UNIVERSITAT DE BARCELONA

Grau de lingüística

Treball de Fi de Grau

Curs 2020-2021

El papel de un lingüista en la mejora de la experiencia del usuario en el proyecto de un chatbot conversacional

Estudiant:
Carmen Pascual Mendizábal

Tutora:
Mariona Taulé I Delor

Barcelona, 15 de Junio de 2021

UNIVERSITAT DE BARCELONA

Coordinació d'Estudis
Facultat de Filologia

Gran Via
de les Corts Catalanes, 585
08007 Barcelona

Tel. +34 934 035 594
fil-coord@ub.edu
www.ub.edu

Amb aquest escrit declaro que sóc l'autor/autora original d'aquest treball i que no he emprat per a la seva elaboració cap altra font, incloses fonts d'Internet i altres mitjans electrònics, a part de les indicades. En el treball he assenyalat com a tals totes les citacions, literals o de contingut, que procedeixen d'altres obres.
Tinc coneixement que d'altra manera, i segons el que s'indica a l'article 18, del capítol 5 de les Normes reguladores de l'avaluació i de la qualificació dels aprenentatges de la UB, l'avaluació comporta la qualificació de "Suspens".

Barcelona, a 15 de Junio de 2021

Signatura: CARMEN PASCUAL MENDIZÁBAL

Membre de:
LERU

Reconeixement internacional de l'excelència
hr
B:KC Barcelona Knowledge Campus
HUBC Health Universitat de Barcelona Campus

Agradecimientos

Me gustaría agradecer su apoyo durante el periodo que ha durado la realización de este trabajo a muchas personas. En primer lugar, a mis compañeros de Quonversa, por la energía y motivación que transmiten, ha sido una ayuda indispensable en mi paso por estas prácticas. A Iñaki, por haberme dado esta oportunidad y confiar en mí. A Ovidio, por ser un líder nato y transmitirnos su energía vital.

Por otro lado, me gustaría agradecerle a Mariona su guía indispensable, ha sido un faro durante este trabajo y me ha aportado la confianza que necesitaba.

A mis padres por apoyarme de forma tan incondicional siempre, y a Marta, por creer tanto en mí y entenderme tan bien.

Resumen

El perfil de un lingüista en el desarrollo de un chatbot es esencial para que la comunicación con el usuario sea eficiente. La creciente aparición de chatbots trae consigo este perfil profesional. A raíz de unas prácticas como lingüista en la startup Quonversa, nace la motivación por documentar una experiencia del trabajo de un lingüista en el proyecto del chatbot Luigi. A lo largo del trabajo, introduzco el concepto de chatbot y cómo se pueden clasificar, qué tareas llevo a cabo y cuáles son las herramientas que utilizo para ello. Me centro también en explicar qué problemas se pueden dar en la comunicación con el usuario y cómo se solucionan.

Palabras clave: chatbot, lingüista, comunicación eficiente, tareas, herramientas.

Resum

El perfil d'un lingüista en el desenvolupament d'un chatbot és essencial per que la comunicació amb l'usuari sigui eficient. La creixent aparició de chatbots dona lloc a aquest perfil professional. Arran d'unes pràctiques com lingüista a la startup Quonversa, neix la motivació per documentar una experiència del treball d'un lingüista dins el projecte del chatbot Luigi. Al llarg del treball, introdueixo el concepte de chatbot i com es pot classificar, quines tasques duc a terme i quines son les eines que utilitzo per això. Finalitzo explicant quins problemes es poden donar en la comunicació amb l'usuari i com es solucionen.

Paraules clau: chatbot, lingüista, comunicació eficient, tasques, eines.

Abstract

In a chatbot's development the linguist is essential for communication with the user to be efficient. The growing emergence of chatbots brings with it this professional profile. This final degree project arises from the motivation to document all the knowledge that I am acquiring as a linguist in practice at the startup Quonversa, in the Luigi chatbot project. Throughout this project I introduce the concept of chatbot and its classification, what tasks I carry out and what tools I use. I also explain what problems can occur in communication with the user and how they can be solved.

Key words: chatbot, linguist, efficient communication, tasks, tools.

ÍNDICE

1. INTRODUCCIÓN

Este trabajo surge de una experiencia propia como lingüista en prácticas en una startup llamada Quonversa[1] en el proyecto de la creación de un chatbot, Luigi, para una agencia de viajes. De esta experiencia me surge la motivación de estructurar todo el conocimiento que estoy adquiriendo, de ampliarlo y enmarcarlo dentro del ámbito de los chatbots, para que pueda servir como una guía, tanto personal como para cualquier persona que pueda estar interesada en conocer una experiencia de un lingüista que trabaja en la creación de un chatbot.

La creciente creación de chatbots trae consigo nuevos perfiles de profesionales. El papel del lingüista en la creación y mejora de un chatbot está dando lugar a algunos de estos perfiles. Cada proyecto tiene unas necesidades concretas, o unas prioridades que decide la empresa. Por ello el papel del lingüista varía en función de cuáles son estas necesidades. Mi papel en el proyecto de creación de Luigi tiene un objetivo claro: dar una buena experiencia de usuario, es decir crear un ambiente que resulte agradable y cómodo para el usuario, y mejorar esa experiencia siempre que sea posible. Nuestro enfoque es el siguiente: el uso del lenguaje moldea experiencias, es esencial la manera en cómo nos comunicamos con el usuario. Para ello, el lingüista dispone de varias herramientas que le permiten mejorar esta comunicación con la aplicación, es decir con Luigi, para que las conversaciones sean lógicas y eficientes. Cada empresa toma sus propias decisiones respecto a qué herramientas usar para ello, en el desarrollo de este trabajo explicaré el uso de DialogFlow[2], una herramienta de Google que utilizamos en Luigi para el Procesamiento del Lenguaje Natural (PLN) y la Comprensión del Lenguaje Natural (CLN).

