cada repuesto y para almacenar la imagen del repuesto como URL, respectivamente.

INVENTARIO MUNDIREPUESTOS SUCURSAL SUR										
COSTO	NO.	CODIGO	DESCRIPCIÓN	VEHÍCULO	LETRAS MOTOR	MARCA	CANTIDAD	UBICACIÓN	PVP	PROVEEDOR
	1	ATF DII	ACEITE CAJA DE CAMBIOS AUTOMATICA	VEHÍCULO		ATF	1	MOSTRADOR	8.00	MV
3.18	2	SAE 75W-85	ACEITE CAJA DE CAMBIOS UBX	VW	VW	GULF	12	MOSTRADOR	6.00	FULL ENERGY
14.56	3	GF-5 GALON	ACEITE DE MOTOR 10 W 30	HASTA 80,000 KM		KENDALL	3	MOSTRADOR	21.00	MOTORFILL
3.92	4	GF-5 LITRO	ACEITE DE MOTOR 10 W 30	HASTA 80,000 KM		KENDALL	12	MOSTRADOR	6.00	MOTORFILL
3.47	5		LIQUIDO DE FRENOS	TODA MARCA		ATE	6	MOSTRADOR	5.00	MOTORFILL
2.89	6		LIQUIDO DE FRENOS	TODA MARCA		WAGNER	6	MOSTRADOR	4.00	MOTORFILL

Figura 16. Cabecera Inventario Mundirepuestos SUR.
Fuente: Autor 2018.

CODIGO	DESCRIPCION	VEHICULO	LETRA MOTOR	Stock	PRECIO	PERCHA	
321 256	ABRILLANTADOR DE LLANTAS	VW	VW	1	5.00	MOSTRADOR	
G 055 538 A2	ACEITE HIDRAULICO ORIGINAL	VW	VW	1	42.69	MOSTRADOR	
G 052162A2	ACEITE CAJA ORIGINAL	AUDI VW	AUDI VW	5	48.00	MOSTRADOR	AUTOBAHN
AT 160	ACEITE HIDRAULICO UBX	VW	VW	11	8.25	MOSTRADOR	FARCONG
SAE 75W-90	ACEITE CAJA DE CAMBIOS EXTROIL	VW	VW	5	8.00	MOSTRADOR	DAVMAR TIRES
SAE 75W-85	ACEITE CAJA DE CAMBIOS UBX	VW	VW	24	8.50	MOSTRADOR	
ATF DII	ACEITE CAJA DE CAMBIOS AUTOMATICA UBX	VW	VW	24	8.00	MOSTRADOR	DAVMAR TIRES

Figura 17. Cabecera Inventario Mundirepuestos Matriz.
Fuente: Autor 2018.

Después de juntar los dos libros y de completar una limpieza tanto de datos, como limpieza de caracteres inválidos y data duplicada, se generará un archivo JSON.

En la Figura 18, se muestra la estructura JSON que tendrá la base de datos “Firebase Realtime Database”; la cual tiene un arreglo de repuestos, con una “KEY” que corresponderá a la columna “NO” y sus propiedades clave-valor como se puede visualizar a continuación.

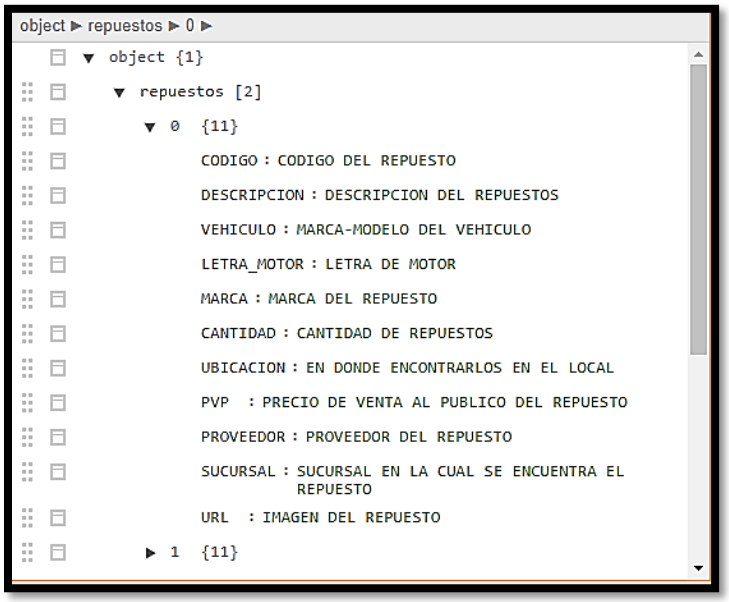


Figura 18. Estructura de “Firebase RealTime Database”.
Fuente: Autor 2018.

2.3.1.2 Arquitectura

En la Figura 19, se observa la arquitectura basada en servicios, en donde la misma empieza con la interacción del usuario a través de Facebook Messenger API, Dialogflow y sus componentes (círculo rosado) hasta un webhook “self-hosted” de Dialogflow. A continuación, se describe el flujo.

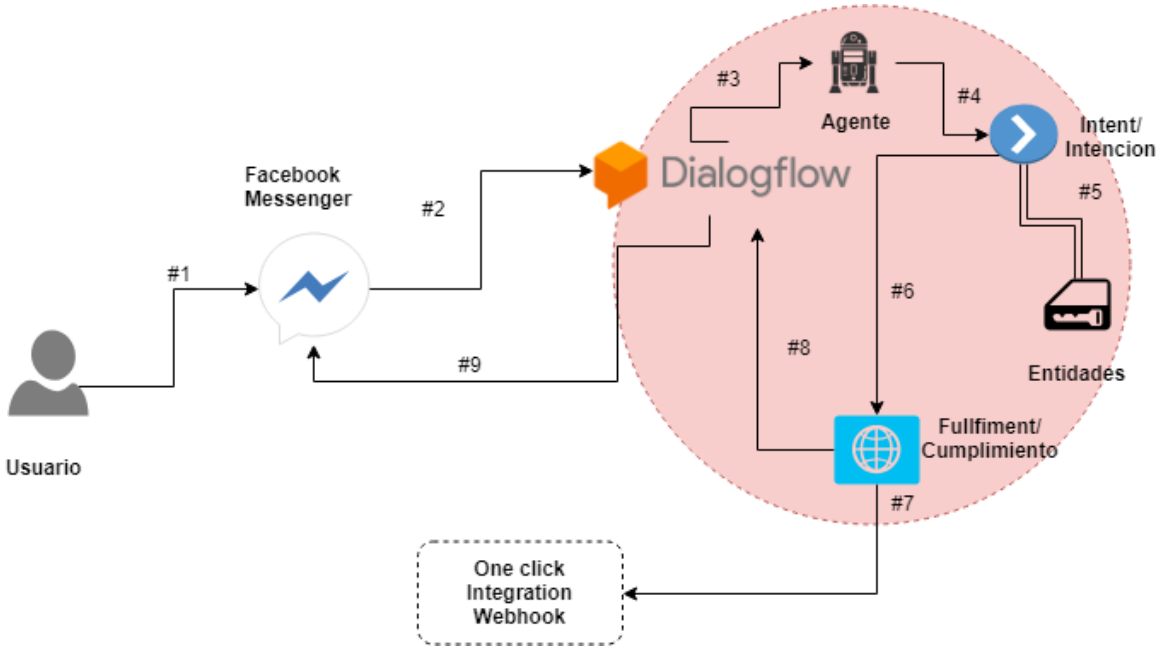


Figura 19. Arquitectura "One Click Integration" de Chatbot.
Fuente: Autor 2018.

#1 Usuario escribe un mensaje.

#2 Messenger envía el mensaje como texto.

#3 Dialogflow recibe el texto y determina a que Agente se lo envía.

#4 El Agente de Dialogflow identifica la Intención del usuario y pasa a la Intención correcta.

#5 La Intención de Dialogflow usa entidades para almacenar valores de parámetros. Ej. [filtro_vehiculo: Volkswagen Golf]. Las intenciones son opcionales y guardan valores de contexto.

#6 La intención de Dialogflow pasa la solicitud junto con las entidades para Fullfilment.

#7 El cumplimiento o "Fulfillment" usa un webhook de "one-click-integration" para Facebook Messenger.

#8 El Fulfillment entrega la respuesta (objeto JSON) desde el webhook.

#9 Dialogflow devuelve la respuesta a Messenger.

A continuación, en la Figura 20, se describe la interpretación de un agente dentro de Dialogflow. Una definición de agente puede tener muchas “Intenciones”, las cuales tienen contextos con parámetros, estos últimos, son definidos por “Entidades”. Las Intenciones son disparadas por eventos y cumplidas por el Webhook.

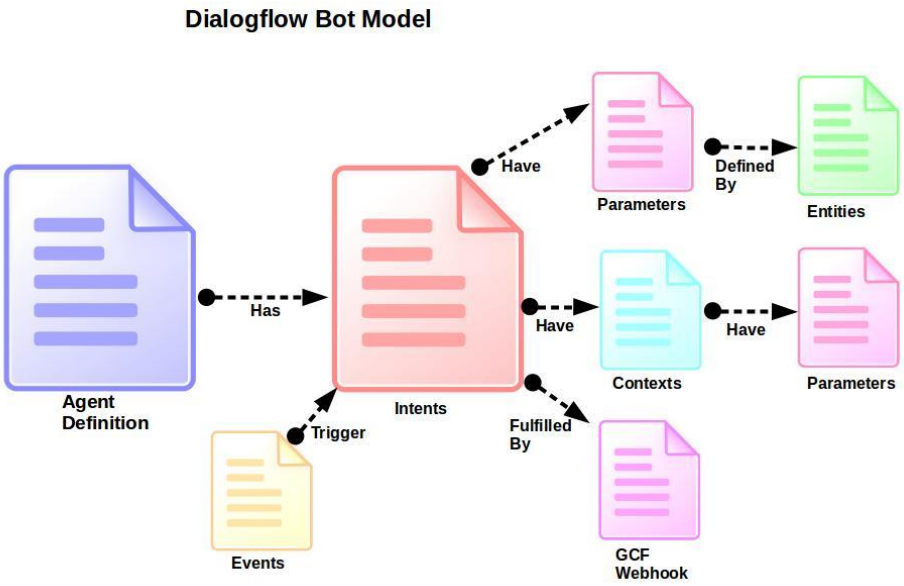


Figura 20. DialogFlow SDKs.
Fuente: Dialogflow, 2018.

El flujo del proceso de “Fullfilment” dentro de Dialogflow es el que se describe en la Figura 21.

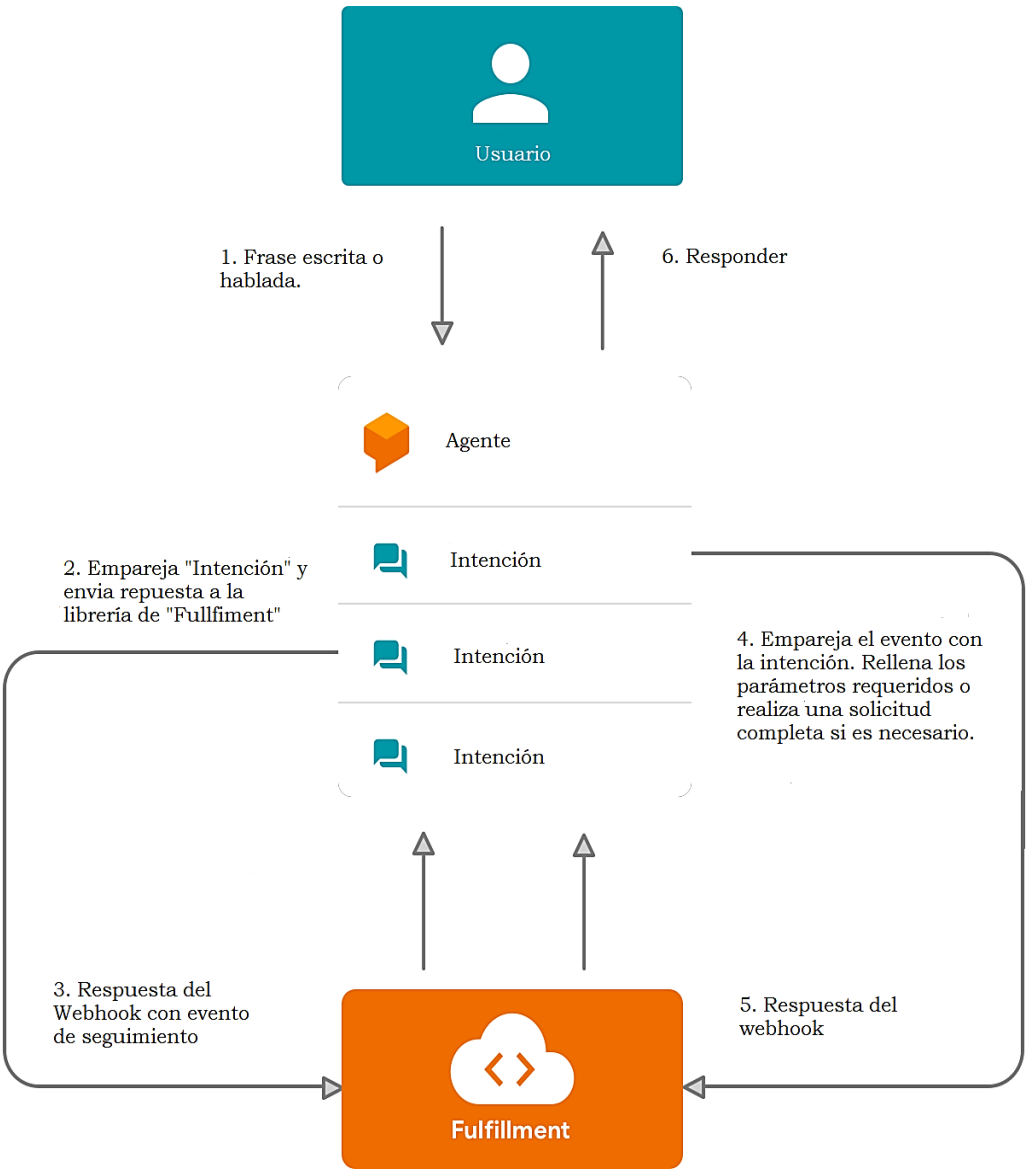


Figura 21. Fulfillment en Dialogflow.
Fuente: Dialogflow, 2018.

Casos de uso

Diagrama a detalle F1 “Responder preguntas frecuentes”.

Descripción: Permite al usuario consultar todas las preguntas frecuentes que el negocio dispone. A su vez cada pregunta puede ser llamada con una intención que haga referencia a sí misma. Las preguntas con sus respectivas repuestas, e intenciones se muestran en la Tabla 2.

El sistema desplegará todas las preguntas con sus respectivas respuestas en elementos de tipo “lista” que soporta Messenger con dichas preguntas si la Intención es “askFAQ”.

Si el sistema detecta cualquier otra intención especificada en la Tabla 2, cuando sea disparado por estas intenciones, solo enviará la respuesta como un mensaje de texto; exceptuando cuando la intención sea “askUbicacion”; en la cual se enviará un componente de tipo “card” que enviara hipervínculos con mapas de google hacia las ubicaciones de cada sucursal.

Tabla 2. Preguntas Frecuentes (FAQ) Mundirepuestos.
Fuente: Dialogflow, 2018.

No	Pregunta	Respuesta	Nota de respuesta	Intención
1	¿Quiénes son?	Somos Importadores de repuestos automotrices	Solo texto	askMision
2	¿Para qué marcas de vehículos disponen repuestos?	Volkswagen, Audi y Skoda	Solo texto	askMarcas
3	¿Cuáles son sus servicios?	Tenemos servicios de entrega y garantía de repuestos.	Solo texto	askServicios

No	Pregunta	Respuesta	Nota de respuesta	Intención
4	¿Puedo escribirles a algún correo?	Escríbenos a nuestro correo: importadoraMundirepuestos@gmail.com	Solo texto	askCorreo
5	¿ Dónde se encuentran ubicados?	Matriz: Versalles 21-78 y Acuña Sur: Av. Alonso de Angulo 127 y Cristobal Tenorio",	Elemento Card con redirección a GoogleMaps	askUbicacion
6	¿Cuáles son sus horarios de atención?	Atendemos: Lunes- Viernes: 08:00hh - 18:30hh Sábado: 08:30hh - 14:00hh	Solo Texto	askHorario
7	¿Cuáles son sus teléfonos de contacto?	Matriz: 0999007885/ 022527775 Sur: 0995090010/ 0961092010 / 025115322	Solo Texto	askContacto
8	¿Tienen una cuenta de Instagram?	Si nuestra cuenta es @Mundirepuestos_ec. Síguenos en https://www.instagram.com/Mundirepuestos_ec/	Solo Texto	askInstagram

Actores: Usuario, Messenger, Dialogflow

Pasos del flujo principal:

1. El usuario envía el mensaje de una Intención descrita en Tabla 2.
2. Messenger envía el mensaje como texto. (E1, E2)
3. Dialogflow recibe el texto y se lo envía al Agente “ChatbotPrueba”.
4. Dialogflow identifica la Intención del usuario y pasa la Intención hacia el webhook. (E3)
5. Si la intención es “askFAQ” el Fulfillment en Dialogflow entrega la repuesta, como elementos tipo “lista” dentro de un objeto JSON con las 8 preguntas frecuentes desde el webhook.
6. Dialogflow devuelve la respuesta a Messenger.

Pasos del flujo alterno:

5. Si la intención pertenece a cualquier otra intención expuesta en la Tabla 2. Se devuelve un objeto JSON con las respuestas de la intención como mensaje de tipo "mensaje".

Excepciones:

E1: Razón: No se detecta internet.

Acción: Messenger cambia de estado a mensaje no enviado.

E2: Razón: API Facebook Messenger falla.

Acción: Messenger cambia de estado a mensaje no enviado.

E3: Razón: No se reconoce ninguna Intención.

Acción: Devuelve un "default fallback intent". En este caso se envía a Messenger el mensaje →"Disculpa no lo he entendido!".

2.3.2 Nivel PSP0.1

2.3.2.1 Arquitectura

A diferencia de la arquitectura usada en la iteración PSP0, como se puede visualizar en la Figura 22, se ha añadido un webhook programado en NodeJS mediante el Framework "Express", el cual se comunica con la base de datos Firebase para hacer la consulta de repuestos del negocio. Si en la base de datos no se encuentra una imagen para un repuesto consultado; se comunica con la API de "Custom Search Google API" y el webhook

devuelve la repuesta al “Fullfilment” dentro de Dialogflow, quien se encarga de devolver la repuesta en un formato JSON a Messenger.

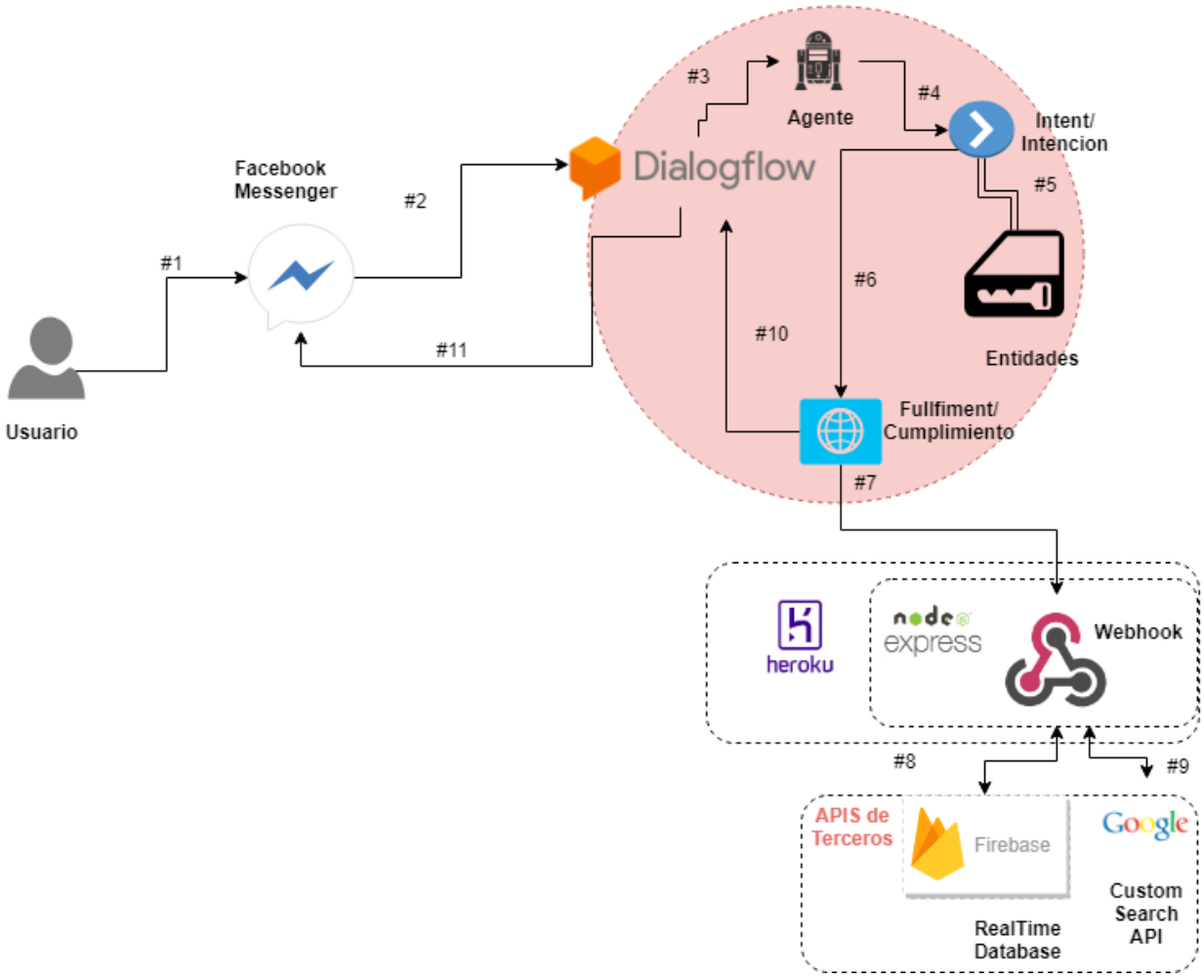


Figura 22. Fulfillment en Dialogflow.
Fuente: Autor 2018.

2.3.2.2 Casos de uso

Para el caso de uso F3 – "Buscar repuestos" se hace referencia al diagrama de actividades ilustrado en la Figura 23.