He estructurado este trabajo de manera que iré acotando el foco de mis explicaciones. En la sección 2, explicaré qué es un chatbot, qué tipos hay según las herramientas de desarrollo de chatbots y cuáles son sus utilidades. A partir de ello, enmarcaré el chatbot Luigi según estas definiciones, describiré los distintos perfiles que trabajan en el proyecto hasta llegar al perfil del lingüista. En la sección 3 explicaré cuál es el papel del lingüista, qué papeles puede desempeñar el lingüista en un proyecto de un chatbot, y llegaré de esta forma a qué papel estoy desempeñando yo concretamente como lingüista en el proyecto de Luigi. En la sección

[1] https://quonversa.com/
[2] https://dialogflow.cloud.google.com/

5, comenzaré a explicar todas las tareas que desempeño, como las llevo a cabo y qué herramientas utilizo para ello.

2. CHATBOTS

En esta sección introduciré el concepto de chatbot, describiré brevemente las modalidades de texto y voz e introduciré los conceptos de Procesamiento del Lenguaje Natural (PLN) y Comprensión del Lenguaje Natural (CNL). A continuación, explicaré en qué ámbitos pueden ser de utilidad los chatbots. Finalmente, termino la sección con una clasificación de los chatbots a partir de dos criterios: si incorporan o no aprendizaje automático, y si la conversación está guiada o no por botones.

2.1. ¿Qué es un chatbot?

Un chatbot o asistente virtual es un programa informático que permite a los seres humanos comunicarse con la tecnología a través del lenguaje natural (Artificial Solutions, 2020). Por ejemplo: en vez de buscar un vuelo en barras de búsqueda, donde rellenas campos como "origen", "destino", "fecha" y "número de pasajeros", lo pides a través de un chatbot como se lo dirías a una persona, es decir formulando una pregunta en lenguaje natural: "Quiero un vuelo de Palma a Barcelona para mañana para 2 personas". Este es un ejemplo, pero no todos los chatbots funcionan de la misma manera (Véase apartado *2.3. Tipos de chatbots)*.

La manera de comunicarse con un chatbot puede ser bàsicamente a través de la voz o de texto. Por lo tanto, una primera distinción entre chatbots sería en función de los métodos o input de entrada, es decir, de la modalidad en que se formula la pregunta (oral o escrita). La diferencia básica es que si se utiliza la voz, lo primero que hace el chatbot es recibir un input de voz y convertir la voz a texto, por ejemplo con un reconocedor de voz, de tal forma que lo que se procesará será también un texto. Tras haberse procesado el texto mediante técnicas de Procesamiento del Lenguaje Natural (PLN) y una vez entendida la intención del usuario mediante técnicas de Comprensión del Lenguaje Natural (CLN), el output o respuesta del chatbot puede ser completar una tarea (por ejemplo, si el usuario pide que el asistente reproduzca una canción), o dar una respuesta concreta. En el caso de chatbots de voz, esta respuesta debe convertirse en voz, es decir, de un texto pasan a voz. Este proceso puede realizarse de manera automática utilizando un sintetizador de voz. Un ejemplo de chatbot que

se basa en este sistema es Alexa[3], un asistente virtual creado por Amazon. En el caso de que el chatbot sea por texto, no hace falta hacer esta conversión de modalidades, la entrada es un texto y la salida un texto también.

Las técnicas, herramientas y recursos de PLN se usan para estandarizar el texto, por ejemplo, convirtiendo todo a minúsculas o corrigiendo errores ortográficos, de forma que tengamos un texto que sea fácilmente procesable para la máquina y se puedan aplicar las técnicas de CLN. La función principal de estas técnicas es entender lo que el usuario ha querido decir, qué intención tiene, cuál es el significado de las palabras que ha utilizado. Un ejemplo puede ser una frase del tipo: "Quiero una entrada de cine para la sesión de esta tarde". Un sistema que incorpora técnicas de CLN debe ser capaz de entender que "esta tarde" contiene el significado de "hoy" y "entre las 15.00 y las 20.00".

Los chatbots se conocen también como bot conversacional de Inteligencia Artificial (IA), asistente de IA, asistente virtual inteligente, agente conversacional, entre otros (Artificial Solutions (2020)). En este trabajo utilizaré el término "chatbot", aunque utilizaré algún otro término de los que he enumerado cuando especifique el tipo de chatbot al que me refiero.

2.2. ¿En qué ámbitos puede ser útil un chatbot?

Los chatbots son muy populares en el entorno de servicio al cliente. Sin embargo, se utilizan también en las empresas para mejorar la experiencia del cliente y la eficiencia. Cada vez más, las empresas están implementando su propio chatbot personalizado en sus páginas web. Las razones pueden ser diversas:

- Resolver dudas del cliente, sustituyendo las llamadas a teleoperadores;

- Preguntar al chatbot información sobre la empresa (contacto, dónde están situados, quién trabaja, etc.), para agilizar el acceso a esta información;

- Hacer pedidos a través del chatbot (compras, reservas, etc.).

Estas son algunas de las utilidades que tiene un chatbot en relación a los clientes, pero

[3] https://developer.amazon.com/en-US/alexa

también puede utilizarse a nivel interno de la empresa o institución:

- Reunir información útil para los empleados, conocimiento nuevo que se adquiera y quiera compartirse, de manera que se pueda acceder en cualquier momento a este de una forma ágil y veloz;

- Como herramienta de organización, ya que recordemos que el chatbot no es solo una interfaz donde se pueda tener una conversación, sino que también puede llevar a cabo acciones tales como reservas y modificaciones.