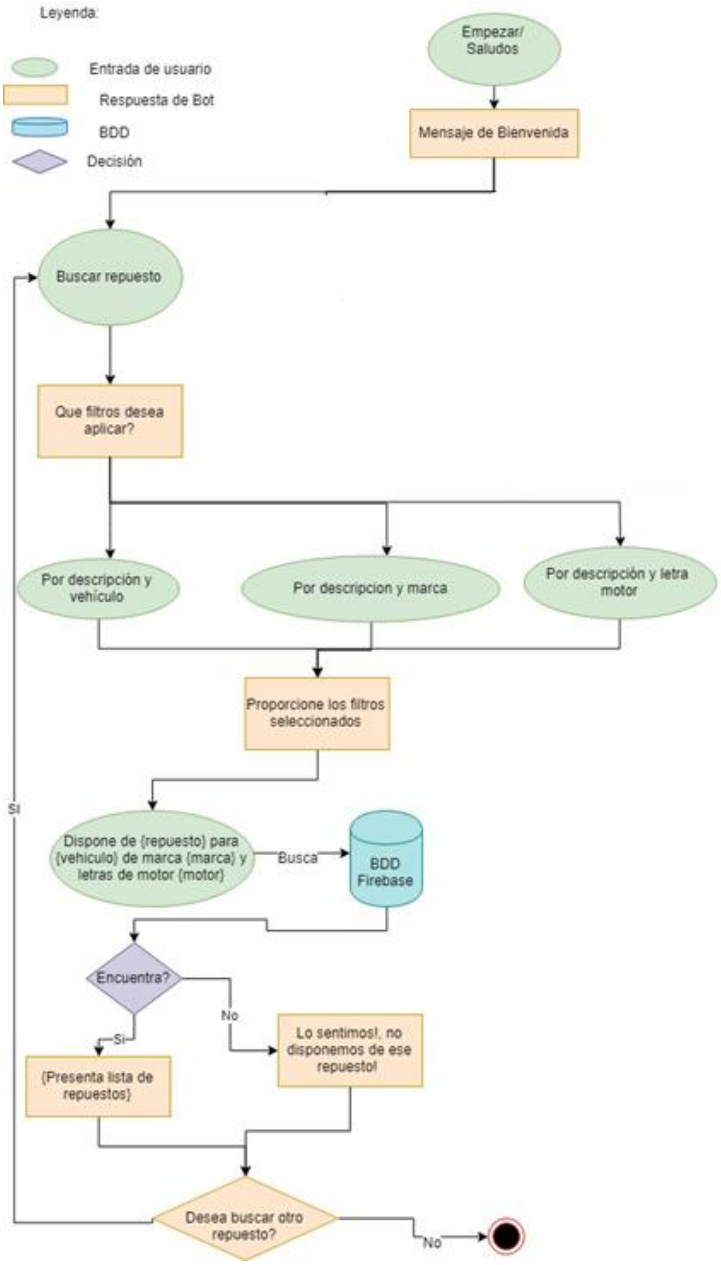


Figura 23. Flujo proceso – "Buscar repuesto".
Fuente: Autor 2018.

Diagrama a detalle F2 “Buscar repuesto”.

Descripción: Permite al usuario buscar un repuesto. Esta Intención puede ser llamada desde el menú expuesto en la Intención de “Bienvenida” o con la Intención “askRepuesto”.

El sistema desplegará en elementos de tipo “lista” con visualización tipo carta que muestra una imagen de los repuestos encontrados, si así es el caso.

La búsqueda se efectúa por filtros, que son tres por:

- “Descripción y Vehículo”
- “Descripción y Marca”
- “Descripción y Motor”

Descripción hace referencia a los repuestos. Ejemplo: “Frenos” o “Barra estabilizadora”.

Marca de repuestos: Ejemplo: “KENDALL”.

Motor hace referencia a letra de motor: Ejemplo: “ABE”.

Actores: Usuario, Messenger, Dialogflow, Webhook, Firebase RealTime Database, Google Custom Search API.

Pasos del flujo principal:

1. El usuario envía el mensaje “Buscar repuesto” o selecciona en el menú inicial “Buscar repuesto”.
2. Messenger envía el mensaje como texto. (E1, E2)
3. Dialogflow recibe el texto y se lo envía al Agente “ChatbotPrueba”.

4. Dialogflow identifica la Intención del usuario y pasa la Intención “askRepuesto” hacia el webhook.
5. El webhook envía el mensaje a Messenger (E1, E2)

   “Ok. Procedamos a la búsqueda. ¿Qué filtros desea aplicar?

   Por Descripción-Vehículo

   Por Descripción-Marca

   Por Descripción-Motor".
6. El webhook envía elementos de “quick-reply” o repuesta rápida, con las opciones:

   Opción 1. Por Descripción-Vehículo.

   Opción 2. Por Descripción-Marca.

   Opción 3. Por Descripción-Motor.
7. El usuario escoge una de esas opciones o las escribe.
8. Messenger envía el mensaje con la opción seleccionada.
9. Dialogflow recibe el texto y se lo envía al Agente “ChatbotPrueba”.
10. Dialogflow identifica la Intención del usuario y pasa la Intención “askForSearchParams” hacia el webhook.
11. El webhook envía el mensaje “Ingrese la información de los filtros seleccionados.”
12. Dependiendo de la opción de filtros el webhook envía el siguiente mensaje:

    Opción 1: 'Ejemplo: Para descripción y vehículo --> Amortiguador posterior para VK GOLF'.

    Opción 2: 'Ejemplo: Para descripción y marca --> Amortiguador delantero marca SACH'.

Opción 3: Ejemplo: Para descripción y letra de motor --> Banda de distribución con letras de motor APK'.

13. El usuario escribe su búsqueda. Ejemplo: Busco “Barra estabilizadora para Volkswagen jetta”.
14. Dialogflow recibe el texto y se lo envía al Agente “ChatbotPrueba”. (E3)
15. Dialogflow identifica la Intención del usuario y pasa los parámetros hacia el webhook.
16. El webhook determina si todos los parámetros han sido cumplidos.
17. Dialogflow pasa la Intención “findRepuesto” con sus parámetros hacia el webhook.
18. El webhook realiza una búsqueda de repuestos dentro de “Firebase Realtime Database” con los parámetros recibidos. (E4, E5)
19. El webhook verifica si los repuestos en la base de datos tienen imagen. (E4)
20. Si la búsqueda devuelve más de 1 resultado el Fullfiment en Dialogflow entrega la repuesta, como elementos tipo “lista” dentro de un objeto JSON con los 4 primeros resultados desde el webhook. Cada elemento de la lista consta de: IMAGEN, DESCRIPCION, PVP, CANTIDAD y SUCURSAL
21. El webhook envía la lista a Messenger.
22. El webhook envía la pregunta “Es este el repuesto que estaba buscando” a Messenger.
23. El usuario responde “SI” o hay una intención de follow-up “FindRepuesto-yes”.
24. El webhook envía a Messenger el siguiente mensaje: “En hora buena! ¡Ha sido un placer atenderlo! ¡Para realizar un pedido contactarse al número 099 509 0010 o al

5 115 322! .....Si desea buscar otro repuesto? Escriba "Buscar repuesto." Si desea volver al menú principal escriba "Empezar".

Pasos del flujo alterno:

16. El webhook envía mensajes para llenar los filtros y parámetros faltantes en la repuesta del usuario. Por ejemplo, si el usuario escogió la opción de filtros descripción y vehículo. Y solo ha proporcionado la descripción. “Busco barra estabilizadora” -Se envía mensajes como “Ingrese vehículo para la búsqueda”.

19. Webhook llama a “CUSTOM SEARCH API” de Google, realizando una búsqueda con el código obtenido de los repuestos y actualiza la base de datos con las URLs de las imágenes recibidas. (E6)

20. Webhook devuelve un mensaje tipo “card” dentro de un objeto JSON con los siguientes atributos del repuesto: IMAGEN, DESCRIPCION, PVP, CANTIDAD y SUCURSAL.

23. (Usuario responde “NO”) o Dialogflow tienen un follow “FindRepuesto-no”. Dialogflow envía a Messenger “Lo siento!. Si desea buscar otro repuesto. Escriba "Buscar repuesto." Si desea volver al menú principal escriba "Empezar"”

Excepciones:

E1: Razón: No se detecta internet.

Acción: Messenger cambia de estado a mensaje no enviado.

E2: Razón: API Facebook Messenger falla.

Acción: Messenger cambia de estado a mensaje no enviado.

E3: Razón: API client de Dialogflow falla.

Acción: No se envía nada.

E4: Razón: Firebase falla.

Acción: El webhook envía el mensaje: “No se ha podido realizar la búsqueda, intente más tarde”.

E5: Razón: No se encuentran repuestos con los parámetros enviados.

Acción: El webhook envía el mensaje: 'Lo sentimos no disponemos con ese repuesto por el momento. Se procederá a su revisión por un empleado.'

E6: Razón: API CUSTOM Search de Google ha cumplido su cuota máxima diaria.

Acción: Se espera hasta el siguiente día para hacer otra consulta a la API.

3 Desarrollo de prototipo

En el presente capítulo se hace énfasis en el trabajo que se realiza tanto en la plataforma Dialogflow y en los webhook, descritos en el capítulo anterior, para la implementación del chatbot. También se realizan pruebas de las funcionalidades especificadas en cada iteración de PSP propuesta.

En cada iteración de PSP es necesario la etapa de postmortem como punto final de cada ciclo. Dentro del postmortem se espera la evaluación del proceso de calidad, en donde se detallan los inconvenientes para cumplir las metas y se determina cuales metas se alcanzaron y aquellas que no.

3.1 Implementación

3.1.1 Nivel PSP0

A continuación, se describe la implementación de las funcionalidades mencionadas anteriormente en la planificación, tanto en las plataformas de Firebase como Dialoglow.

3.1.1.1 Firebase

Para crear una base de datos en tiempo real se debe crear un proyecto en Firebase. El nombre del proyecto es Chatbot con un plan spark o gratuito.

Por motivos de seguridad se ocultarán las WEB API KEY y otras credenciales dentro de la configuración de Firebase RealTimeDatabase.

El proyecto tiene la configuración general que se visualiza en la Figura 24. No posee integraciones con otras plataformas.

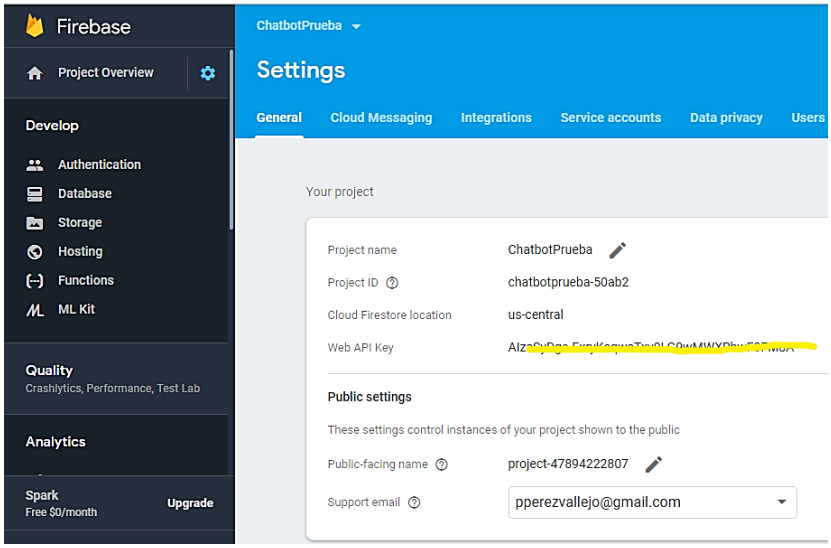


Figura 24. Proyecto Firebase Configuración General.
Fuente: Autor 2018.

Tras realizar la normalización de data y obtener un total de 3173 diferentes repuestos estructurados en el archivo “inventario.json” JSON se procede a la importación como se evidencia en la Figura 25.

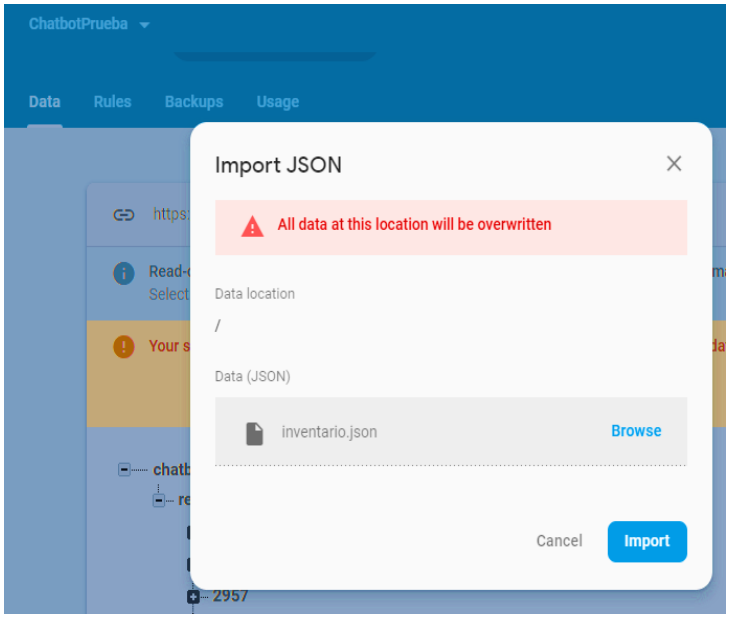


Figura 25. “inventario.json”.
Fuente: Autor 2018.

En la Figura 26, se evidencia la carga exitosa de la importación y posterior a la misma se efectúan las reglas de escritura y lectura de la base de datos.

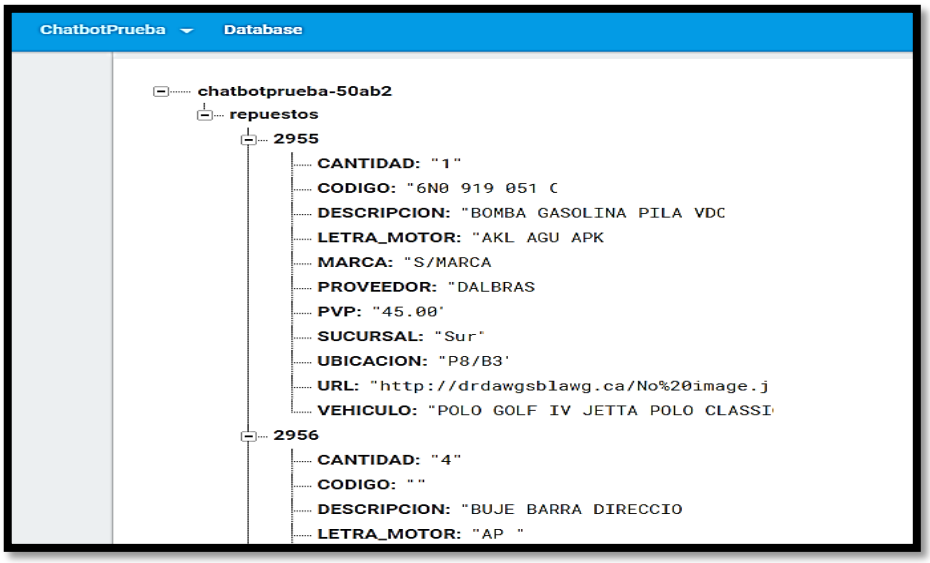


Figura 26. RealTime Database" Inventario Mundirepuestos.
Fuente: Autor 2018.

3.1.1.2 Intenciones en Dialogflow

Dentro de Dialogflow se crea un Agente "ChatbotPrueba" sin ningún template por defecto de agentes, con las siguientes características. Como se puede ver en la Figura 27, se ha habilitado el lenguaje español general con opciones de Machine Learning (ML) debido a la deprecación de la versión V1 del API de Dialogflow, se ha usado la versión V2.

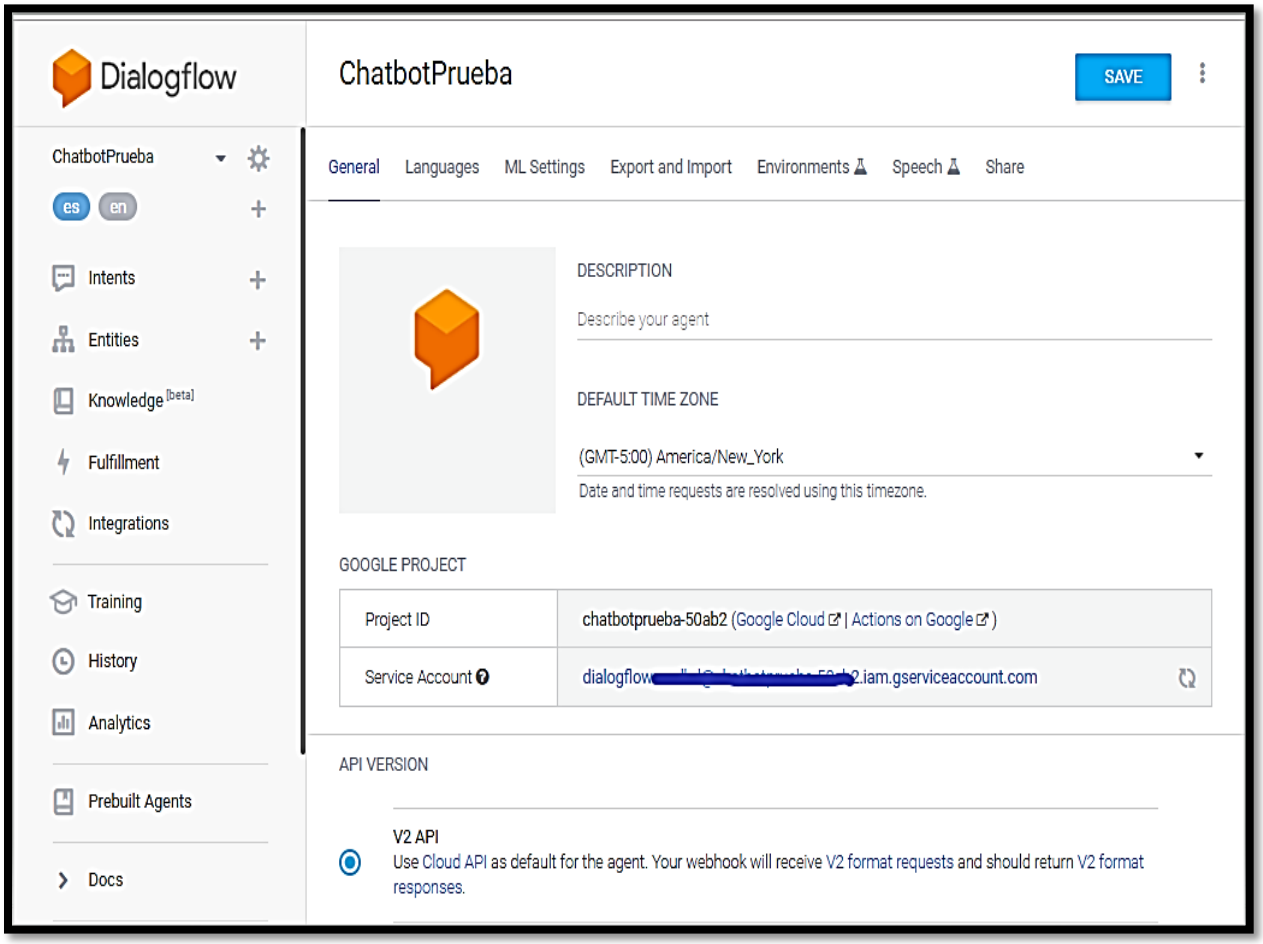


Figura 27. Dialogflow configuración general "Chatbot Prueba".
Fuente: Autor 2018.

Intención de Bienvenida

Para empezar, se crea una Intención de "Bienvenida por defecto", con la siguiente configuración. Como se observa en la Figura 28, esta intención no tendrá contextos, pero si eventos que la disparen como lo son "Welcome" y "Facebook Welcome".

Frases de entrenamiento son: "Empezar", "Hola", "Buenos días", etc.

Acción: input.welcome

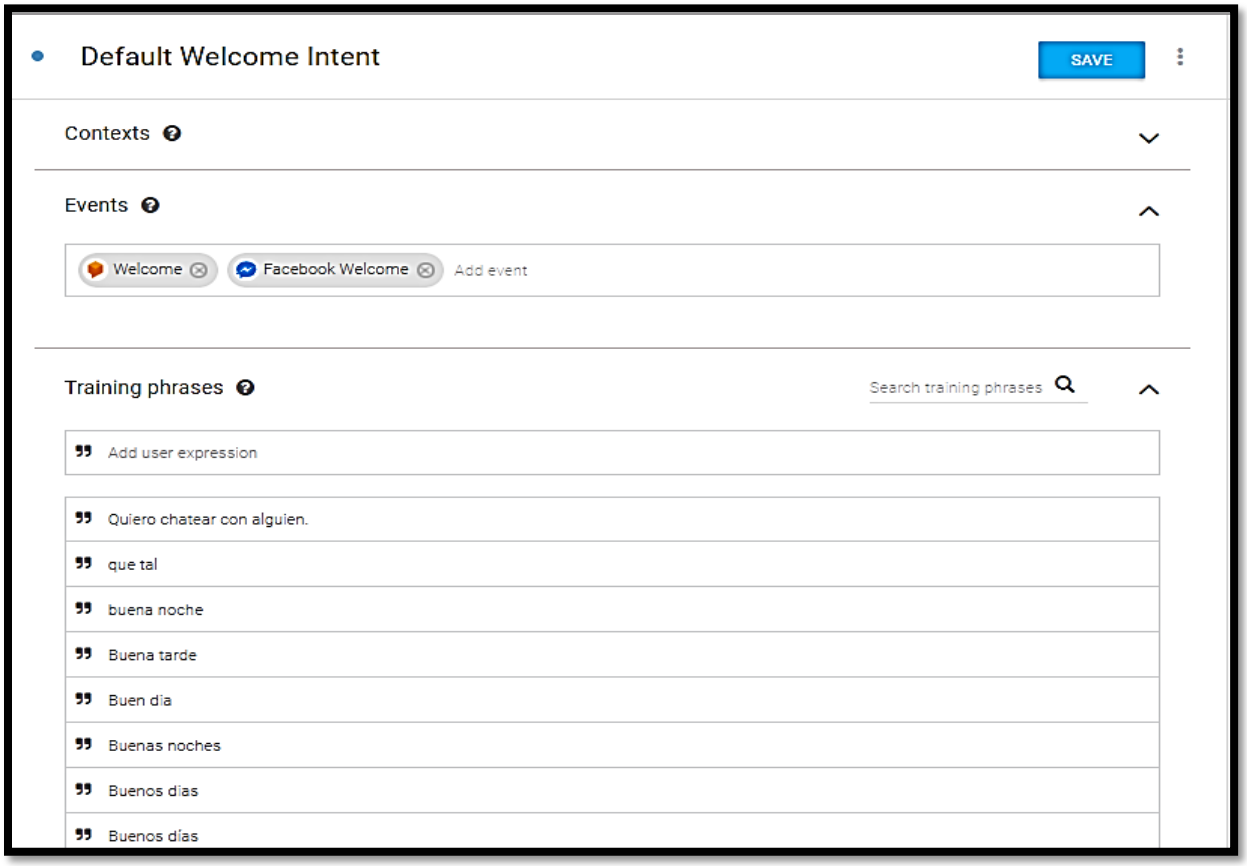


Figura 28. Dialogflow Intención "Default Welcome Intent" 1.
Fuente: Autor 2018.

Como se observa en la Figura 29, la respuesta que se configura es una repuesta rápida o "quick replies" con la configuración que se visualiza. Pregunta "Qué puedo hacer por usted?"

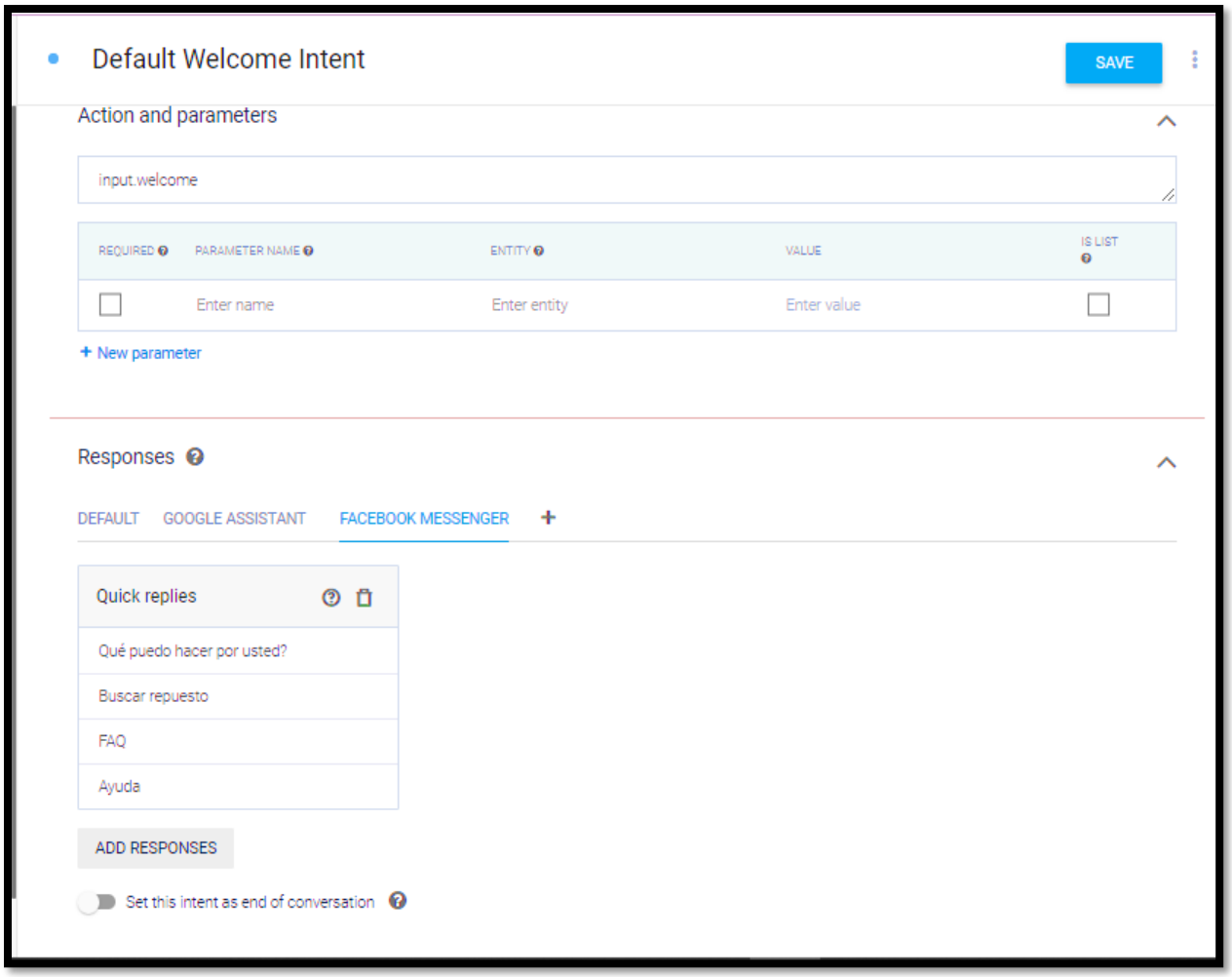


Figura 29. Dialogflow Intención "Default Welcome Intent" 1.
Fuente Autor 2018.

Intención de "askFAQ"

Dicha Intención no posee contextos ni eventos de disparo. Sus frases de entrenamiento son las siguientes: "Preguntas frecuentes" y "faq". Cabe recalcar que estas frases no son CASE SESITIVE, como todas las que se registren en esta sección para demás Intenciones.

Como acción se envía la frase “askFAQ”. La configuración se puede observar en la Figura 30.

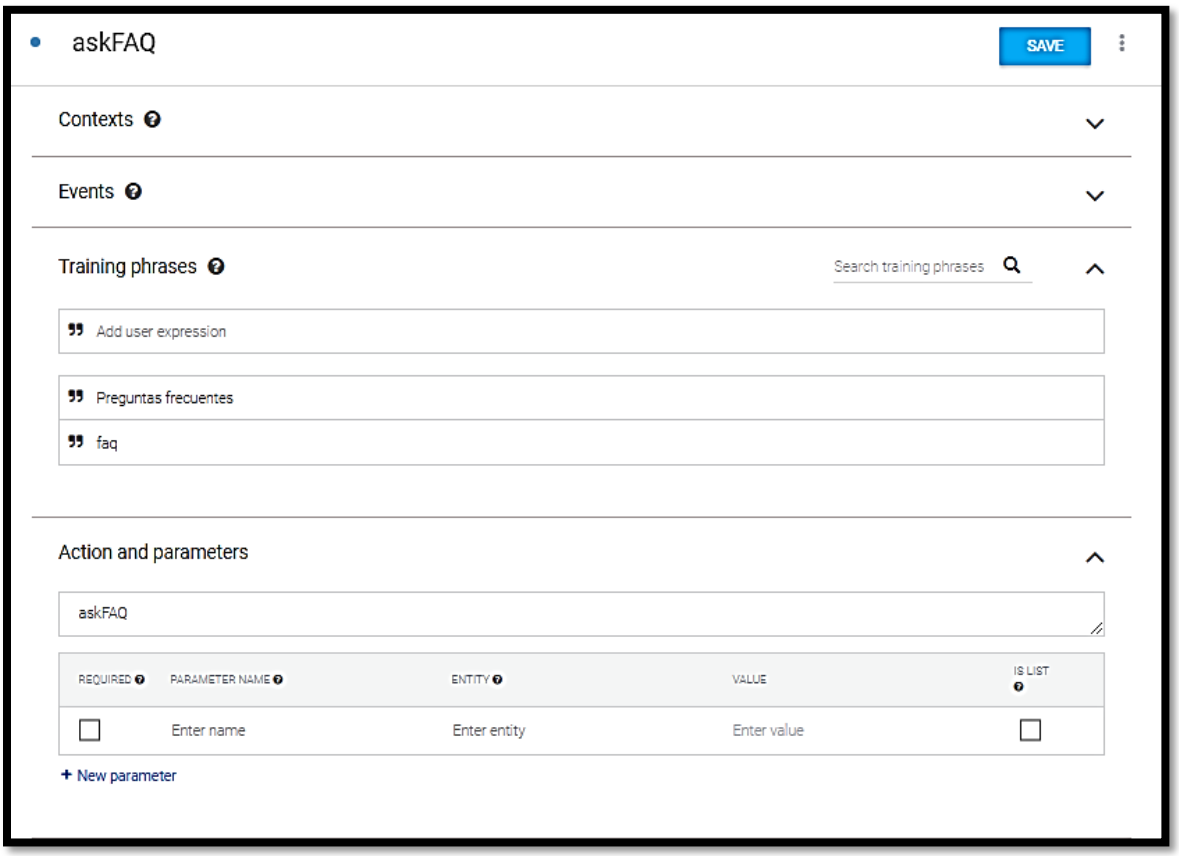


Figura 30. Dialogflow Intención “askFAQ”.
Fuente: Autor 2018.

A diferencia de la intención “Default Welcome”, hay que configurar en la integración de Facebook un “custom payload”, con mensajes de tipo “Lista”, para las repuestas como se muestra en la Figura 31 y 32. Hay que tener en cuenta que se si existen más de una repuesta configurada, se escogen de forma aleatoria antes del envió hacia las determinadas plataformas.

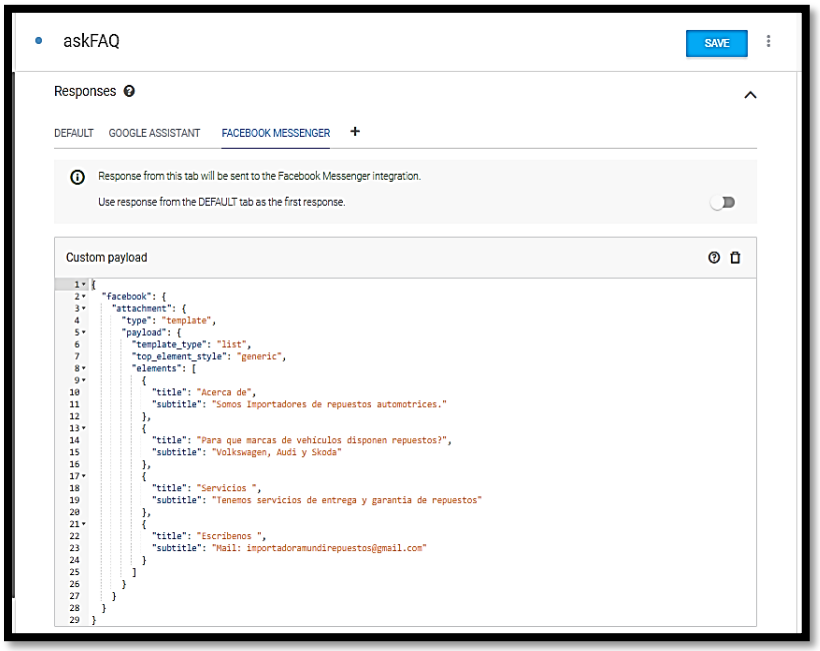


Figura 31. Dialogflow Intención "askFAQ" custom payload 1.
Fuente: Autor 2018.

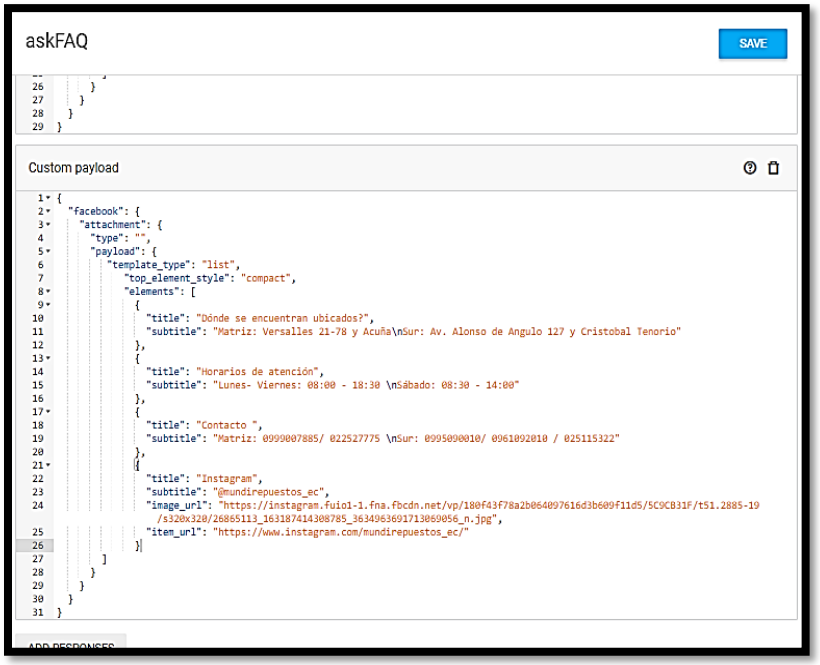


Figura 32. Dialogflow Intención "askFAQ" custom payload 2.
Fuente: Autor 2018.

Para las demás Intenciones individuales de Preguntas frecuentes; como: “askContacto”, “askCorreo”, “askHorario”, “askInstagram”, “askMarcas”, “askServicios”, ninguna de ellas posee contextos de entrada o salida, pero si con acciones con sus mismos nombres. Su respuesta es un mensaje de Texto que se configura en la pestaña. “DEFAULT”, pero dentro de la pestaña de Facebook Messenger en respuestas, se puede utilizar a la respuesta por defecto como primera respuesta, como se muestra en la Figura 33 y 34.

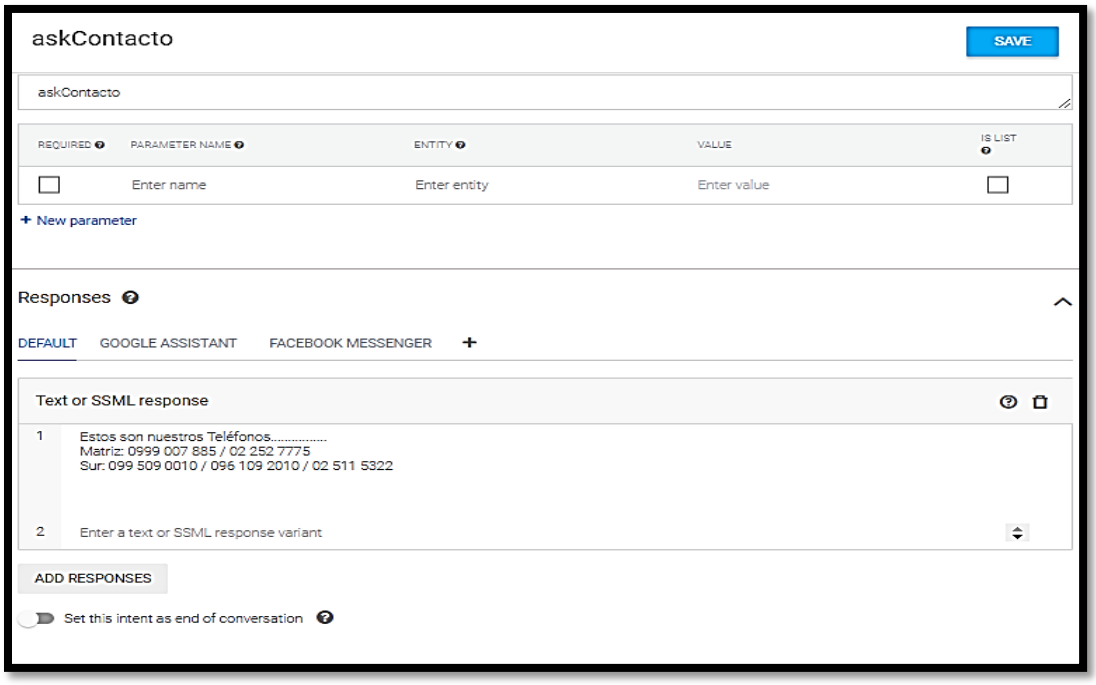


Figura 33. Dialogflow Intención “askContacto” 1.
Fuente: Autor 2018.

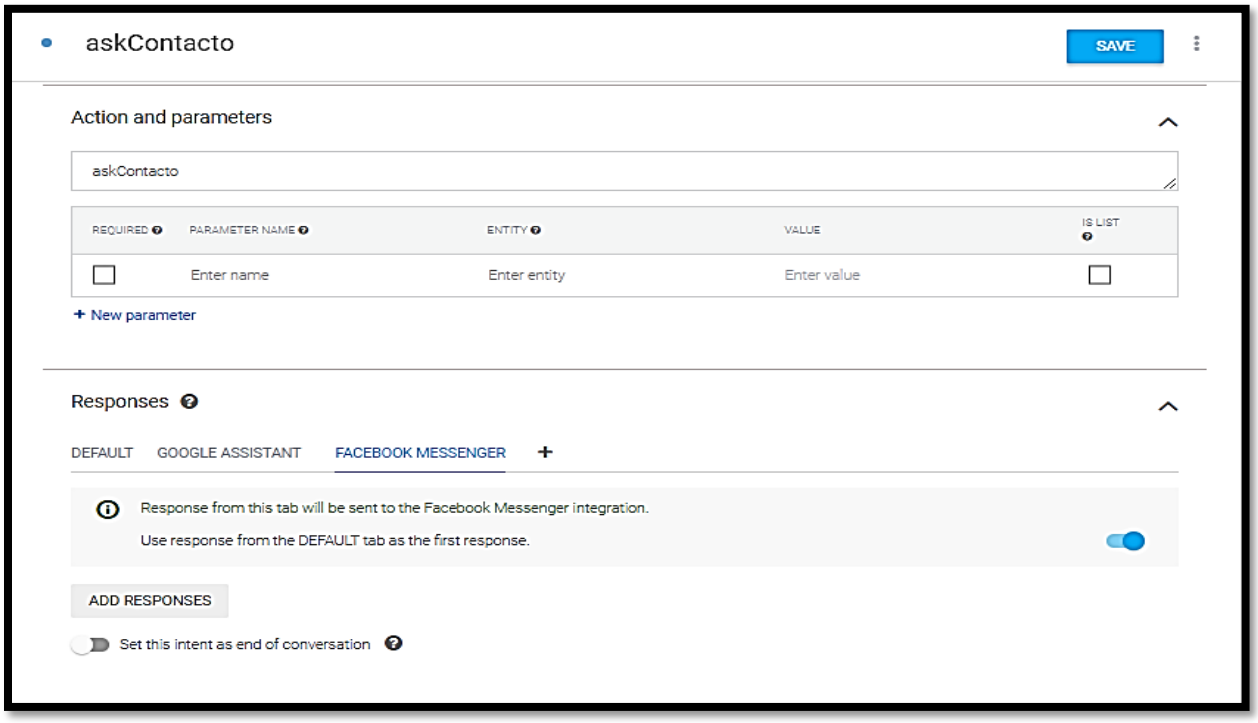


Figura 34. Dialogflow Intención “askContacto” 2.
Fuente: Autor 2018.

Si se enfoca en la intención individual “askUbicacion”, la diferencia radica en que su respuesta es un “Custom Payload”, en el cual el mensaje es de tipo lista y sus elementos son de tipo “card”, con imágenes y redirecciones URL específicas como se muestra en la Figura 35.

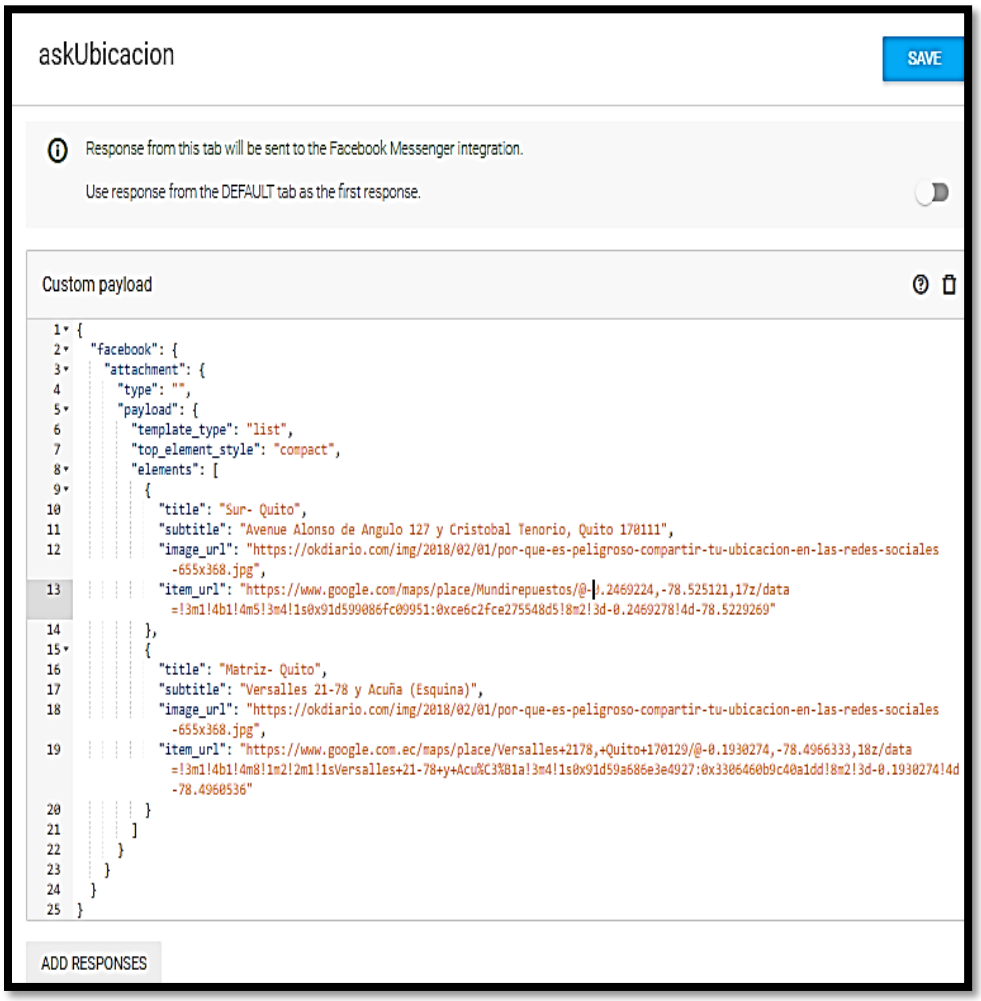


Figura *35.* Intención “askUbicacion” custom payload.
Fuente: Autor 2018.

3.1.1.3 “One-click Integration”

Dentro del panel de aplicaciones de desarrollador de Facebook, se crea una aplicación con el nombre “MundirepuestosChatbot” con las configuraciones básicas que se muestran en la Figura 36.

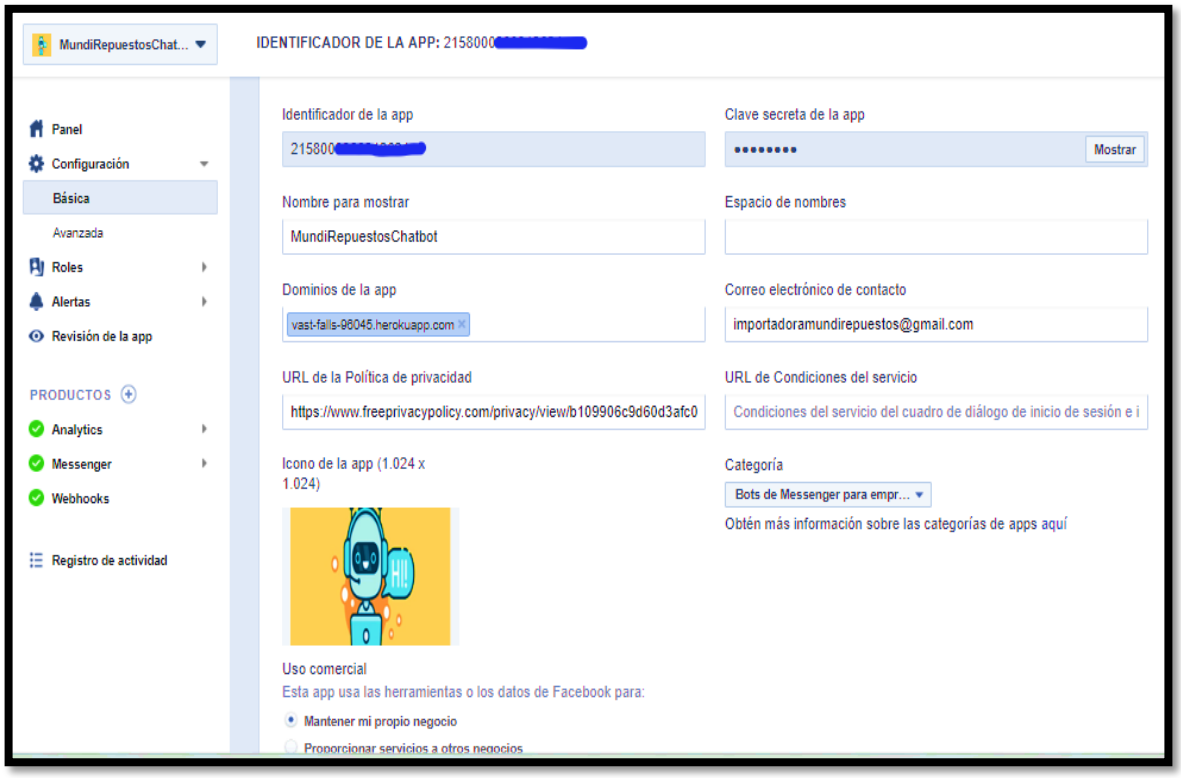


Figura 36. Facebook Developer App – Configuración básica.
Fuente: Autor 2018.