Es importante remarcar que la intención de esta creciente implementación de los chatbots no es sustituir el trabajo del ser humano, sino complementarlo. En la mayoría de los casos en los que un chatbot toma el control de las acciones, debe estar supervisado actívamente por un humano, del que aprende constantemente. Además, siempre que este no es capaz de resolver algo, se pasa el mando al humano, por lo que podríamos decir que el humano es el que da la solución final.

2.3. Tipos de chatbots

2.3.1. Clasificación según las herramientas de desarrollo

Se puede hacer una clasificación de los chatbots según las herramientas de desarrollo utilizadas, que pueden ser de dos tipos: las que están basadas en modelos lingüísticos (se basan en reglas) o las basadas en modelos de aprendizaje automático (basadas en modelos de Inteligencia Artificial, IA) (Artificial Solutions, 2020).

2.3.1.1. Chatbots basados en modelos lingüísticos

Este tipo de chatbot se caracteriza porque permite determinar previamente cuál será la respuesta a una pregunta. Estos siguen una lógica que suele basarse en reglas del tipo "sí/entonces", y así se crean los flujos de diálogo o de conversación.

Para que sea efectivo, se debe entrenar al chatbot con diversas formas de formular una misma pregunta, teniendo en cuenta aspectos sintácticos y léxicos. Así, se puede asegurar que las

preguntas que tengan una misma intención reciben la misma respuesta. Estos chatbots requieren un mantenimiento constante por parte de un humano, que se encargará de observar los problemas que se dan y así poder afinar el comportamiento del chatbot.

¿Qué ventajas puede tener utilizar las herramientas de desarrollo basadas en modelos lingüísticos? Al poder determinar de antemano las respuestas, y que el chatbot aprende directamente del humano que lo supervisa, ofrecen un alto nivel de control y flexibilidad. Se puede decidir su comportamiento de principio a fin.

¿Cuáles son sus desventajas? Es cierto que las interacciones con estos agentes son muy específicas y muy estructuradas. Tienden a parecerse a las soluciones de FAQs, y sus capacidades son básicas: carecen de inteligencia. Además, requieren de un proceso lento para su desarrollo.

2.3.1.2. Chatbots basados en aprendizaje automático

Los chatbots basados en aprendizaje automático (machine learning) utilizan más datos y son más predictivos. Aprenden de patrones y mejoran con sus experiencias. Son más complejos que los basados en reglas, son capaces de aplicar inteligencia predictiva y tienen más en cuenta el contexto conversacional.

En cuanto a sus ventajas, la conversación puede ser más personalizada y ofrecer una mejor experiencia del usuario. Si hemos dicho que los chatbots basados en reglas pueden ser rígidos y estructurados, estos son más conversacionales y tienen al alcance un número mayor de datos, lo que les permite ser más predictivos. Gracias a las técnicas de inteligencia artificial que utilizan pueden conectarse con recursos externos a través de APIs que permiten que los productos y servicios se comuniquen con otros, sin necesidad de saber cómo están implementados. Por ejemplo, si el chatbot es para hacer pedidos en una pizzería, un recurso muy útil es una API que le permita estar conectado a Google Maps. No tendría sentido crear una aplicación propia para tratar las direcciones si ya hay una aplicación especializada en ello, y con un número mucho mayor de datos. Esto permite que tanto las peticiones como las respuestas puedan ser más complejas.

Sin embargo, el aprendizaje automático tiene como consecuencia que es muy difícil

intervenir, su mejora puede quedar fuera del control del humano. Tay[4], un bot conversacional con inteligencia artificial para la plataforma de Twitter creado por Microsoft en 2016, es un claro ejemplo de esto. Este chatbot pretendía imitar los patrones de lenguaje de una adolescente estadounidense de 19 años. A las 16 horas de su lanzamiento, Microsoft tuvo que suspender la cuenta de Tay por estar publicando tweets ofensivos. Tay es un buen ejemplo para entender que la intervención humana es importante. El humano debe aportar su empatía y ética. La tecnología copia y reproduce patrones, sin cuestionarlo. Nuestra capacidad de cuestionar y reflexionar acerca de cómo pueden afectar ciertos comportamientos a los usuarios es esencial para que el avance tecnológico sea de calidad.

2.3.1.3. Modelo híbrido

Este modelo se beneficia de las herramientas de desarrollo de ambos tipos de chatbots. Gracias al modelo basado en reglas, permite tener bajo control el comportamiento del chatbot, "también ofrece transparencia en el funcionamiento del sistema para que los usuarios corporativos puedan comprender la aplicación y garantiza que se mantenga una personalidad coherente con un comportamiento alineado con las expectativas de la empresa. El enfoque híbrido permite que las integraciones de aprendizaje automático vayan más allá del ámbito de las reglas lingüísticas, para hacer inferencias inteligentes y complejas en áreas donde el uso del enfoque lingüístico es difícil, o incluso imposible (Artificial Solutions, 2020).".

Así pues, con estos modelos se pueden diseñar los flujos de diálogo, son capaces de comprender la intención de un cliente sin importar cómo esté redactada la pregunta y se pueden integrar con sistemas de *back-office* y APIs para que las respuestas sean mucho más personalizadas y complejas.

2.3.2. Clasificación según el uso o no de botones

El uso o no de botones en un chatbot es otra decisión a tomar a la hora de saber qué camino seguir en su creación. La manera de trabajar con cada una de estas opciones es distinta.