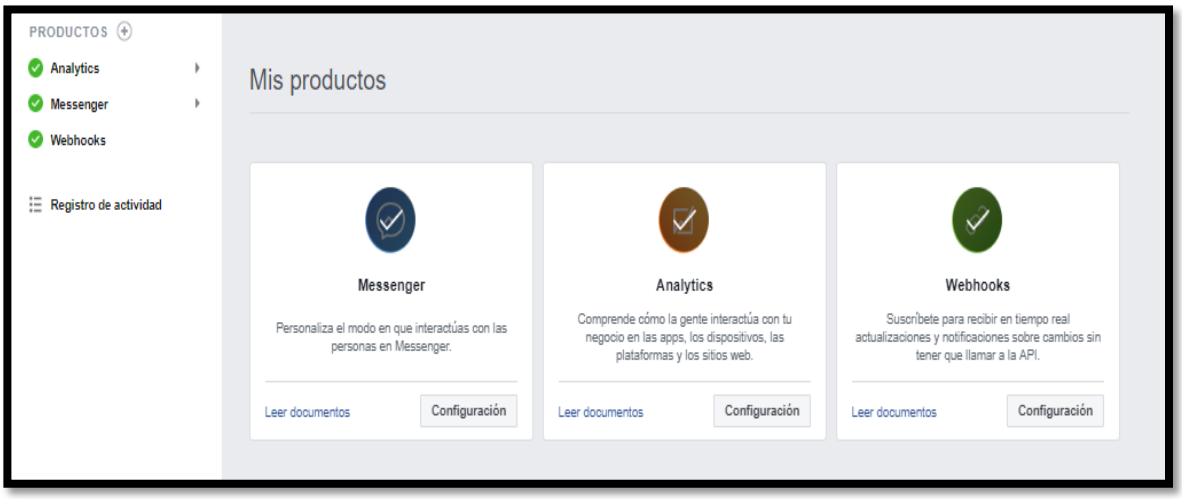


Figura 37. Facebook Developer App – Productos.
Fuente: Autor 2018.

El dato más importante, es el que se genera un token de acceso a la página pública de Mundirepuestos. Cabe destacar que los Webhook habilitados mandatorios para el correcto funcionamiento del chatbot son “message” y messaging_postbacks”, como se muestra en la Figura 38.

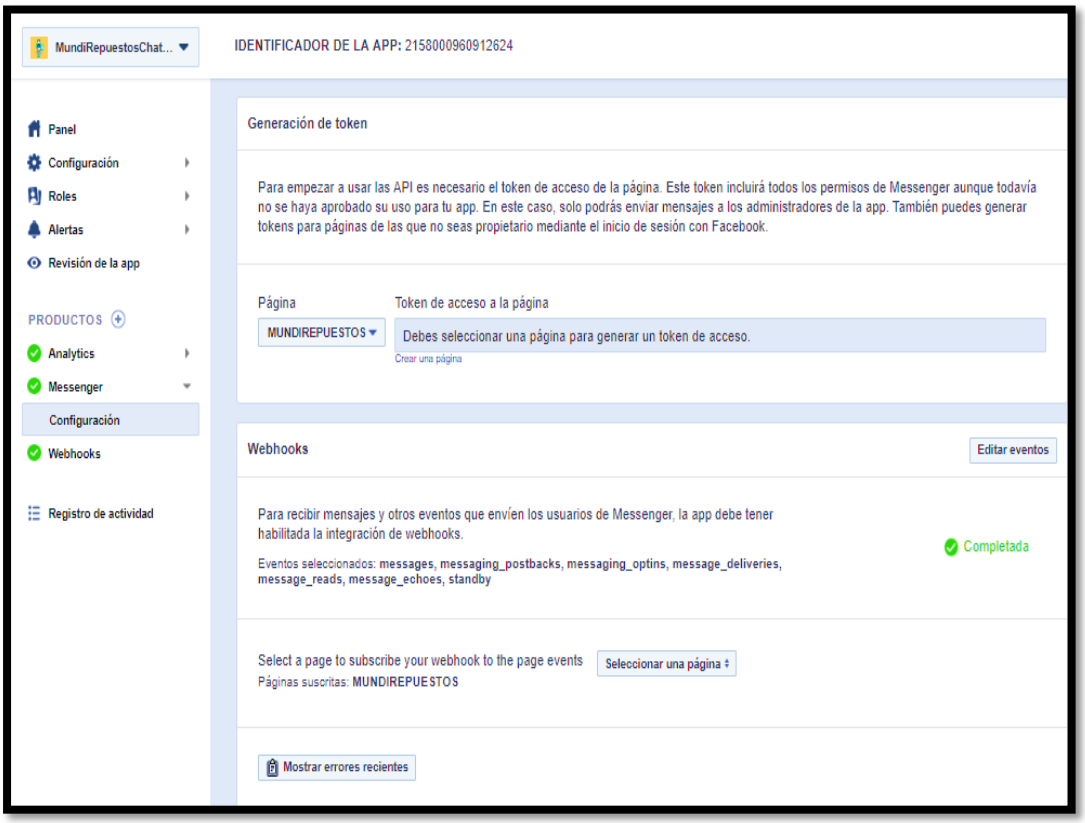


Figura 38. Facebook Developer App – Configuración de Messenger.
Fuente: Autor 2018.

El token de acceso a la página que se obtuvo anteriormente, se coloca en Dialogflow en la sección de “Integrations”. Esta información es muy sensible por eso se censura. Pero el dato que va a copiar es la URL de callback, ver Figura 39.

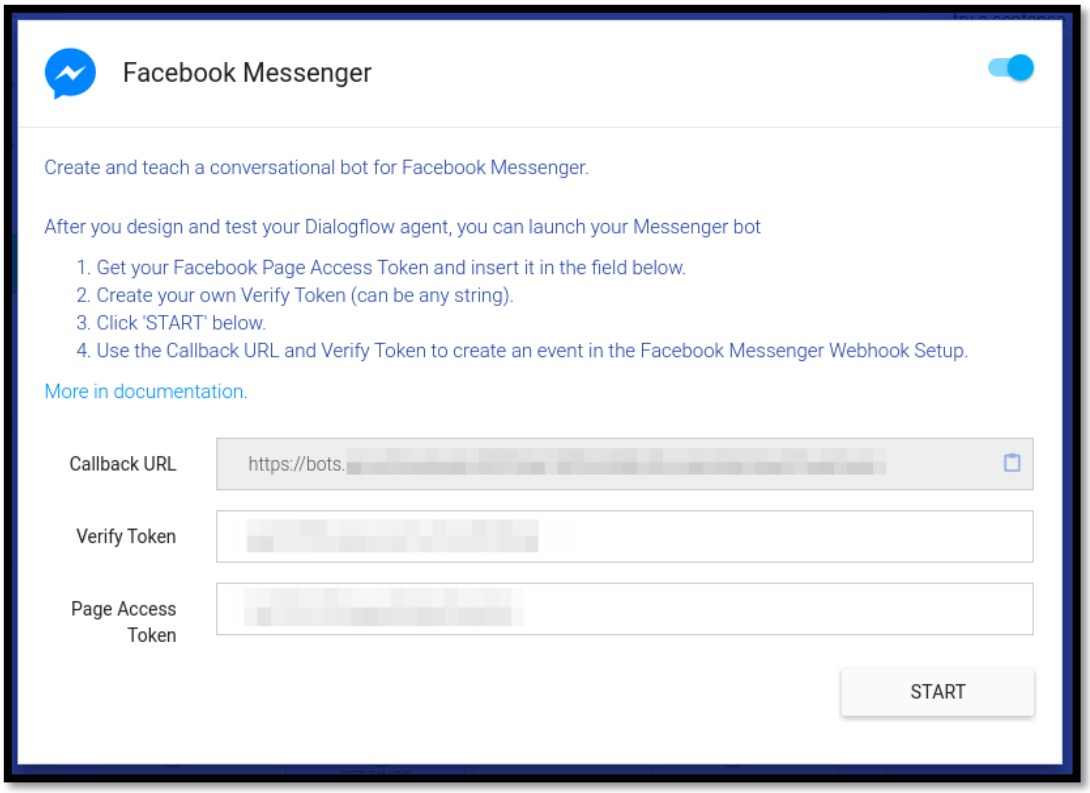


Figura 39. Dialogflow – Facebook "one-click-integration".
Fuente: Autor 2018.

Con la URL de callback que generó automáticamente Dialogflow se puede ir a la opción del menú de "Fullfilment" y en la sección de Webhook registrar la misma, como se muestra en la Figura 40.

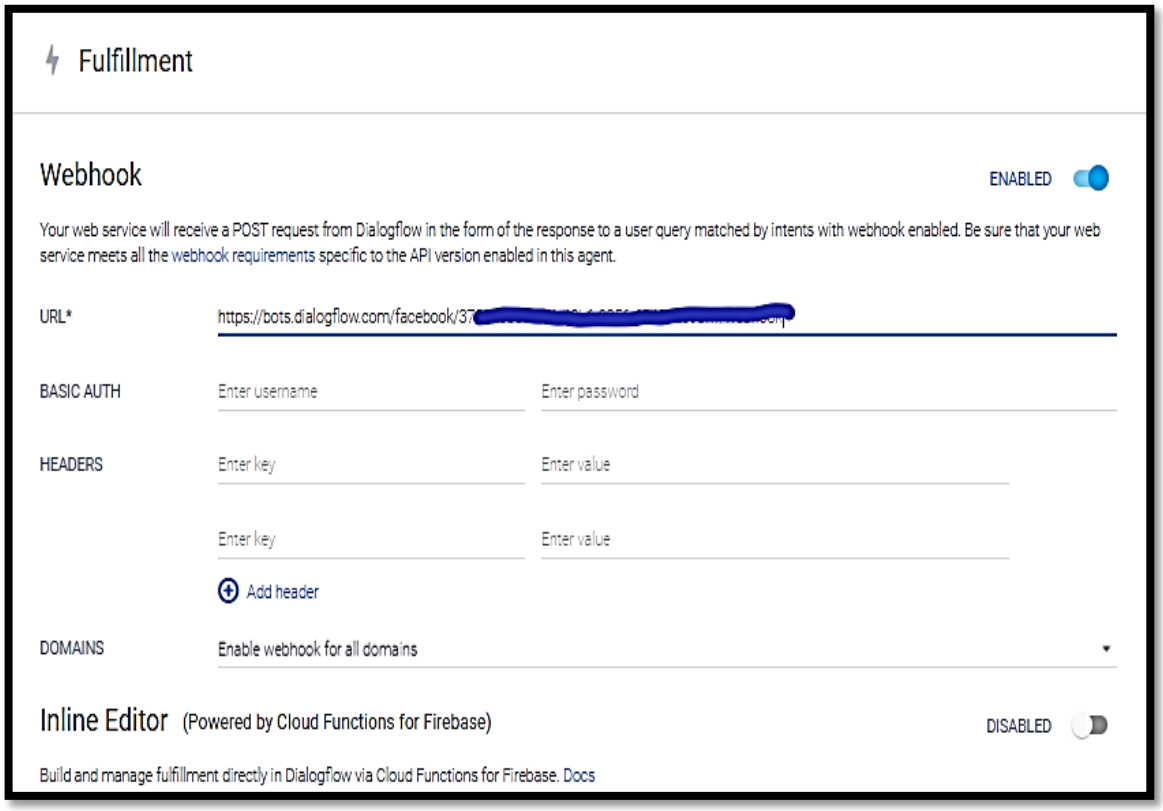


Figura 40. Configuración de One Click Integration Webhook.
Fuente: Autor 2018.

3.1.1.4 Entidades

Para el entrenamiento del chatbot las entidades a crearse son las siguientes: filtro, marca, motor, repuesto y vehículo. En la creación de entidades se puede especificar el valor de referencia con sus respectivos lo que sirven para la construcción de Intenciones cuando se necesiten parámetros. En la Figura 41, se observa lo descrito anteriormente para la entidad “repuesto”. Se puede cambiar el tipo de visualización a “RAW” y se puede subir un archivo CSV o JSON para entrenar las entidades.

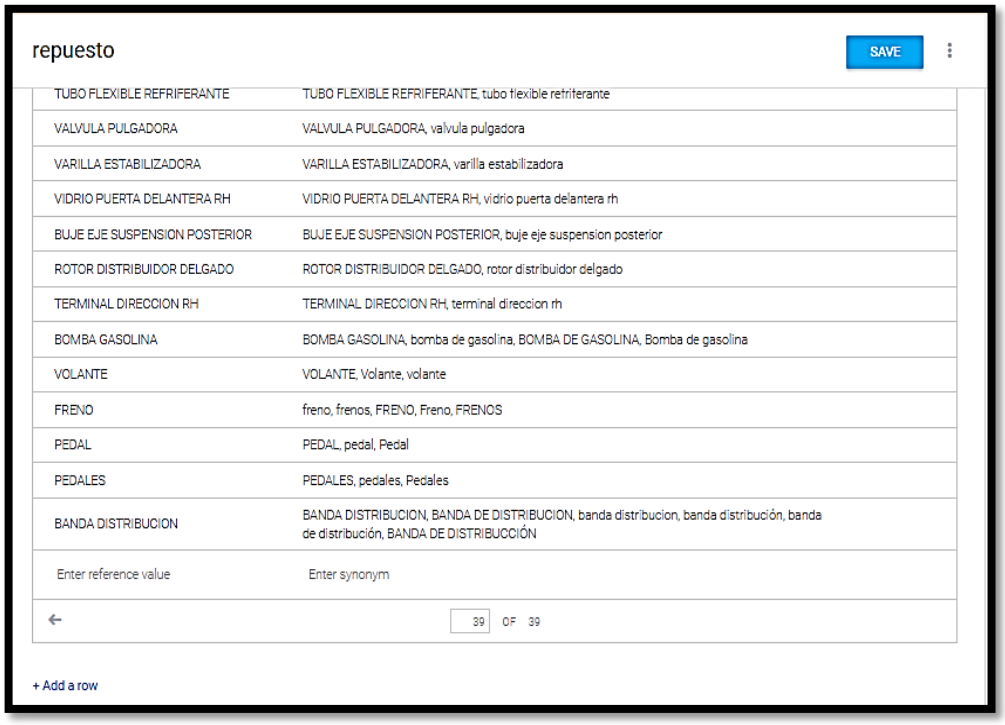


Figura 41. Entidad - Repuesto.
Fuente: Autor 2018.

En la Figura 42, se observa la configuración para la entidad “filtro” para marca, vehículo, descripción y motor, con sus respectivos sinónimos.

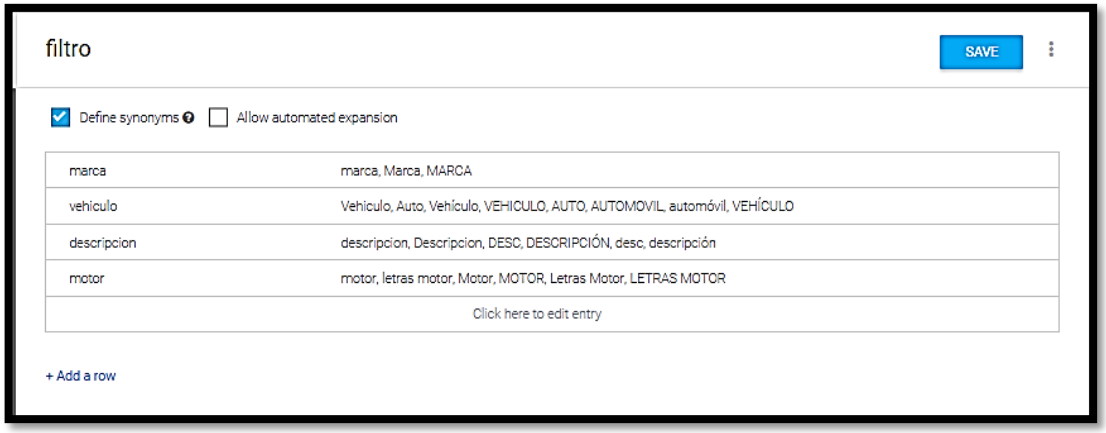


Figura 42. Entidad - Filtro.
Fuente: Autor 2018.

En la Figura 43, se visualiza parte de la configuración para la entidad “marca”, las cuales se precargan con los datos obtenidos del inventario de Mundirepuestos.

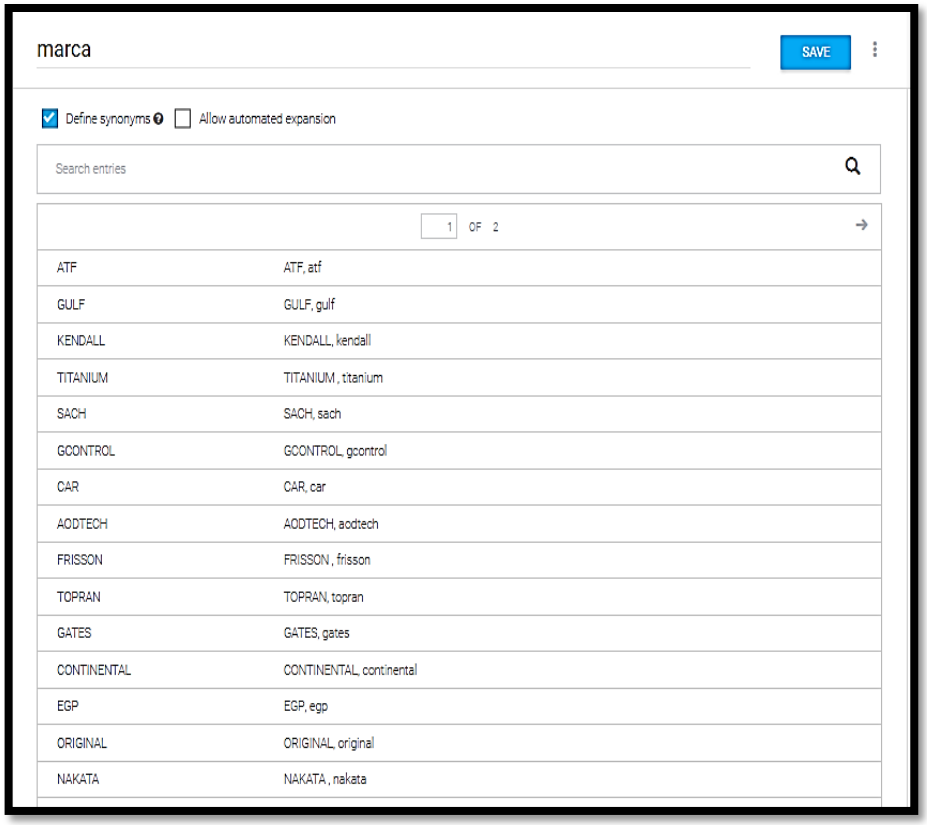


Figura 43. Entidad - Marca. Autor 2018.
Fuente: Autor 2018.

En la Figura 44, se observa la configuración para la entidad “motor”, los cuales se precargan con los datos obtenidos del inventario de Mundirepuestos con sus respectivos sinónimos.

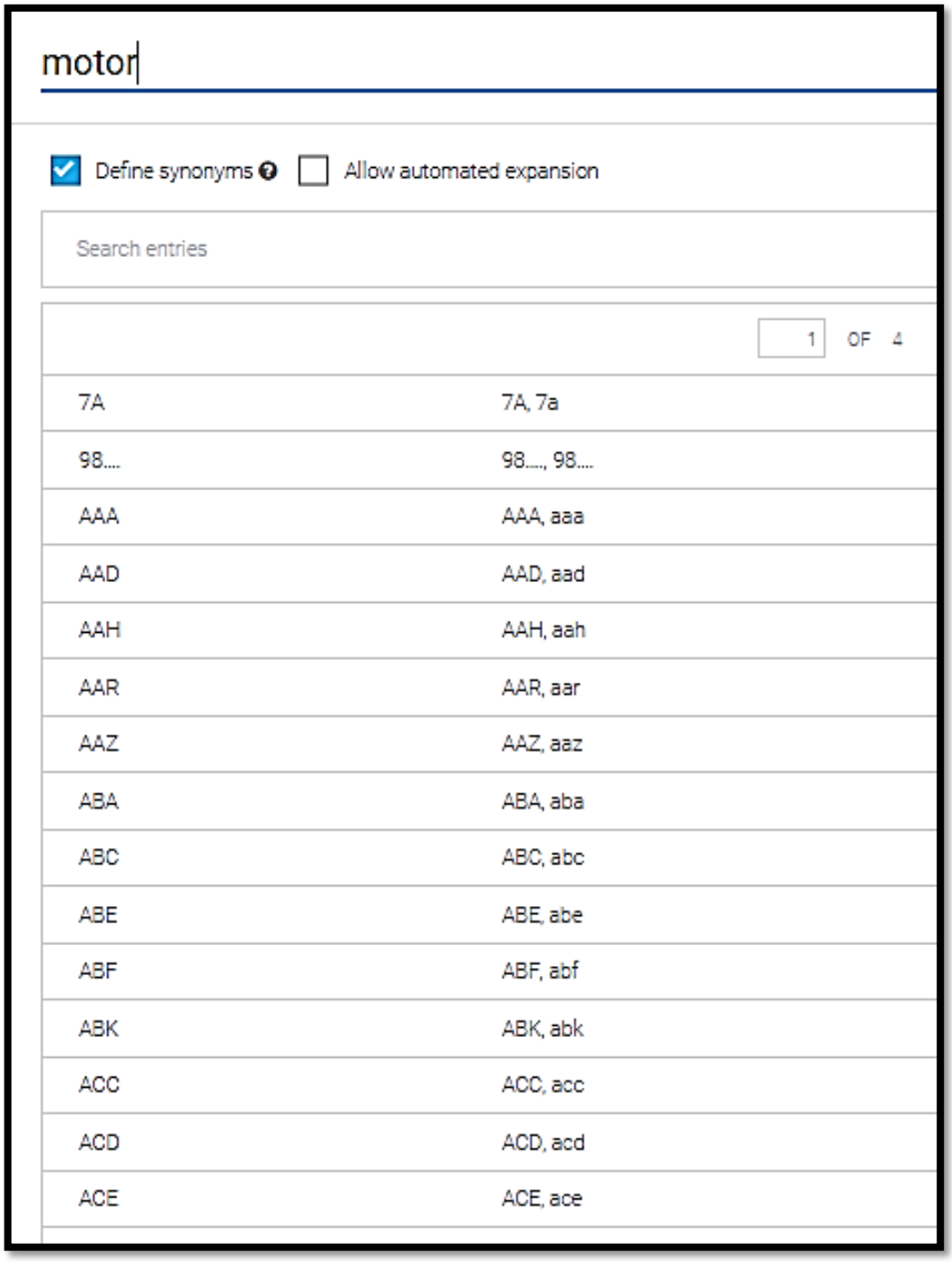


Figura 44. Entidad - Motor.
Fuente: Autor 2018.

En la Figura 45, se observa la configuración para la entidad “vehiculo”, los cuales se precargan con los datos obtenidos del inventario de Mundirepuestos con sus respectivos sinónimos.

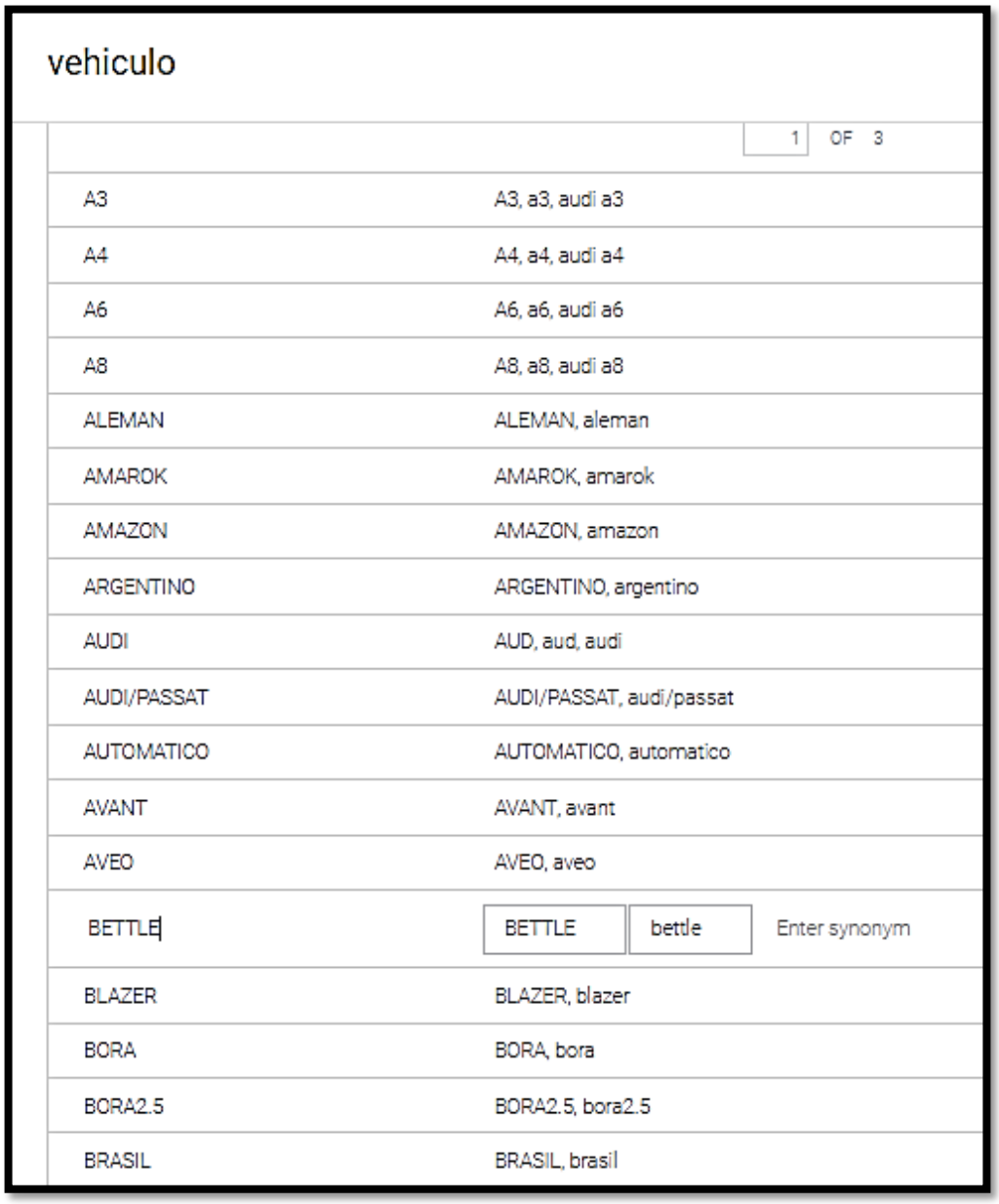


Figura 45. Entidad – Vehículo.
Fuente: Autor 2018.

3.1.2 Nivel PSP0.1

3.1.2.1 Intenciones en Dialogflow.

Intención “askRepuestos”:

Dentro de la creación del Intento se enviará un contexto de salida llamado “ask_for_search_params”. A su vez, la acción que lo dispara se llamará “askDescription”. Su repuesta es la siguiente: “Ok. Procedamos a la búsqueda. ¿Qué filtros desea aplicar?”. Las frases de entrenamiento de esta intención son las que se muestran en la Figura 46.

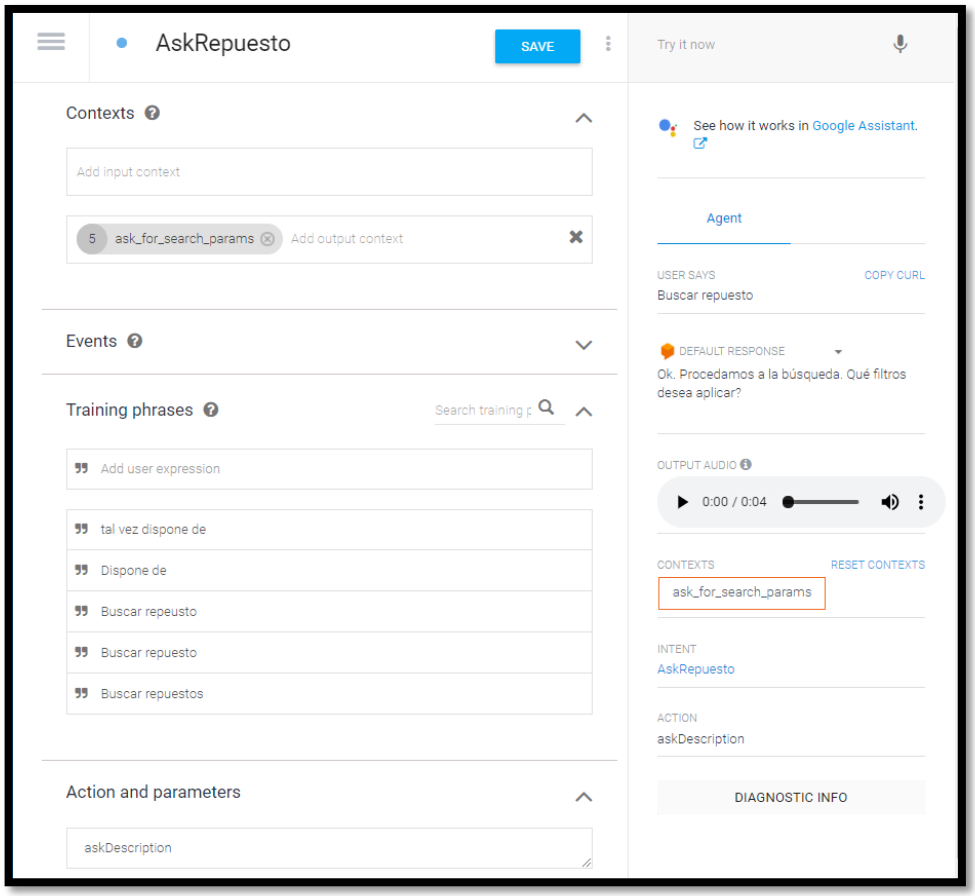


Figura 46. Intención – askRepuesto.
Fuente: Autor 2018.

Tras la Intención “askRepuesto”, la Intención “agent.askForSeachParams” se crea con las siguientes configuraciones: El contexto de entrada será “ask_for_search_params” y el contexto de salida “ask_for_description”, ver Figura 47.

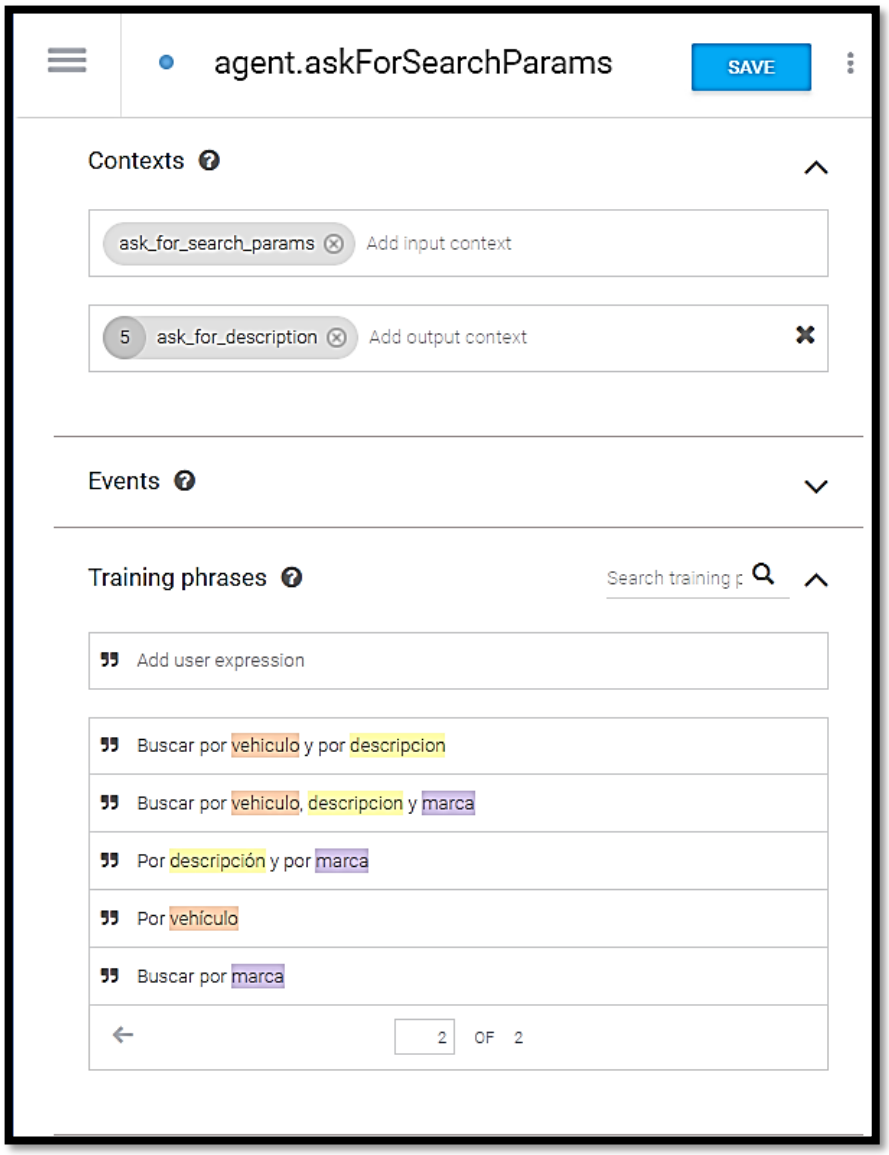


Figura 47. Intención – askForSearchParams 1.
Fuente: Autor 2018.

Sin eventos y con la Acción “askForSearchParams” y parámetros de filtros respectivos, la respuesta por defecto es “Ingrese la información de los filtros seleccionados”, ver Figura 48.

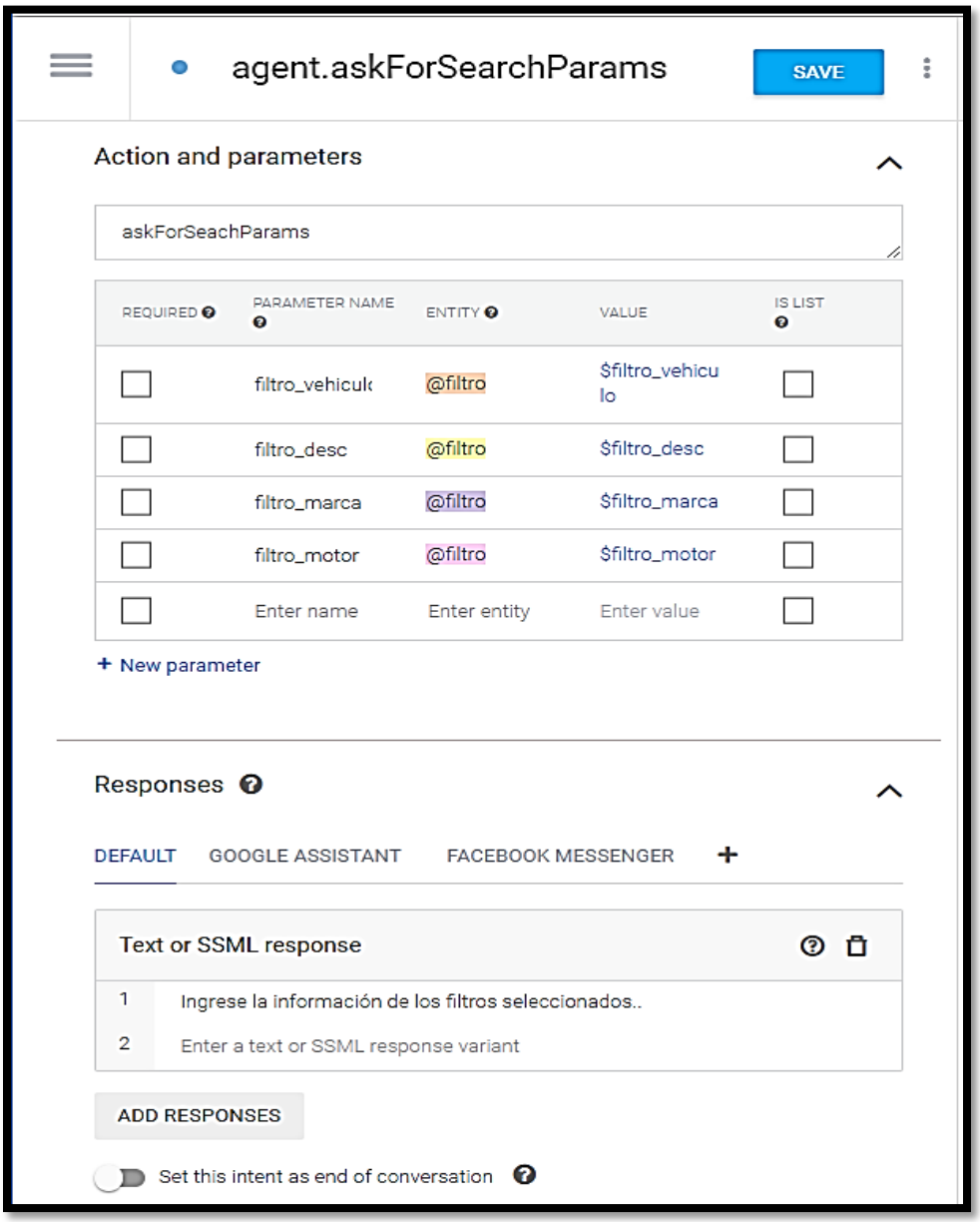


Figura 48. Intención – askForSearchParams 1.
Fuente: Autor 2018.

3.1.2.2 Webhook en NodeJS Express.

Dentro del proyecto se necesita instalar las dependencias NPM de Express junto "body-parser" para analizar los cuerpos de solicitudes entrantes.

Dentro de un archivo "index,js" en la carpeta "src" se setea a Express en el puerto 5000, ver Figura 49.

```
require('dotenv').config({ path: 'variables.env' });

const express = require('express');
const bodyParser = require('body-parser');

const app = express();

app.use(bodyParser.json());
app.use(bodyParser.urlencoded({ extended: true }));

app.listen(process.env.PORT || 5000, () => console.log('Express server is
listening on port 5000'));
```

Figura 49. Express- "index.js".
Fuente: Autor 2018.

Para configurar al webhook del chatbot, Facebook requiere de un paso de verificación que garantice que el webhook esta autenticado y funcionando antes de aceptar el mismo. Para este cometido se crea un archivo "verify-webhook.js", en el directorio "src", como se puede visualizar en la Figura 50.

```
const verifyWebhook = (req, res) => {
  let VERIFY_TOKEN = '...';

  let mode = req.query['hub.mode'];
  let token = req.query['hub.verify_token'];
  let challenge = req.query['hub.challenge'];

  if (mode && token === VERIFY_TOKEN) {
    res.status(200).send(challenge);
  } else {
      res.sendStatus(403);
    }
};

module.exports = verifyWebhook;
```

Figura 50. Express- “index.js”.
Fuente: Autor 2018.

En la Figura 51, se configura el endpoint que recibirá los eventos desde Facebook Messenger que se agrega como una petición GET en el archivo “index.js”.

```
const verifyWebhook = require('./verify-webhook');
.
.
.
app.get('/webhook/', verifyWebhook);
.
.
.
```

Figura 51. Express- Get Endpoint.
Fuente: Autor 2018.

Antes de Integrar a Dialogflow con el bot, se debe configurar la autenticación. Para hacer esto, dentro del proyecto de las configuraciones del agente, en de la sección Google Project, (ver Figura 52). Aquí se puede redirigir al panel de Google Cloud

Platform; en donde se generará un key en formato JSON para el servicio de Dialogflow, (ver Figura 53). Dentro de este archivo, se obtendrá el valor de "private_key" y "cliet_email".

Debido a la sensibilidad de estos datos es importante almacenar sus valores como variables de entorno antes de proceder. Para lograr ese objetivo se utilizó la dependencia de "dotenv", que es un paquete que sirve especialmente para eso. En el control de versión "git" que se efectúa este archivo se encuentra en .gitignore , indicando que no se tomará en cuenta en este proceso.

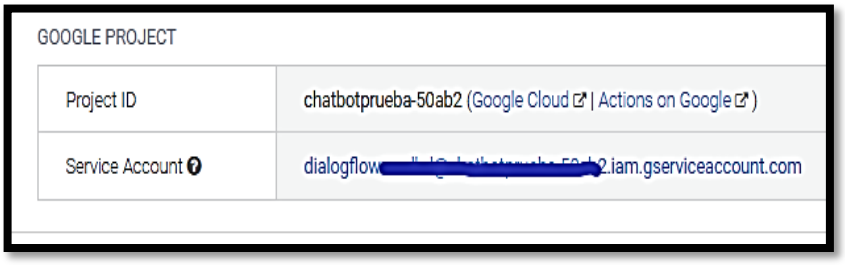


Figura 52. Dialogflow – Google Project.
Fuente: Autor 2018.

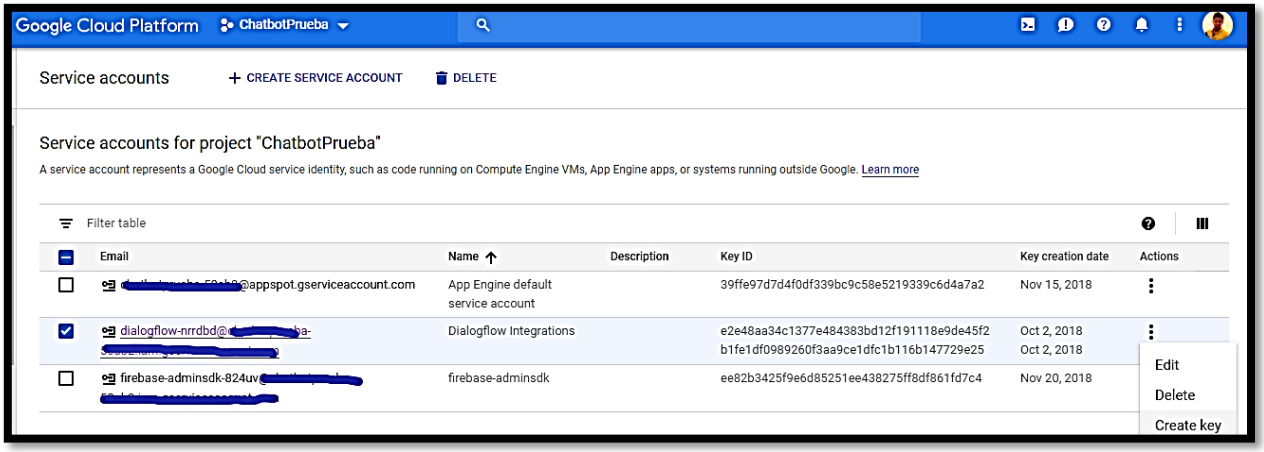


Figura 53. Google Cloud Platform – Service Keys.
Fuente: Autor 2018.

Posterior a esto, se procede a agregar el SDK que provee Dialogflow para Node.js como dependencia en el proyecto. Así también, se hace uso del paquete "node-fetch", el cual permite enviar peticiones a la API de Facebook, a su vez se instalan las dependencias 'image-search-google' y "Firebase-admin" para próxima utilización de las librerías de "Custom Search Google API" y Firebase, respectivamente.

En el archivo "process-message.js" en el directorio "src", se almacena la mayor cantidad de lógica de negocio. Aquí se encuentran las llamadas a las librerías mencionadas anteriormente, principalmente se evidencia que el texto que se recibe desde Facebook Messenger hacia Dialogflow. Después se extrae la repuesta adecuada de cada Intención de Dialogflow y enviar el resultado de vuelta a Messenger.

Cabe recalcar que dependiendo de la Intención se hacen llamada a otras funciones declaradas en el archivo "process-message", como: las de búsqueda de imágenes, búsqueda de repuestos por parámetros dentro de Firebase, entre otras.

Para enviar los eventos de mensaje desde Facebook, en el archivo "index.js" se añade lo que se muestra en la Figura 54.

```
const verifyWebhook = require('./verify-webhook');
.
.
.
app.post('/webhook/', messageWebhook);
.
.
.
```

Figura 54. Express – Post Endpoint.
Fuente: Autor 2018.

Finalmente, para procesar cada mensaje obtenido mediante Facebook en un archivo “message-webhook.js” dentro del directorio “src” se tiene la siguiente configuración, ver Figura 55.

```
const processMessage = require('./process-message');

module.exports = (req, res) => {
  if (req.body.object === 'page') {
    req.body.entry.forEach(entry => {
      entry.messaging.forEach(event => {
        if (event.message && event.message.text) {
          processMessage(event);
        }
      });
    });

    res.status(200).end();
  }
};
```

Figura 55. Express – “message-webhook.js”.
Fuente: Autor 2018.

El código completo se encuentra en un repositorio de GitHub, en el siguiente enlace: https://github.com/Elludenf/chatbot-Mundirepuestos.

3.1.2.3 Heroku.

Para concluir con el requerimiento no funcional F3 y cumplir con la arquitectura planteada, en el panel de Heroku se crea una aplicación con región en Estados Unidos, (ver Figura 56).