[4] https://www.technologyreview.es/s/10110/el-chatbot-racista-de-microsoft-el-mejor-ejemplo-evitar-para-la-ia

2.3.2.1. Chatbots con botones

La mayoría de chatbots que existen ahora mismo usan botones. Este tipo de chatbots no requiere tanto trabajo a nivel de PLN y CLN, y permite un control total de la conversación. Un chatbot puramente con botones guía la conversación a partir de la elección de opciones. No permite la comunicación a partir de la entrada de un texto propio, y si esta opción está integrada, en muchas ocasiones es para registrar ese texto y procesarlo posteriormente por un humano, o guardarlo en una base de datos. Un ejemplo de este tipo de chatbots es Woebot[5], un chatbot fundado por la psicóloga Alison Darcy en 2017. Este chatbot ha sido diseñado para ayudar al usuario en la mejora de su salud mental. Está basado en la psicología cognitiva, cuyas terapias están basadas en el principio básico que el pensamiento moldea o potencia nuestras emociones. Woebot guía al usuario durante toda la conversación a través de botones. Su función es la de facilitar que seas consciente de tus emociones, y de qué pensamientos están provocando o acentuando esa emoción. En algún momento de la conversación pide que el usuario se exprese por escrito, pero no aplica CLN, no comprende el mensaje, simplemente lo guarda, y hace que el usuario reflexione acerca de lo que ha escrito (Véase Figura 1)

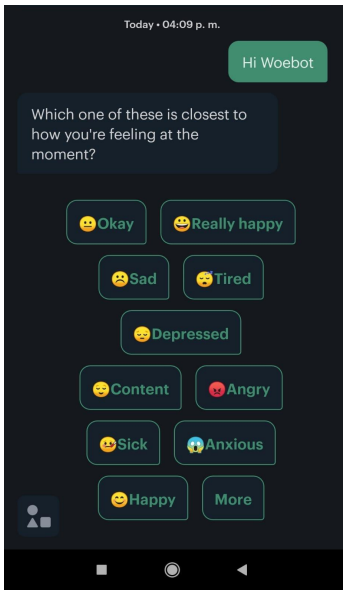

Figura 1. El chatbot Woebot

[5] https://woebothealth.com/

2.3.2.2. Chatbots puramente con Comprensión del Lenguaje Natural.

Este tipo de chatbot permite la comunicación a partir de la entrada de texto. Requieren un gran trabajo a nivel de CLN y PLN. La conversación es totalmente abierta, el usuario puede preguntar lo que quiera. Este tipo de chatbot permite una conversación más natural que los que utilizan solamente botones, y su impresión es más parecida a la de un chat en vivo. Gracias a esta poca rigidez en la conversación, el usuario puede preguntar o pedir que se realice una acción en concreto (por ejemplo: poner música), y eso permite que el equipo que se dedica a hacer evolucionar al chatbot pueda basarse en esta nueva entrada de peticiones para ello. Requieren un trabajo constante por parte del equipo de creación del chatbot, ya que la clave para el buen funcionamiento de este es el entrenamiento diario. Esto es así ya que al ser una conversación abierta, se pierde el control de la conversación en ocasiones, si el usuario pide algo que no está previsto. La conversación puede finalizar por no existir una buena comprensión en esos casos.

Este modelo es el que requiere la intervención más activa del perfil de un lingüista, ya que al ser 100% conversacional se necesita una mayor reflexión acerca de cómo expresar las cosas para que no haya malentendidos y que la conversación sea lo más fluida posible.

2.3.2.3. Modelo mixto (botones y comprensión del lenguaje natural)

El uso de botones y la comprensión del lenguaje natural puede ser también una opción. El usuario puede comunicarse a través de texto o voz, y tener el soporte de los botones en alguna ocasión. Normalmente, el uso de estos botones suele ser a nivel de sugerencias, y se basan en cuáles son las peticiones más frecuentes que se suelen dar. De esta forma, el usuario puede seleccionar una de las opciones de los botones si coincide con la que le interesa.

En estos casos, hay que tener en cuenta que para afinar la comprensión del lenguaje natural es de vital importancia que el chatbot tenga el mayor número posible de información de entrada, ya que ahí está la clave de la mejora del chatbot. Cuánta más entrada de textos haya, más información se puede tener de todas las distintas formas de pedir o preguntar las cosas que quiere el usuario. Trabajar en el entrenamiento es esencial para una mejora en la comprensión del lenguaje natural. En el caso de este tipo de chatbots, el uso de botones puede permitir que no se pierdan algunas conversaciones por falta de comprensión, o reduce la cantidad de veces

que se le hace repetir al usuario lo que ha dicho, pero la entrada de texto será inferior que en un chatbot que funciona puramente con comprensión del lenguaje natural.

2.4. Perfiles que intervienen en su creación

En la creación de un chatbot intervienen varios perfiles profesionales. Los desarrolladores o programadores, los cuales se encargan de convertir la idea y el diseño de chatbot en realidad. Los programadores son los encargados de desarrollar la lógica que hay detrás de todas las posibles conversaciones. También se encargan del mantenimiento de los entornos y de solucionar posibles problemas técnicos que puedan darse que no estén relacionados con el PLN o la CLN, así como las integraciones con otros canales, es decir, hacer posible que podamos comunicarnos con el chatbot a través de distintos canales, como Whatsapp, Telegram o Hangouts.

Otro perfil fundamental es el de los expertos en el negocio. Para que un chatbot proporcione información veraz y soluciones eficientes, es necesario que haya profesionales que se encarguen de definir cuál es la dirección que deben seguir los desarrollos o si la información que proporciona es correcta, está actualizada y sigue la línea de la empresa.

Evidentemente, será necesario el perfil del diseñador gráfico para el diseño de la interfaz del chatbot. Cada empresa toma sus propias decisiones al respecto: el diseñador puede ser una persona del propio proyecto, o alguien externo.