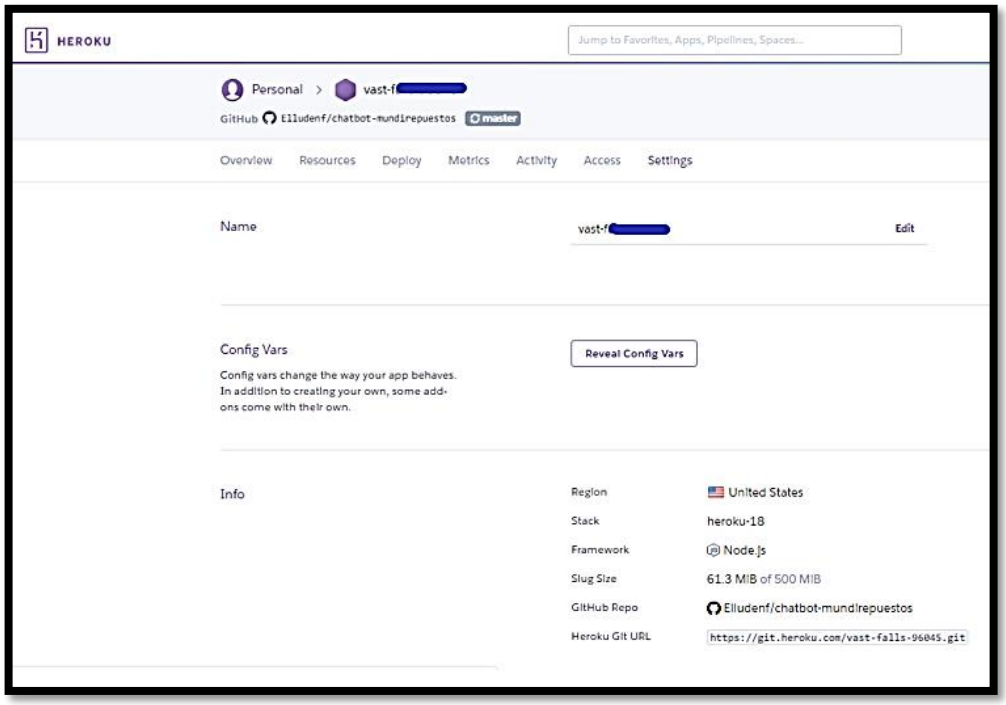


Figura 56. Configuración global de Heroku.
Fuente: Autor 2018.

En la sección de “Config Vars” o variables de configuración se ubicarán las variables de entorno que se configuraron anteriormente, (ver Figura 57).

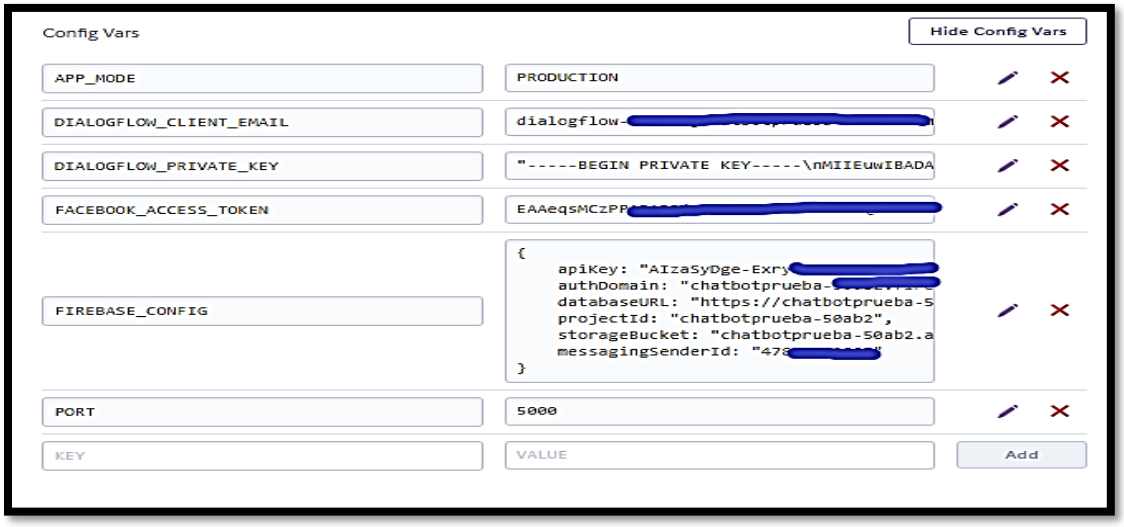


Figura 57. Heroku-variables de entorno.
Fuente: Autor 2018.

La configuración de despliegue se realiza mediante GitHub, y está habilitada la integración continua, esto quiere decir que si se realiza un “commit” en el repositorio de GitHub el ambiente de Heroku automáticamente intentará desplegarse, (ver Figura 58).

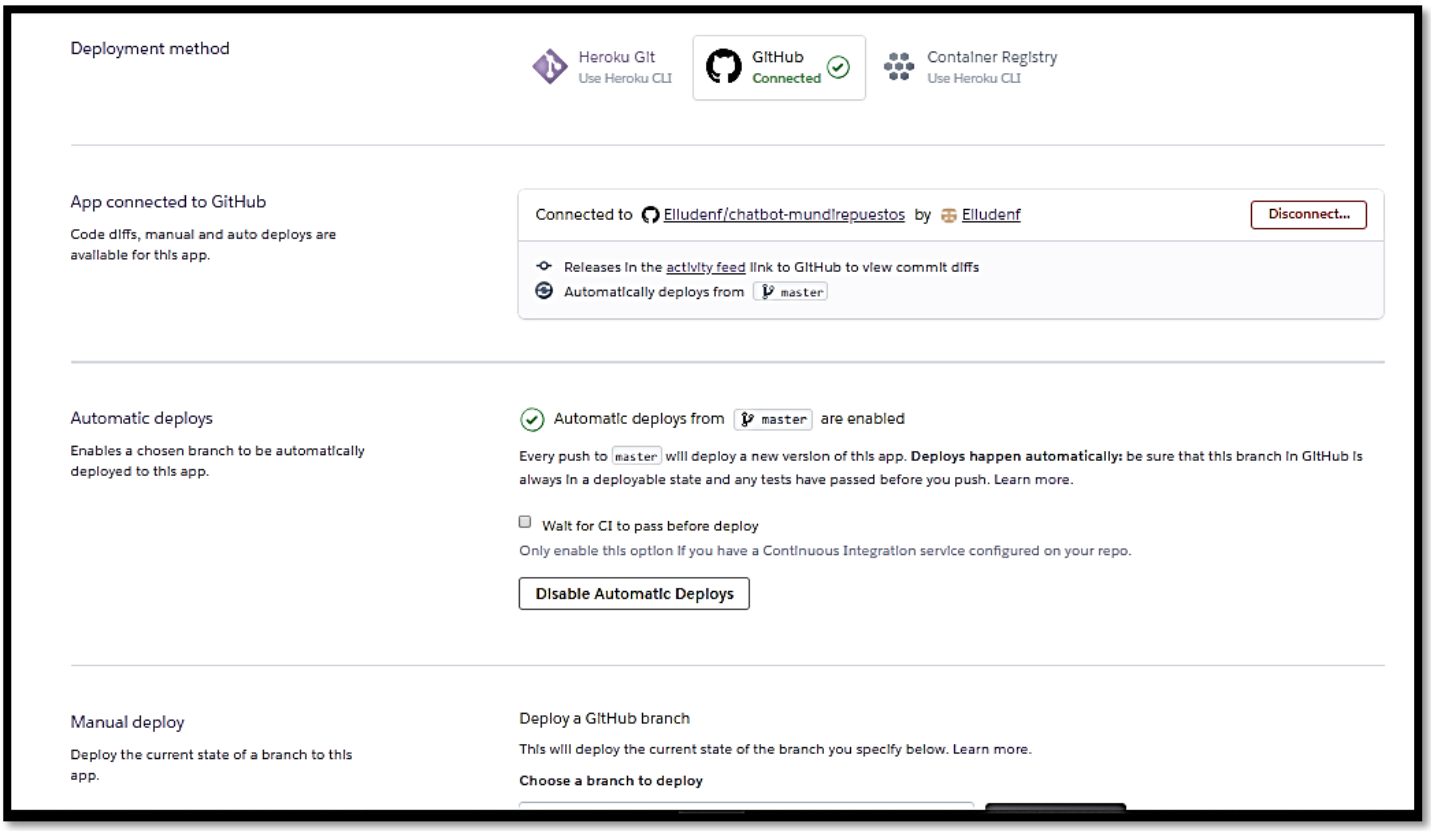


Figura 58. Heroku despliegue automático.
Fuente: Autor 2018.

Al finalizar el despliegue se visualiza que en la sección “Dominios y certificados” se genera una URL, la cual se utilizará para las configuraciones de webhook dentro de la Aplicación de desarrollador de Facebook y en las configuraciones de Fullfilment de Dialogflow como se realizó en la iteración de PSP0, (ver Figura 59).

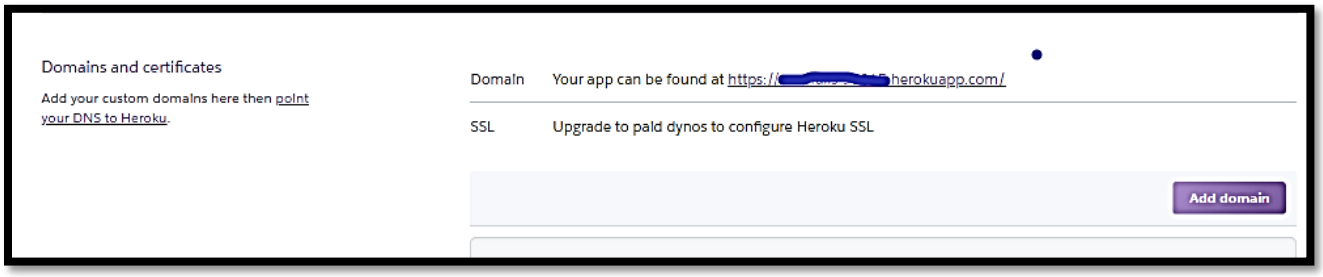


Figura 59. Heroku URL de dominio.
Fuente: Autor 2018.

3.2 Pruebas

Son las investigaciones empíricas y técnicas cuyo objetivo es proporcionar información objetiva e independiente sobre la calidad del producto a la parte interesada o stakeholder.

3.2.1 Nivel PSP0

Gracias a Dialogflow realizar pruebas es mucho más rápido, no hay de necesidad de probar la interacción con la integración a Facebook Messenger, desde Messenger, ya que se puede utilizar el emulador propio de Dialogflow.

Cuando una Intención se detecta con el input de un Usuario, se visualiza por plataforma la respuesta, así como los contextos que posee actualmente la conversación, también la Intención y Acción que se ejecutó. Así es el caso de la Intención “Default Welcome Intent”, como se puede visualizar en la Figura 60.

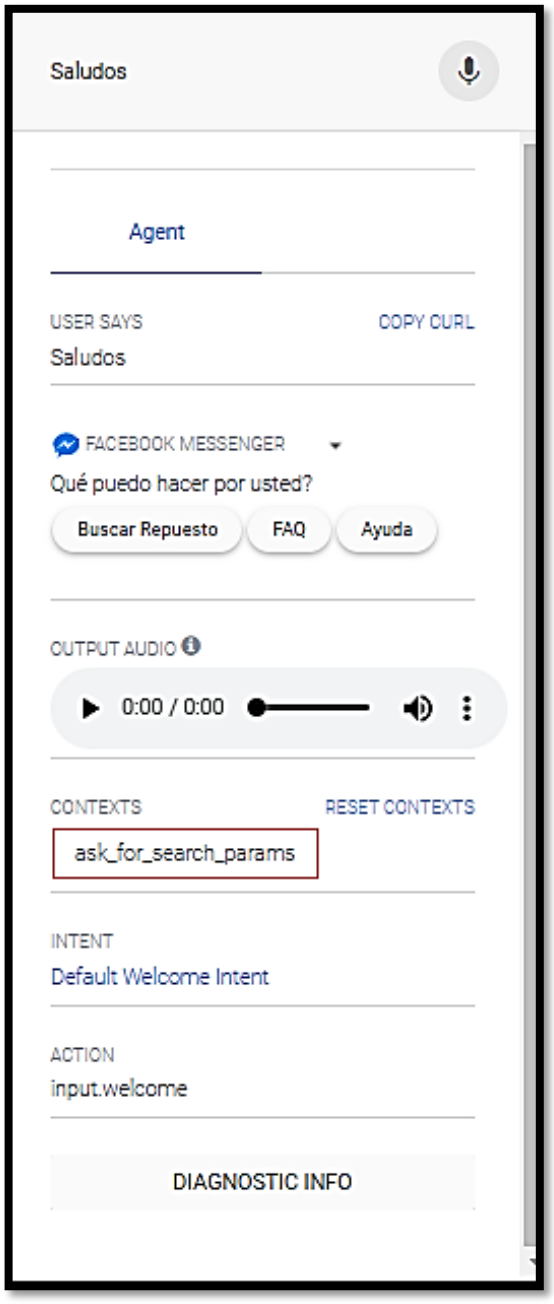


Figura 60. Emulador de conversación de Dialogflow.
Fuente: Autor 2018.

Cuando el input del usuario no se empareja con ninguna Intención, se devuelve una Intención de Fallback como en la Figura 61.

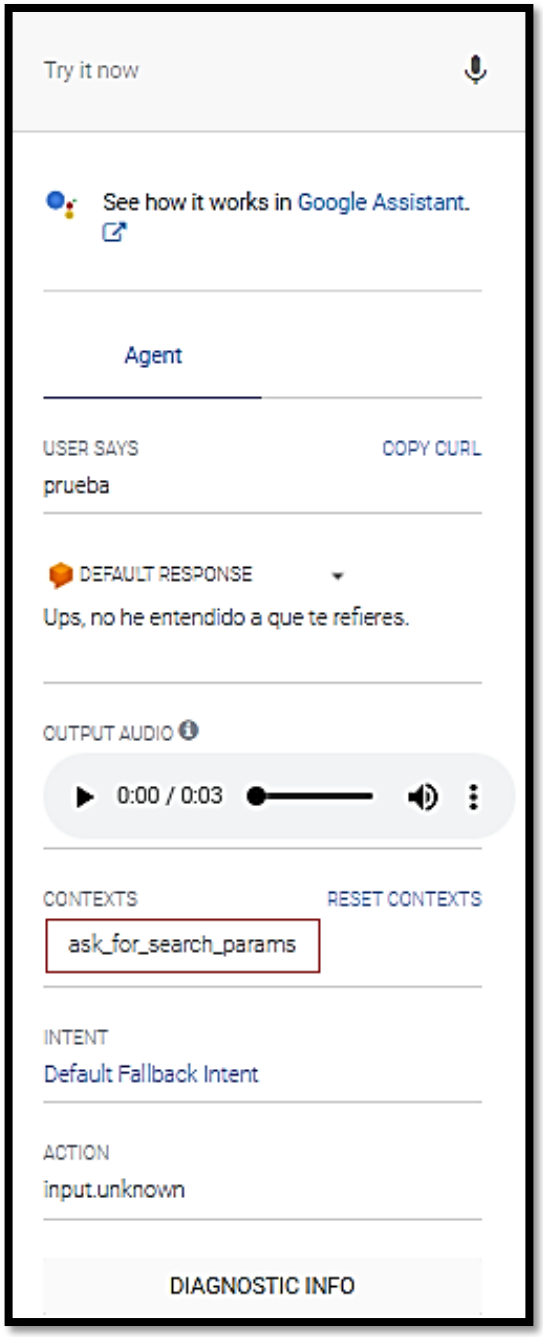


Figura 61. Intención no identificada – Dialogflow.
Fuente: Autor 2018.

Cuando se hizo pruebas usando la aplicación de Facebook Messenger para Android, se evidenció que el idioma por defecto que interpreta Messenger es el que está configurado en el dispositivo móvil. Sin importar de que el texto que se envió haya sido escrito en español.

Dentro de Dialogflow en la sección de Training o entrenamiento, se puede visualizar cuando una intención falla y se puedo corregir y asignar, si así es el caso, a una Intención ya configurada en la plataforma. Esto también es transcendental para las pruebas que se realizaban, (ver Figura 62).

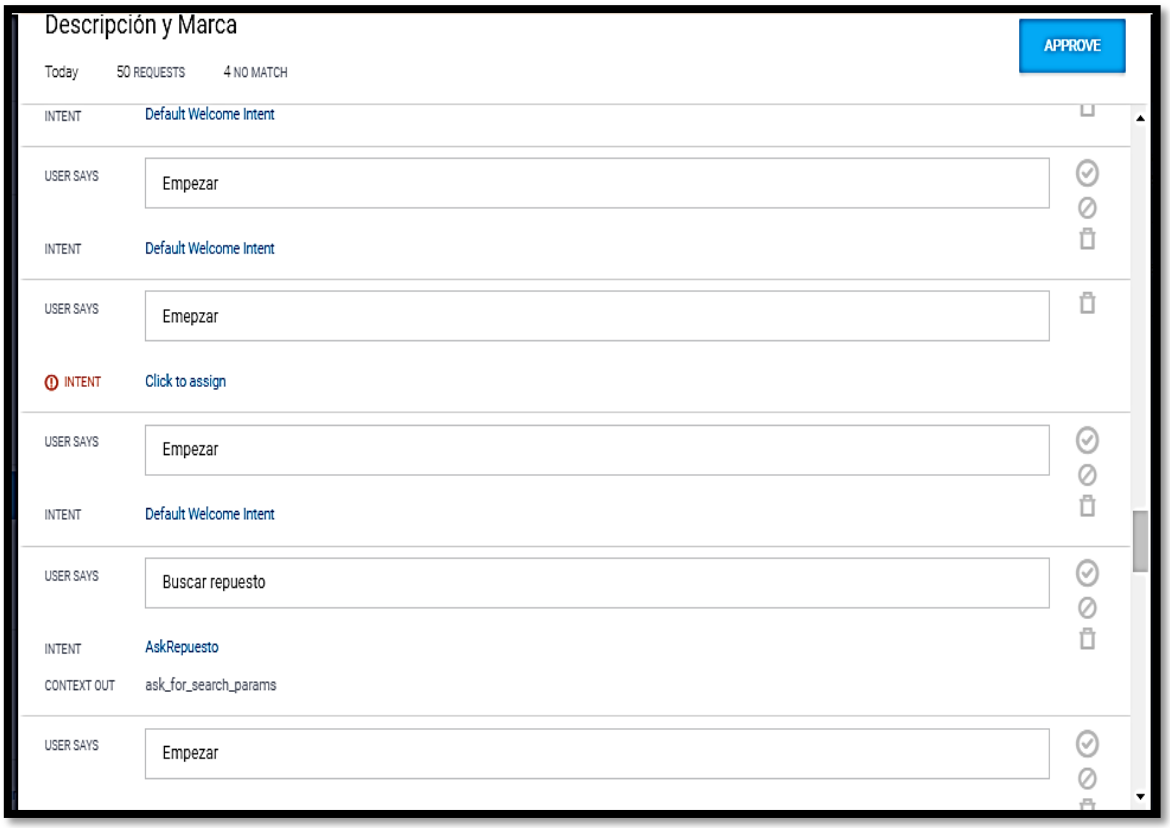


Figura 62. Historial de conversación – Dialogflow
Fuente: Autor 2018

El producto final para esta iteración es siguiente: vía Facebook Web y App de Messenger para Android respectivamente, (ver Figura 63).

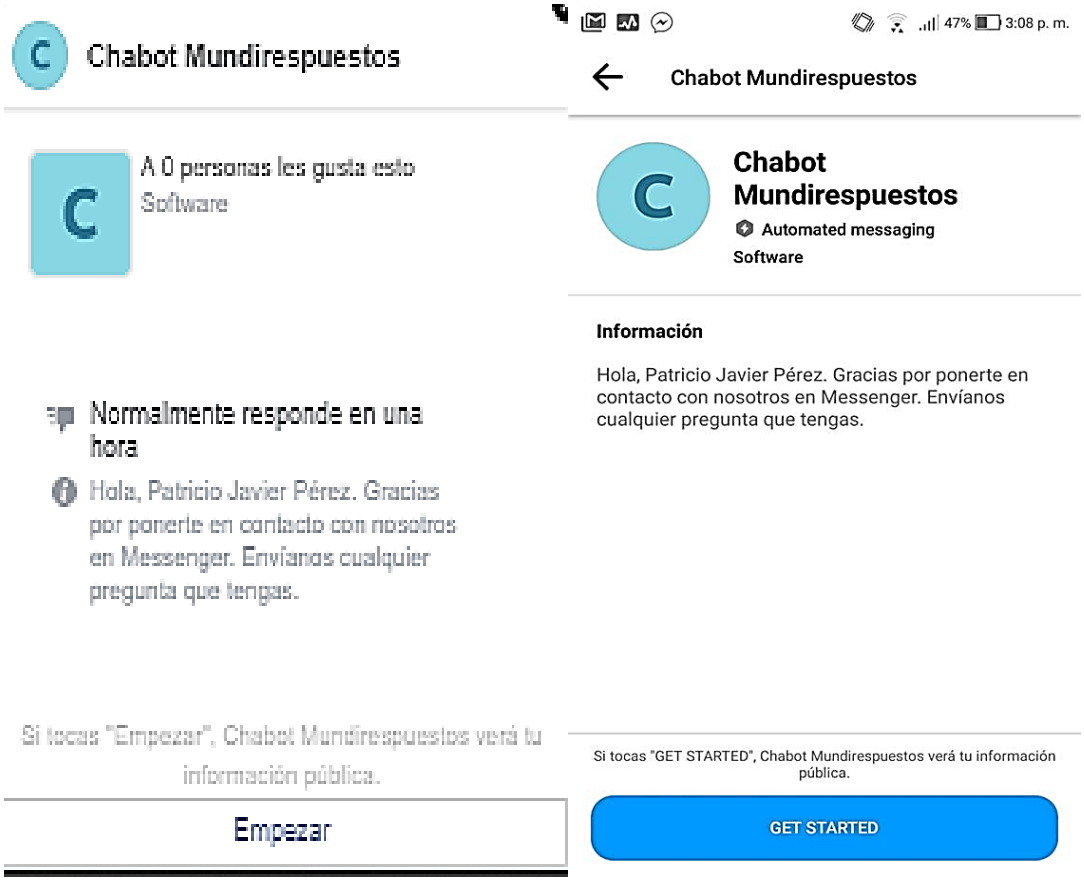


Figura 63. UI de Chatbot.
Fuente: Autor 2018.

En la Figura 64, se muestra las pruebas que se hizo con respecto a la Intención “askFAQ” , “askUbicacion y “askContacto”.

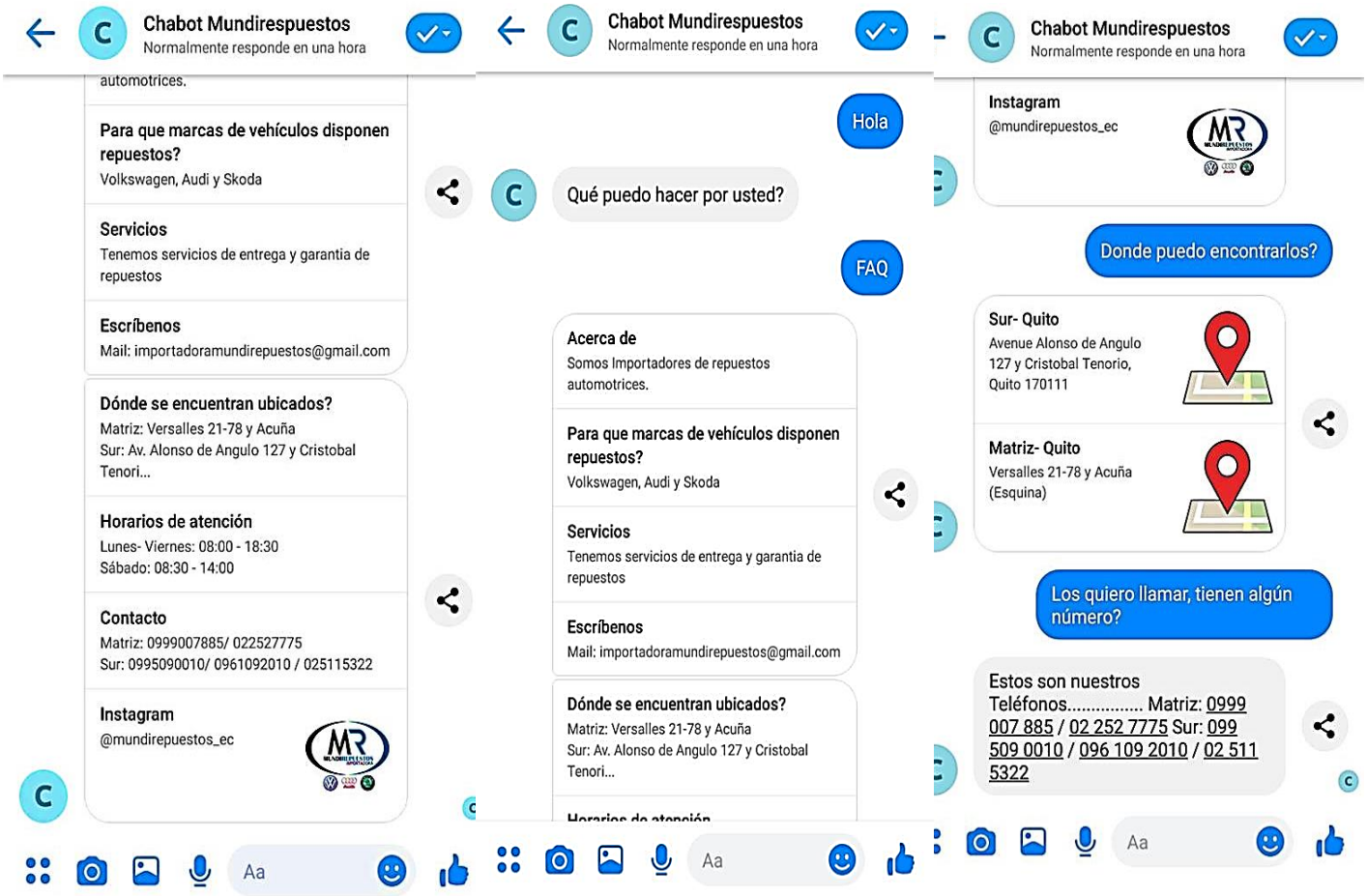


Figura 64. Prueba inteciones 1.
Fuente: Autor 2018.

Cuando Dialogflow no identifico una intención se produce el siguiente resultado, ver Figura 65. Devuelve a Messenger un Default Intent en este caso, el mensaje “Decias?”.



Figura 65. Intención fallida “askMision”.
Fuente: Autor 2018.

3.2.2 Nivel PSP0.1

Como se discutió en la iteración anterior Dialogflow permite ver contextos gracias a su emulador, pero para debuggear y realizar pruebas en esta iteración se utilizó mucho la consola del webhook creado en Express, (ver Figura 66).



Figura 66. Consola Node.js Express.
Fuente: Autor 2018.

También para pruebas se utilizó un proxy “ngrok”, el cual se comporta como un túnel y convertía a el ambiente local en un URL con certificado SSL, la cual permite probar el webhook en Facebook, debido a que este solo permite URL con protocolo HTTPS.

La herramienta POSTMAN para la prueba de llamada a APIS externas permitía ver el estado de la petición, así como tener una mejor visualización de los resultados entregados. En la Figura 67, se hace un llamado hacia la API de Custom Search de Google.

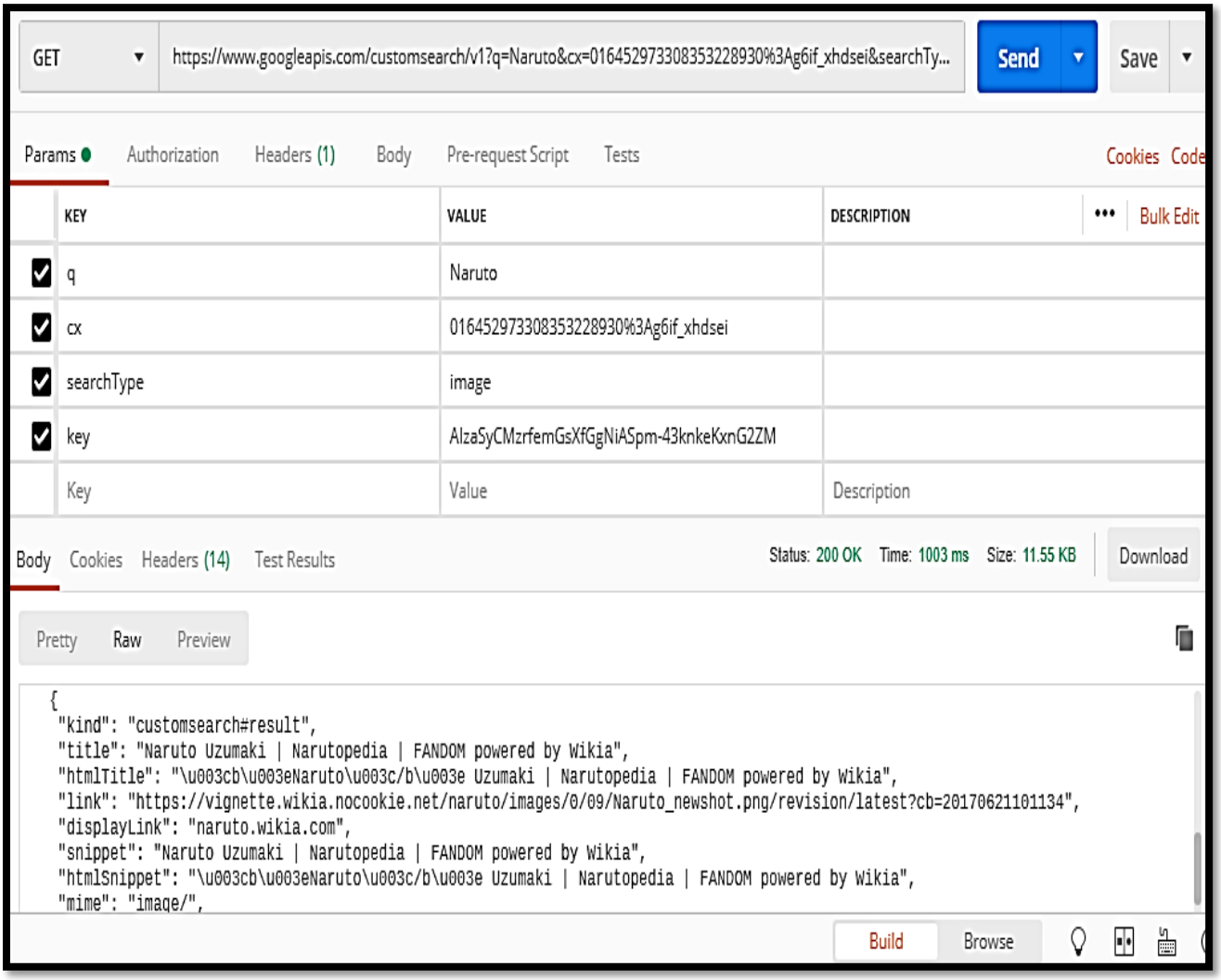


Figura 67. Postman – Google Custom Search.
Fuente: Autor 2018.

Cuando había un error en la petición en las pruebas las respuestas eran las siguientes con estado 400 de Bad Request, (ver Figura 68).

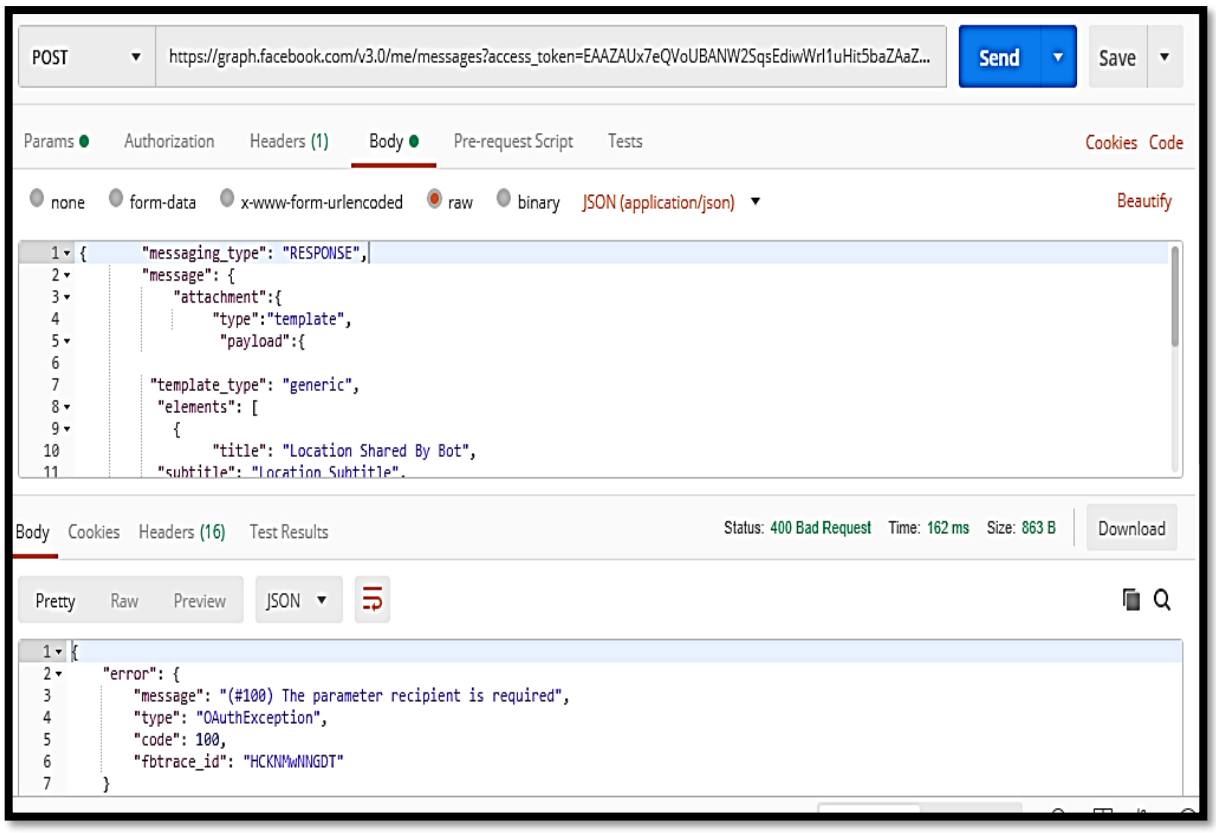


Figura 68. Postman – Bad Request Facebook API.
Fuente: Autor 2018.

Es importante ver el historial de conversaciones para las pruebas para ver si el flujo y las intenciones fueran las correctas; así como que el agente haya determinado las entidades enviadas en cada mensaje. Como en la Figura 69 se visualiza los filtros de parámetros, entidades de marca y repuesto, respectivamente subrayadas.

Así mismo, si no se reconoció una intención o un mensaje no se envió por consecuencia, se puede asignar hacia una intención, correcta. En la Figura 70, se evidencia que la característica de Dialogflow de “follow-up” o seguimiento fallo, así que se le asigna la intención “FindRepuesto - yes”.

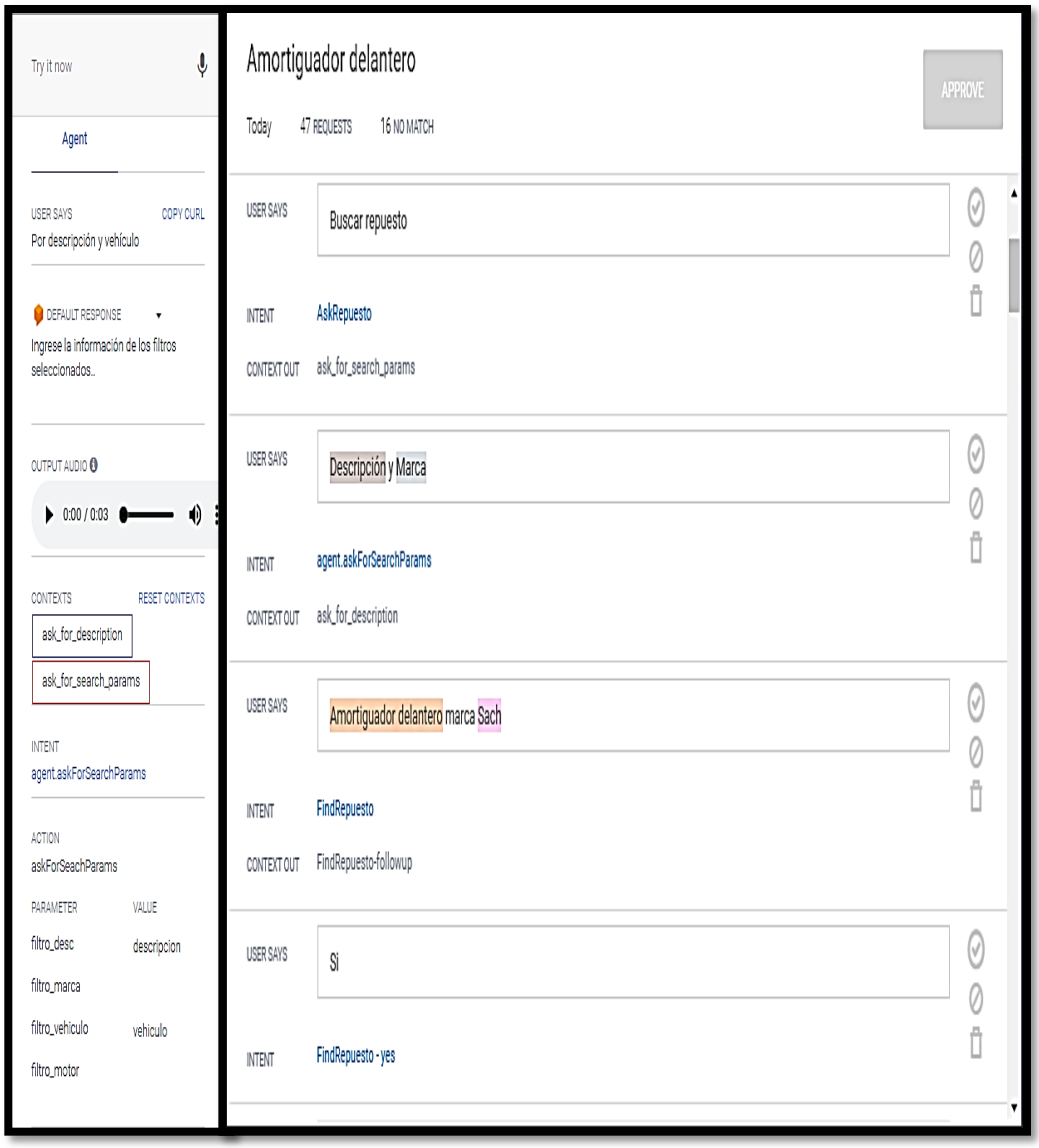


Figura 69. Dialogflow. Parámetros y Entidades dentro del historial.
Fuente: Autor 2018.

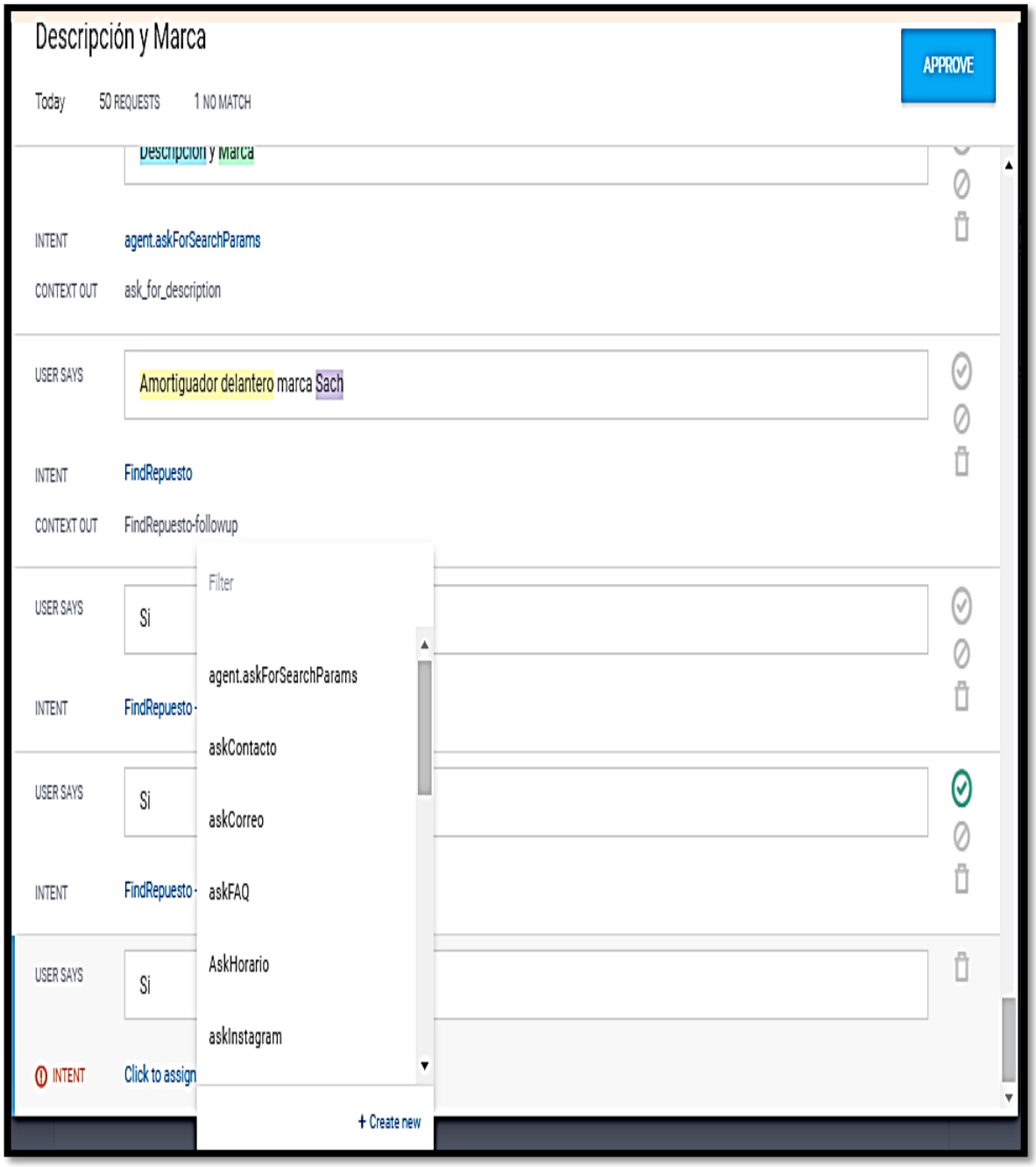


Figura 70. Dialogflow - Asignación correcta de una Intención.
Fuente: Autor 2018.

Esta opción se debió ejecutar previo a la visualización de repuestos ya sea en lista o en carta por el usuario. Preguntándole al usuario. “Es este el repuesto que buscaba?”, (ver Figura 71 y 72).



Figura 71. Búsqueda respuesta flujo parte 1.
Fuente: Autor 2018.



Figura 72. Búsqueda respuesta flujo parte 2.
Fuente: Autor 2018.

3.3 Postmortem

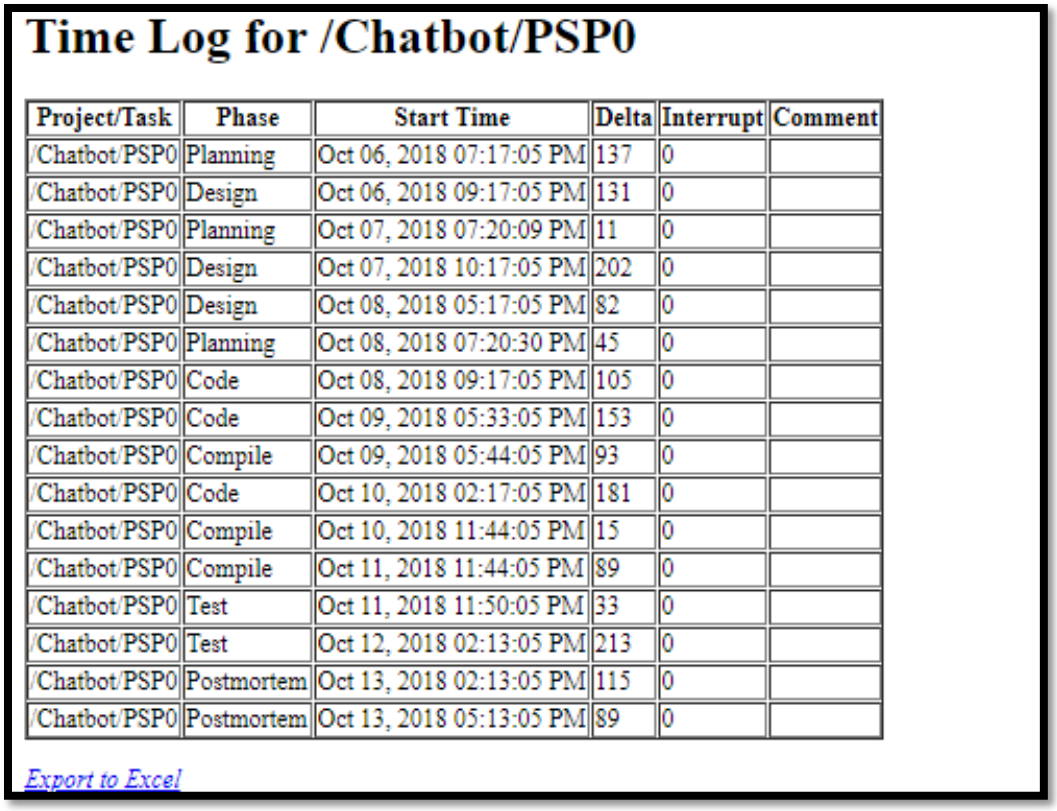
3.3.1 Nivel PSP0

Satisfactoriamente se lograron los objetivos planteados, como configurar la base de datos en Firebase. Pese a ser mi primer acercamiento con bases de datos no relacionales, la configuración de Firebase no fue un impedimento. Dentro de Dialogflow las intenciones de Bienvenida como la de las respectivas preguntas frecuentes fueron configuradas sin complicaciones.