Finalmente, el papel que desarrollo en este trabajo es el de lingüista. El papel que puede tener un lingüista es fundamental en el proyecto de un chatbot, en especial si el chatbot trabaja en su mayoría con Comprensión del Lenguaje Natural. El chatbot está en constante aprendizaje, y la forma como este se comunique dependerá de la capacidad de comunicación de la persona que lo entrena. Un lingüista aporta un amplio conocimiento del lenguaje, herramientas para solucionar problemas que se pueden dar en la comunicación, como puede ser la ambigüedad. Aporta riqueza en el lenguaje, estilo, y la posibilidad de simplificar sin perder información por el camino.

Además, a la hora de diseñar las conversaciones y de resolver problemas, una cualidad añadida al lingüista será la capacidad de abstracción, que le permitirá visualizar el mayor

número de posibilidades que se puedan dar en las conversaciones, o de localizar dónde se está dando el problema para poder darle solución (Véase sección 4.3.3.). Otras cualidades añadidas son la empatía, la comprensión del usuario y las habilidades comunicativas con un equipo técnico. Esto último puede pasar por alto, pero es fundamental que el lingüista sea capaz de ser muy concreto a la hora de explicar su propuesta, como ha diseñado la conversación y como se tendrían que tratar todos los posibles casos que pueden darse en una conversación. Comunicarse con el equipo técnico es fundamental, y a esta conversación deberá acompañarle una buena documentación de la propuesta.

El lingüista puede tener o no conocimientos de programación, esto será decisión de la empresa. Lo que sí que será imprescindible para dar valor al proyecto es que domine la herramienta de gestión de PLN y CLN que se utilice, para poder hacer las modificaciones oportunas en las frases que dirá el chatbot sin depender de un programador para ello, si no su trabajo será muy poco ágil.

Algunos ejemplos de títulos que puede recibir el lingüista dentro del proyecto pueden ser: conversational designer, lingüista computacional, *speech engineer* o *bot master*.

3. LUIGI

En esta sección explico cuál es la función del chatbot Luigi, qué tipo de chatbot es según la clasificación que he explicado en la sección *2.3.* y qué perfiles profesionales intervienen en su desarrollo.

3.1. ¿Qué es Luigi?

Luigi es un chatbot diseñado por la startup Quonversa (Véase figura 2). Es un asistente virtual enfocado a una agencia de viajes. La labor de Luigi es la de asistir a usuarios en la reserva de vuelos, resolver todas las posibles dudas que puedan surgir sobre aerolíneas, políticas de equipaje, normativas de los países, etcétera. Luigi es un producto que ofrece soluciones bot para una agencia de viajes. Una solución bot no es ni más ni menos que la aplicación de automatismos a cuestiones que resuelven los humanos: venta telefónica, *call centers* o post venta. Las soluciones bot son evolutivas, deben verse como el aprendizaje que realiza un humano para desarrollar tareas.

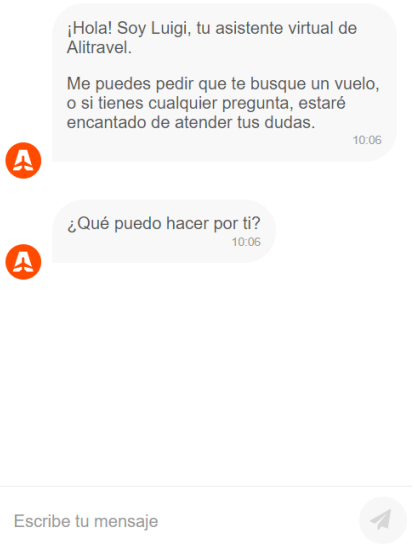

Figura 2. El chatbot Luigi dando la bienvenida a un usuario nuevo.

Según las herramientas de desarrollo de chatbots que he definido en el apartado *2.3.*, podemos clasificar a Luigi como un modelo híbrido. Las conversaciones de Luigi han sido diseñadas a partir de modelos lingüísticos, basados en reglas. Pero además, utiliza Inteligencia Artificial. Está conectado a sistemas de *backend*, por lo que tiene acceso a información que da unas respuestas personalizadas y concretas. Un sistema de *backend* recoge información de los usuarios u otros sistemas de tratamiento de datos en la compañía. Es el encargado de gestionar la información que proporciona el usuario. Trabaja con aprendizaje automático, que está integrado en la herramienta de PLN y CLN que utiliza. En concreto, la herramienta que utilizamos es Dialog Flow, creada por Google. Luigi es capaz de llevar a cabo acciones, por lo que puedes reservar un vuelo a través del chat sin necesidad de salir de él en ningún momento.

Luigi trabaja exclusivamente con Comprensión del Lenguaje Natural, y su input es texto. La idea es que puedas hablar a través del chat de la misma forma como lo harías con un humano en un chat en vivo, siempre teniendo en cuenta que estás hablando con un chatbot que está aprendiendo. Si en algún momento el chatbot no es capaz de resolver una duda, o de finalizar una acción, se pasará el mando a un agente humano. Este agente recibe toda la información que el usuario ya ha proporcionado al chatbot, de forma que no se empiece la conversación de cero (Véase figura 3).

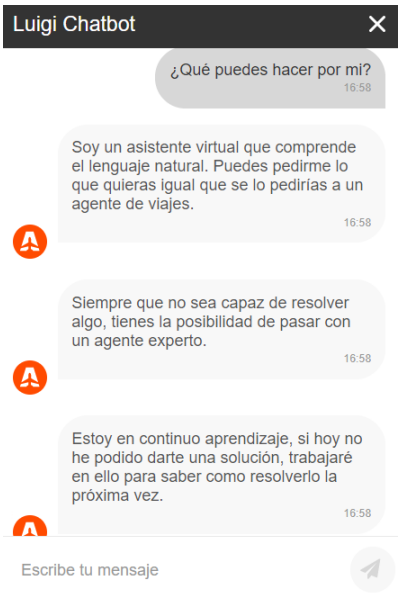

Figura 3. Luigi explica al usuario cómo trabaja

3.2. Perfiles que intervienen en el desarrollo de una solución bot en Luigi

El equipo de Luigi es multidisciplinar (Véase figura 4). Se compone de un equipo técnico (Core), encargado de gestionar la arquitectura de los entornos, llevar el mantenimiento de la plataforma, llevar a cabo la automatización de procesos y crear herramientas de monitorización tanto para el equipo de desarrollo (detectar errores, controlar los tiempos de respuesta, etc.) como para el bot master (visualización de conversaciones y detección de las rupturas en el flujo conversacional).

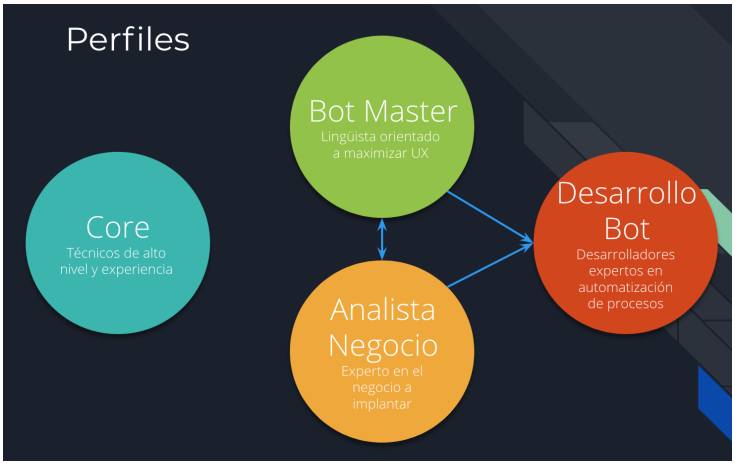

Fig.4. Esquema de los perfiles que trabajan en el proyecto Luigi.

Por otro lado, los analistas de negocio, los cuáles son expertos en el terreno de negocio en el que actúa el chatbot. Ante una pregunta de un cliente, debe saber qué debe contestar un humano, no importa el estilo, pero si el contenido. Debe conocer las fuentes de datos que se utilizan para dar una respuesta eficiente al cliente.

Finalmente, lo que nos concierne en este trabajo: el lingüista. Mi objetivo principal como lingüista en este proyecto consiste en maximizar la experiencia de usuario (UX). Esta labor puede suponer tanto mejoras (nuevas implementaciones y nuevos desarrollos), como el buen mantenimiento del contenido. El título que recibo como lingüista en este proyecto es el de bot master. El bot master es el "profesor" del chatbot. Es quién le enseña a comunicarse. Por eso es interesante que un perfil lingüista sea el bot master. En definitiva, la personalidad que adopte el chatbot dependerá en gran medida de la propia personalidad del bot master.

Como se observa en la Figura 4, los desarrolladores, analistas de negocio y el bot master están siempre conectados. El bot master está en constante comunicación con el analista de negocio, y ambos se comunican con los desarrolladores, de forma que el proyecto avance siempre con la colaboración de estos tres perfiles.

A continuación, explicaré todas las tareas que realizo en el proyecto, como las realizo, qué herramientas utilizo para ello y de qué manera mi formación como lingüista aporta valor en la realización de tareas.

4. EL PAPEL DEL LINGÜISTA

Como he explicado anteriormente, mi papel como lingüista en el proyecto del chatbot Luigi tiene un objetivo claro: dar una buena experiencia de usuario y mejorarla diariamente. El título que se me ha dado como lingüista es el de bot master. ¿Por qué elegir el perfil de un lingüista para este trabajo? Luigi trabaja puramente con comprensión del lenguaje natural, y el lingüista puede ayudar y aportar ese conocimiento sobre el funcionamiento del lenguaje natural y de las lenguas en particular. El botmaster es el profesor del chatbot. Es la persona que diariamente debe analizar los problemas que se hayan dado en las conversaciones, detectar sus causas y dar soluciones. Para ello, tiene al alcance varias herramientas, con las

que detectar todos los posibles problemas que se puedan estar dando, y las soluciones para estos pueden ser muy variadas. En este apartado explicaré las herramientas que utilizo, qué problemas se pueden detectar y cómo les doy solución.

Dado que mi llegada al proyecto de Luigi viene tras un año del lanzamiento de este proyecto, mi trabajo no parte de cero. Lo primero que he tenido que hacer es editar textos. Mi primera función fue la de edición de textos en las preguntas frecuentes (FAQs). Este será un primer apartado en la explicación de mis tareas. Mientras explique mis tareas, introduciré todos los conceptos esenciales para entender en qué consiste mi trabajo y todo lo que he aprendido acerca de la herramienta Dialogflow. Dialogflow, como ya he dicho anteriormente, es la herramienta de CLN que utilizamos en Luigi. Por ello es especialmente interesante a la hora de entender el trabajo que puede realizar un lingüista en el proyecto de un chatbot.

4.1. Edición de Preguntas frecuentes (FAQs)

La primera tarea que llevé a cabo al llegar a Quonversa fue la edición de las preguntas frecuentes que tenían integradas en el chatbot Luigi (Véase figura 5). El objetivo de esta tarea es conseguir claridad en las respuestas, evitar ambigüedades y simplificar. El contenido de estas respuestas las aporta un experto en negocio, en este caso en vuelos. ¿De qué manera se lleva a cabo esta tarea? Hay que fijarse en varios aspectos: el léxico que se utiliza, la sintaxis o estructura de las frases, la semántica y la ortografía.