En lo que respecta al entrenamiento de entidades, la visualización de editor en línea sirve para cuando solo se tiene pocas entradas o registros para cada entidad. Es mejor utilizar la funcionalidad RAW para subir archivos CSV.

Dentro de esta iteración se evidenció en gran magnitud la limitación latente del uso del webhook de Facebook con “One click Integration”. Muchos elementos que soporta Facebook Messenger nativamente, no las soporta Dialogflow lo que dificulta la elaboración de un chatbot robusto. Adecuarse al funcionamiento de Dialogflow es relativamente fácil, entender cada componente, sus usos y aplicaciones es fundamental para la construcción de este chatbot. Para la siguiente iteración notando la limitación de este webhook, se implementará un webhook en un webserver propio. Debido a que con este pseudo webhook, no se puede enviar respuestas con variables y consultas hacia APIS de terceros, como Firebase o Google Custom API.

Como se evidencia en la Figura 73, se observa el Time Log del Nivel 0 del marco de trabajo.



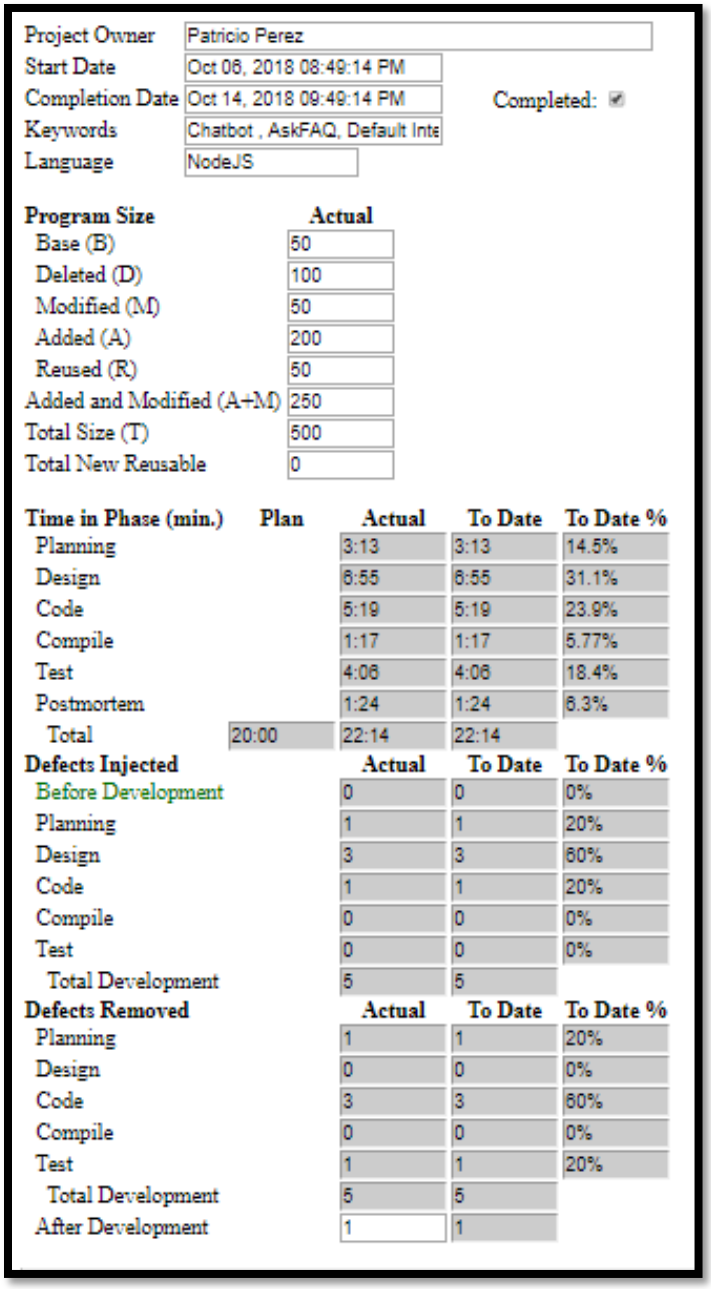
Time Log for /Chatbot/PSP0

Project/Task	Phase	Start Time	Delta	Interrupt	Comment
/Chatbot/PSP0	Planning	Oct 06, 2018 07:17:05 PM	137	0	
/Chatbot/PSP0	Design	Oct 06, 2018 09:17:05 PM	131	0	
/Chatbot/PSP0	Planning	Oct 07, 2018 07:20:09 PM	11	0	
/Chatbot/PSP0	Design	Oct 07, 2018 10:17:05 PM	202	0	
/Chatbot/PSP0	Design	Oct 08, 2018 05:17:05 PM	82	0	
/Chatbot/PSP0	Planning	Oct 08, 2018 07:20:30 PM	45	0	
/Chatbot/PSP0	Code	Oct 08, 2018 09:17:05 PM	105	0	
/Chatbot/PSP0	Code	Oct 09, 2018 05:33:05 PM	153	0	
/Chatbot/PSP0	Compile	Oct 09, 2018 05:44:05 PM	93	0	
/Chatbot/PSP0	Code	Oct 10, 2018 02:17:05 PM	181	0	
/Chatbot/PSP0	Compile	Oct 10, 2018 11:44:05 PM	15	0	
/Chatbot/PSP0	Compile	Oct 11, 2018 11:44:05 PM	89	0	
/Chatbot/PSP0	Test	Oct 11, 2018 11:50:05 PM	33	0	
/Chatbot/PSP0	Test	Oct 12, 2018 02:13:05 PM	213	0	
/Chatbot/PSP0	Postmortem	Oct 13, 2018 02:13:05 PM	115	0	
/Chatbot/PSP0	Postmortem	Oct 13, 2018 05:13:05 PM	89	0	

Export to Excel

Figura 73. PSP – Time Log.
Fuente: Autor 2018.

En la Figura 74, se observa el script de Resumen del plan de proyecto o Project Plan Summary que se ha generado tras haber culminado la fase de Postmortem en PSP.



Project Owner	Patricio Perez
Start Date	Oct 06, 2018 08:49:14 PM
Completion Date	Oct 14, 2018 09:49:14 PM
Completed:	☑
Keywords	Chatbot , AskFAQ, Default Inte
Language	NodeJS

Program Size	Actual
Base (B)	50
Deleted (D)	100
Modified (M)	50
Added (A)	200
Reused (R)	50
Added and Modified (A+M)	250
Total Size (T)	500
Total New Reusable	0

Time in Phase (min.)	Plan	Actual	To Date	To Date %
Planning		3:13	3:13	14.5%
Design		6:55	6:55	31.1%
Code		5:19	5:19	23.9%
Compile		1:17	1:17	5.77%
Test		4:06	4:06	18.4%
Postmortem		1:24	1:24	6.3%
Total	20:00	22:14	22:14	

Defects Injected	Actual	To Date	To Date %
Before Development	0	0	0%
Planning	1	1	20%
Design	3	3	60%
Code	1	1	20%
Compile	0	0	0%
Test	0	0	0%
Total Development	5	5	

Defects Removed	Actual	To Date	To Date %
Planning	1	1	20%
Design	0	0	0%
Code	3	3	60%
Compile	0	0	0%
Test	1	1	20%
Total Development	5	5	
After Development	1	1	

Figura 74. PSP0 – Resumen del plan de proyecto.
Fuente: Autor 2018.

3.3.2 Nivel PSP0.1

Esta iteración, se cumplieron todos los objetivos planteados; como realizar la funcionalidad de búsqueda y desplegar el webhook en la plataforma Heroku. Sin lugar a duda, se requirió de más recurso de tiempo, tanto para la preparación e investigación de usos de APIS de terceros integradas hacia el ambiente a desarrollar con Dialogflow. Un gran impedimento fue el uso de la Google Custom Search API, debido a su límite diario para hacer tan solo 100 peticiones diarias.

Al comienzo de la implementación de esa función se intentó buscar en todo el registro total de repuestos de 3173 repuestos y actualizar la información de imagen, en la base de datos de Firebase, pero las peticiones siempre fallaban, debido a que la versión gratuita de ese API tiene esa limitación.

La herramienta que sirvió enormemente para el debug de la aplicación fue POSTMAN, la cual permitió evaluar las repuestas y visualizar errores con facilidad.

Se trabajó también en una página pública de prueba llamada Chatbot Mundirepuestos, tras la aprobación de la aplicación por parte de la gerencia de la empresa, se migró a la página pública.

Pese a que se quiso hacer uso de las características de texto a voz o viceversa que proporciona Dialogflow, Messenger no las soportaba. Para futuras iteraciones se plantea integrar a asistentes de voz como Google Assistant.

El resumen de plan de proyecto para esta iteración, se presenta en la Figura 75 y 76.

Project Settings

Project Owner	Patricio Perez	
Start Date	Oct 25, 2018 05:07:43 PM	
Completion Date	Dec 12, 2018 05:47:56 PM	Completed: ☑
Keywords	Funcionalidad Buscar repuest	
Language	NodeJS	

Program Size

	Plan	Actual	To Date
Base (B)		155	0
Deleted (D)		20	0
Modified (M)		35	0
Added (A)		542	1127
Reused (R)		23	73
Added and Modified (A+M)	500	577	827
Total Size (T)		700	1200
Total New Reusable		30	30

Time In Phase

	Plan	Actual	To Date	To Date %
Planning	6:33	5:50	9:03	14.5%
Design	11:34	9:04	15:59	25.7%
Code	17:58	19:31	24:50	39.9%
Compile	1:42	1:04	2:21	3.78%
Test	5:50	3:58	8:04	13.0%
Postmortem	1:24	0:32	1:56	3.11%
Total	45:00	39:59	62:13	

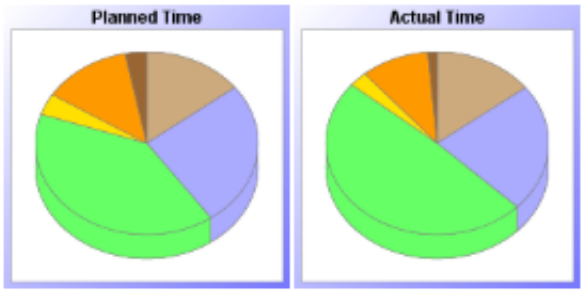


Figura 75. PSP0.1 – Resumen del plan de proyecto parte I.

Fuente: Autor 2018.

Defects Injected

	Actual	To Date	To Date %
Before Development	0	0	0%
Planning	0	1	20%
Design	7	3	60%
Code	6	1	20%
Compile	0	0	0%
Test	0	0	0%
Total	13	5	



Defects Removed

	Actual	To Date	To Date %
Planning	0	1	20%
Design	4	0	0%
Code	4	3	60%
Compile	2	0	0%
Test	3	1	20%
Total	13	5	
After Development	13	1	



Figura 76. PSP0.1 – Resumen del plan de proyecto parte II.

Fuente: Autor 2018.

En la Figura 77, se evidencia el incremento de interacciones por semanas con la página web y el chatbot dentro de Facebook.

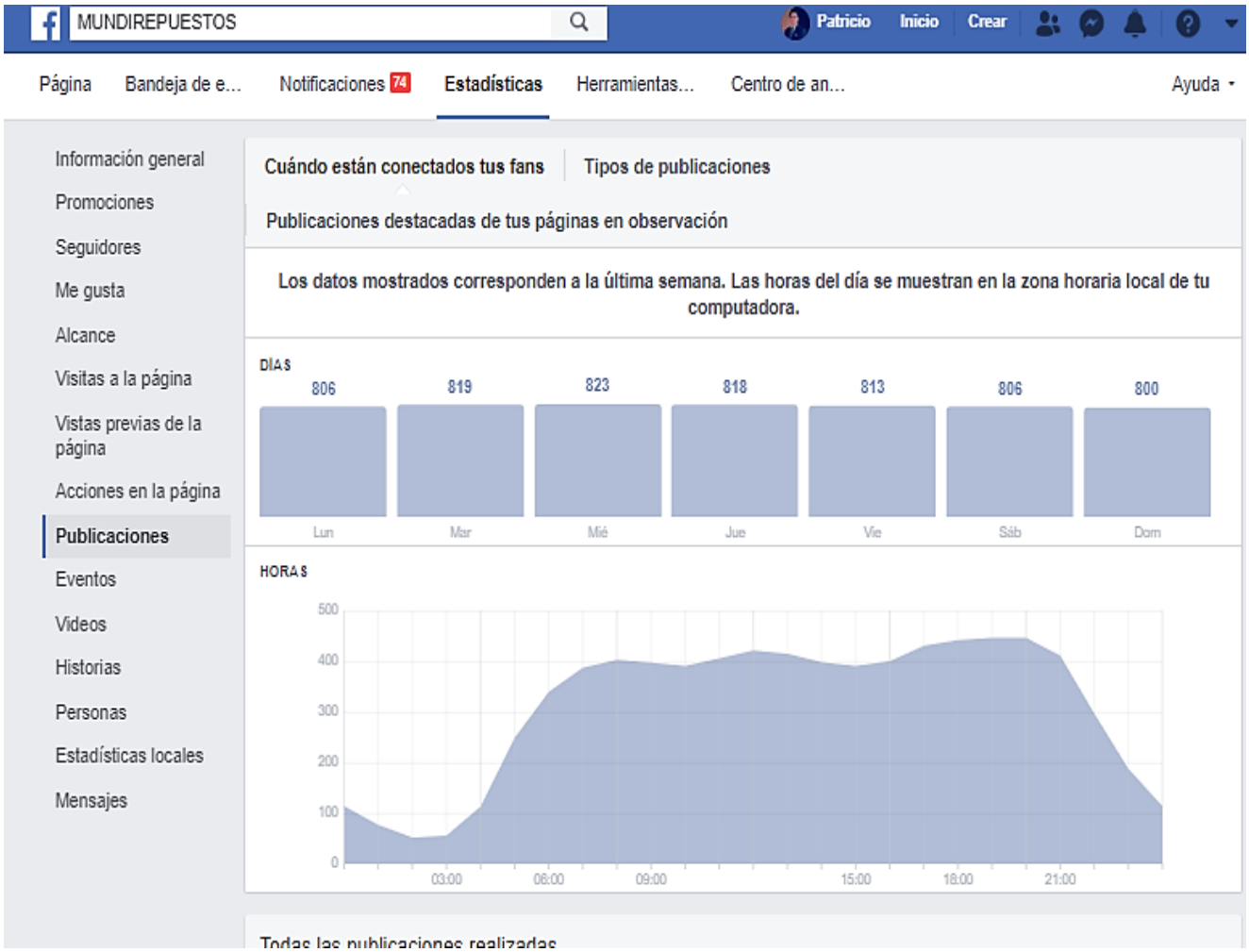


Figura 77. Gráfica de interacciones por semana de FB.

Fuente: Autor 2018.

4 Conclusiones y Recomendaciones

4.1 Conclusiones

- Se determinó que una de los principios usados por los chatbots, es el procesamiento de lenguaje natural (NLP), que permite extraer las conversaciones dentro de un diálogo.
- Existen dos formas para determinar el NLP, las conversaciones basadas en intenciones y las conversaciones basadas en flujo.
- Las conversaciones basadas en flujos usan redes neuronales recurrentes (RNN) o redes con memoria (LTSM), pero estas tienen un mayor costo de implementación, su entrenamiento cuesta más recursos de tiempo y dinero.
- Las conversaciones basadas en intenciones, son las nuevas tendencias en herramientas con respectos a agentes conversacionales, las nuevas plataformas y SDKs, hacen usa de las mismas. Las mismas guardan contextos y hacen uso de aprendizaje de máquina.
- Existen muchas plataformas y librerías que se presentan como Inteligencia Artificial como Servicio (AIaaS), muchas de estas librerías tienen planes gratuitos, como: Dialogflow, LUIS, Lex, Wit.ao, Chatfuel, etc.
- Dialogflow fue la herramienta más idónea para este trabajo, presentaba integraciones de un solo click a la plataforma de mensajería instantánea de Facebook Messenger y varios SDKs para la implementación de un webhook propio

que consuma servicios de APIs de terceros, como fue el caso de la API de Firebase y Google Custom Search API.

- En Heroku, una plataforma como servicio (PaaS), se desplego el chatbot, programado en Node.js Express, con un sistema de integración continúa, vinculado a GitHub.
- En las analíticas y estadísticas, tanto de la aplicación de chatbot como en la página publica de Facebook del negocio, se muestra el incremento de interacciones con los clientes.
- El uso de la practicas PSP, se adecuó perfectamente para este trabajo, debido que la implementación de dicho programa se realizaría tan solo por un programador. El ciclo de vida clásico en cascada utilizado, permite controlar cada fase con la finalidad de tener un proceso de calidad como se puede ver en la poca cantidad de defectos inyectados en cada Fase. Eso indica un buen análisis de los requerimientos y buen diseño de alto y bajo nivel.

4.2 Recomendaciones

- Para futuras versiones de este chatbot, se recomienda integrar hacia más canales de mensajería instantánea o asistentes virtuales, como “Google Assitant”, para hacer uso de funcionalidades dentro de Dialogflow, como son la interpretación de mensajes de texto a voz y viceversa.
- Para almacenar variables y configuraciones que guarden información sensible como lo son llaves secretes para APIS o servicios, se recomienda el uso de variables

de entorno dentro del desarrollo, como son el uso de la librería “dotenv” dentro de NPM.

- Se recomienda visualizar cada semana el historial dentro de la plataforma Dialogflow, para corregir y entrenar las conversaciones que hayan fallado.
- También, se recomienda entrenar a las entidades de repuestos, vehículos y marcas con más catálogos.
- Como PSP no se encuentra atado a una metodología como es la de cascada se puede utilizar en conjunto con una metodología Ágil como SCRUM, pero dado al tamaño de equipo de trabajo, pese a que hay una versión personal adaptada, para proyectos en donde se involucren más personas, se recomienda el uso de las prácticas PSP con esta metodología.

5 Lista de referencias

AWS (2018). Amazon Lex. Recuperado el día 29 de octubre del 2018 de https://aws.amazon.com/es/lex/

Colah, Christopher (2015). Understanding LSTM Networks. Recuperado el día 24 de oct del 2018 de http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Conversable (2018) About Conversable. Recuperado el día 29 de octubre del 2018 de http://conversable.com/about/

Data Monsters. (2017). 25 Chatbot Platforms: A Comparative Table. Recuperado el día 29 de octubre del 2018 de https://chatbotsjournal.com/25-chatbot-platforms-a-comparative-table-aeefc932eaff

DialogFlow. (2018). DialogFlow SDKs. Recuperado el día 29 de octubre de 2018 de https://dialogflow.com/docs/sdks

Fitrianie, S. (2002). My_Eliza A multimodal Communication System, Master of Science thesis, Delft University of Technology. Recuperado el 29 de oct. de 2017 de http://www.kbs.twi.tudelft.nl/Publications/MSc/2002-FitrianiMSc.html

Gartner (2011). CRM Strategies and Technologies to Understand, Grow and Manage Customer Experiences Early. Recuperado el 29 de oct. de 17 de https://www.gartner.com/imagesrv/summits/docs/na/customer-360/C360_2011_brochure_FINAL.pdf

Humphrey, W. S. (2001). Introducción al Proceso Software Personal. Madrid: PEARSON EDUCACION, S.A.

IBM (2018). Build an enhanced IT help desk chatbot on IBM with Watson Assistant. Recuperado el día 29 de octubre del 2018 https://www.ibm.com/developerworks/ibmi/library/i-it-helpdesk

ICapps. (2017). The Chatbot Masquerade: Crafting a personality with NLP and grammar. Recuperado el día 24 de oct. del 2018 de https://www.icapps.com/blog/linguistics-behind-chatbots

Janarthanam Srini, (2017). Hands-On Chatbots and Conversational UI Development.

Joey Whelan. (2018). Google Dialogflow - Input Validation. Recuperado el día 11 de diciembre de 2018 de http://joeywhelan.blogspot.com/2018/04/google-dialogflow-input-validation.html

Mukherjee.Kunal (2017). What is Microsoft's Bot Framework?. Recuperado el día 29 de octubre del 2018 de https://www.quora.com/What-is-Microsofts-Bot-Framework

Rashid Khan & Anik Das. (2018). Build better chatbots. India. Pág. 4.

Shridhar. Kumar (2017) Generative Model Chatbots. Medium. Recuperado el día 24 de octubre de 2018 de https://medium.com/botsupply/generative-model-chatbots-e422ab08461e

Techcrunch (2015). Facebook Acquires Wit.ai to Help Its Developers With Speech Recognition. Recuperado el día 29 de octubre del 2018 de https://techcrunch.com/2015/01/05/facebook-wit-ai/

TechEmergence. (2016). TechEmergence Concensus. AI and Consumer Tecnology. Recuperado el 29 de oct. de 17 de https://www.slideshare.net/TechEmergence/ai-founders-and-executivespredict-5year-trends-on-consumer-tech

Wallace, R. (2003). The Elements of AIML Style. ALICE A.I. Foundation

6 Glosario de términos

Termino	Descripción	Páginas
SDK	Conjunto de herramientas de desarrollo de software que le permite al programador o desarrollador de software crear una aplicación informática para un sistema concreto, por ejemplo ciertos paquetes de software, entornos de trabajo, plataformas de hardware, computadoras, videoconsolas, sistemas operativos, etcétera.	24, 26, 28, 30, 37, 38, 39, 40, 44, 57, 95, 115, 118, 119
API	conjunto de subrutinas, funciones y procedimientos que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.	8, 25, 26, 27, 28, 29, 32, 33, 35, 36, 37, 40, 41, 42, 43, 49, 50, 55, 61, 64, 67, 68, 69, 72, 95, 107, 108, 113, 116, 121
Webhook	Un webhook (también llamado devolución de llamada web o API push HTTP) es una forma en que una aplicación puede proporcionar otras aplicaciones con información en tiempo real. Un webhook entrega datos a otras aplicaciones a medida que ocurren, lo que significa que obtiene datos inmediatamente.	27, 33, 47, 48, 50, 55, 56, 60, 61, 64, 65, 66, 67, 68, 69, 80, 81, 83, 84, 92, 93, 95, 96, 98, 105, 106, 113, 116, 120
Webserver	Un servidor web es un programa que utiliza HTTP (Protocolo de transferencia de hipertexto) para servir los archivos que forman las páginas web a los usuarios, en respuesta a sus solicitudes, que son reenviadas por los clientes HTTP de sus computadoras	50,113
Endpoint	Un endpoint de un canal de comunicación. Para las API, un endpoint puede incluir una URL de un servidor o servicio. El lugar donde las API envían solicitudes y donde vive el recurso, se denomina endpoint.	50, 93, 95
URL	URL significa Uniform Resource Locator, y se usa para especificar direcciones en la World Wide Web. Una URL es la identificación de red fundamental para cualquier recurso conectado a la web (por ejemplo, páginas de hipertexto, imágenes y archivos de sonido).	47, 50, 53, 67, 79, 82, 83, 92, 98, 99 ,106
HTTP		39, 41, 42, 106