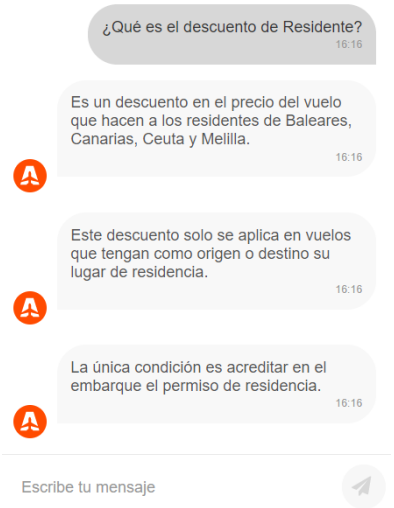

Figura 5. Ejemplo de pregunta frecuente en el chatbot Luigi

En primer lugar, en cuanto al léxico, se debe evitar los tecnicismos. Por ejemplo, términos como “cia”, que quiere decir “compañía aérea” o “aerolínea”. Otro ejemplo es “100% de gastos”, que quiere decir “importe total”. El usuario debe entender lo que se le está diciendo, y este tipo de tecnicismos puede entorpecer mucho la comunicación.

En segundo lugar, en cuanto a los aspectos sintácticos, es importante prestar atención a la estructura de las frases que se le muestra al usuario. Mi tarea principal en este sentido es la de simplificar el texto para que se pueda entender mejor. Por ejemplo, se debe evitar frases muy largas, el uso de muchas coordinadas o subordinadas. También es importante suprimir información redundante.

En tercer lugar, en cuanto a aspectos semánticos, es importante prestar atención al significado de lo que se está diciendo, hay que ser muy cuidadoso en este aspecto. Un ejemplo de esto es la diferencia entre estas dos frases: “¿En qué municipio resides?” y “¿En qué municipio estás empadronado?”. El significado de estas dos preguntas no es el mismo. Hay que ser muy concreto en la manera de plantear las preguntas.

Finalmente, la ortografía. Es importantísimo la corrección de las faltas ortográficas. No se puede permitir que el chatbot cometa faltas de ortografía. Las faltas de ortografía que cometa el usuario las procesamos (PLN) y estandarizamos el texto para poder comprenderlo (CLN) después.

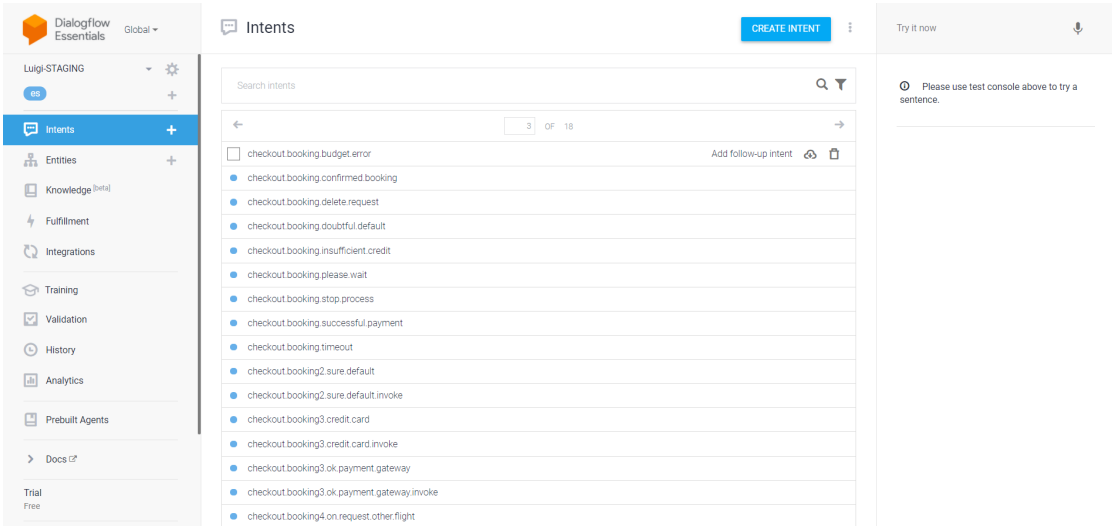

Figura 6. Interfaz de Dialogflow

La mayoría de las respuestas a estas preguntas frecuentes se encuentran en el propio agente de Dialogflow (Véase figura 6).

En este apartado explicaré el concepto de *intent* que aparece en Dialogflow. Como se observa en la figura 6, lo que está marcado en azul son los *intents*. Un *intent* no es ni más ni menos que una intención del usuario. Cada intención del usuario requiere la creación de un nuevo *intent* dentro del agente de Dialogflow. Una intención, básicamente, es lo que quiere el usuario, su petición, qué necesita concretamente. Las preguntas frecuentes se encuentran cada una recogida en *intents* específicos para cada pregunta frecuente. En Luigi, estos *intents* se nombran con el prefijo "faq." (Véase figura 7).

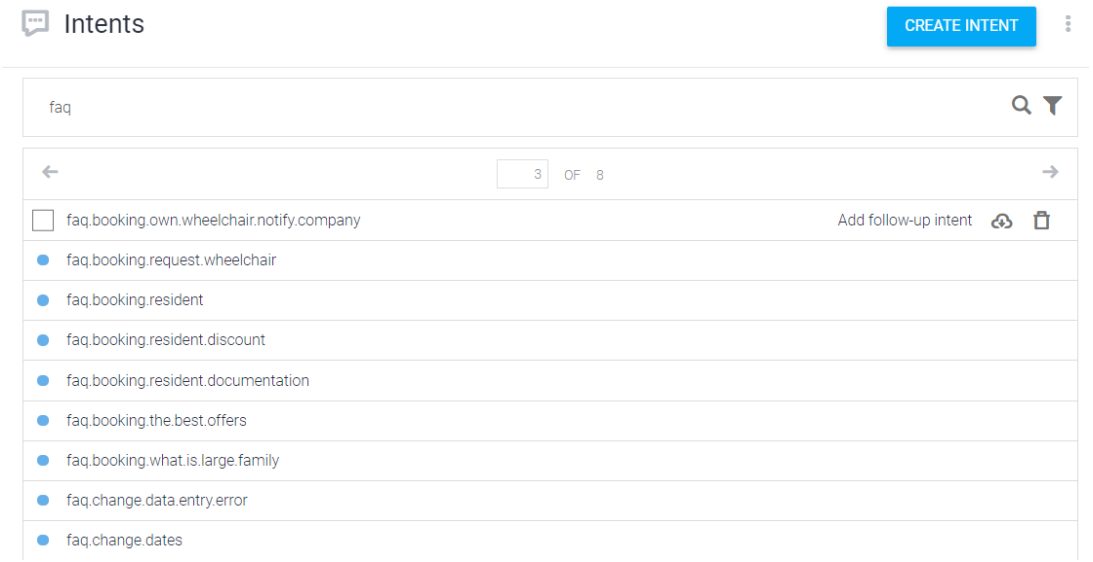

Figura 7. Búsqueda de *intents* con el prefijo "faq."

En la figura 5 se ve un ejemplo de pregunta frecuente en el chatbot Luigi. La respuesta a la pregunta frecuente (¿Qué es el descuento de residente?) ha sido editada en el agente de Dialogflow. En la figura 8 podemos observar dónde están guardadas estas respuestas.

Para su edición, se hacen los cambios oportunos en los recuadros correspondientes, y se le da al botón de "SAVE" que se ve arriba a la derecha. Estos cambios se ven reflejados al instante en el chatbot. Una vez se realiza un cambio de este tipo, se puede probar de inmediato y comprobar si está todo correcto.

Figura 8. Una muestra de las respuestas del *intent* "faq.booking.resident".

4.2. *Automatización de procesos manuales*

Otra de las tareas que tiene el botmaster es la de detectar todos aquellos procesos que hay que acabar derivando a un agente humano por no ser capaces de resolverlo de forma automatizada. Para entender esto, es importante tener presente el siguiente esquema (Véase figura 9).

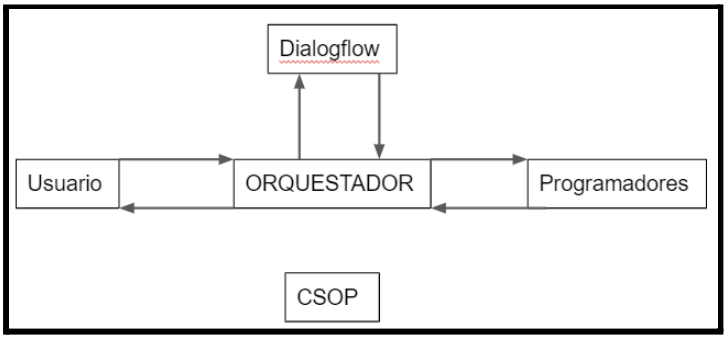

Figura 9. Esquema de los movimientos internos de las peticiones de los usuarios

Cada uno de estos componentes tiene su función en el procesamiento, comprensión y respuesta que se le dará al usuario. En primer lugar, el orquestador, el programa central que gestiona la relación entre los entornos, de cuyo mantenimiento se encarga uno de los técnicos del equipo. Este es el que decide de forma automática a dónde se deben enviar cada una de las peticiones, y el que mantiene el flujo de la conversación para no perder el hilo. Cualquier

mensaje del usuario que entre, deberá pasar siempre por el Dialogflow, que será el encargado de procesar y comprender el mensaje, con tal de detectar a qué *intent* corresponde la petición del usuario. Si la respuesta se puede dar directamente desde el Dialogflow, no tenemos por qué pasar por una lógica de negocio (programadores), se le devuelve un mensaje al usuario. En caso de que la respuestas requiera de más datos y no sea una respuesta genérica, habrá que pasar por lógica de negocio.

Todos los circuitos que se ven en el esquema (Véase figura 9) están automatizados, todo está previamente diseñado. En el momento que algo no está automatizado, no se podrá resolver de forma automática y por tanto se le derivará al usuario a un agente humano experto. En el esquema de la figura 9, este agente se encuentra en CSOP. Siempre que alguna petición acabe en CSOP, el botmaster debe detectarlo y ver cómo esto puede ser automatizado.

4.2.1. Flujo conversacional

En el momento en que se detecta una petición del usuario que no está automatizada, lo primero que se debe hacer es hablar con el experto de negocio. Es importante resolver: qué le diría al usuario en este caso, cuál es el proceso a seguir y qué casos pueden darse. Hay que contemplar la mayor cantidad de casuísticas posibles.

A raíz de esta conversación, se debe empezar a diseñar el flujo de diálogo. A continuación, muestro un ejemplo. Un usuario pide que quiere hacer una consulta sobre su reserva. Este proceso no está automatizado, por lo que se le pasa a un agente para que pueda hacer la consulta. Paralelamente, se inicia el proceso para llegar a automatizar esta petición. Para diseñar el flujo de diálogo, la herramienta que utilizamos en Luigi es “Google Drawings”, una herramienta de Drive[6].

A estos esquemas se les llama diseños funcionales (Véase figura 10). Se trata de un esquema intuitivo, en el que la conversación avanza según el movimiento vertical hacia abajo de los recuadros azules de la izquierda. El recuadro en color rojo muestra un posible mensaje del usuario, en este caso “Quiero consultar mi reserva”. A partir de ahí, Dialogflow detecta la intención del usuario, que es la de consultar una reserva. Entonces, comienza el proceso.

[6] https://www.google.com/intl/es_es/drive/

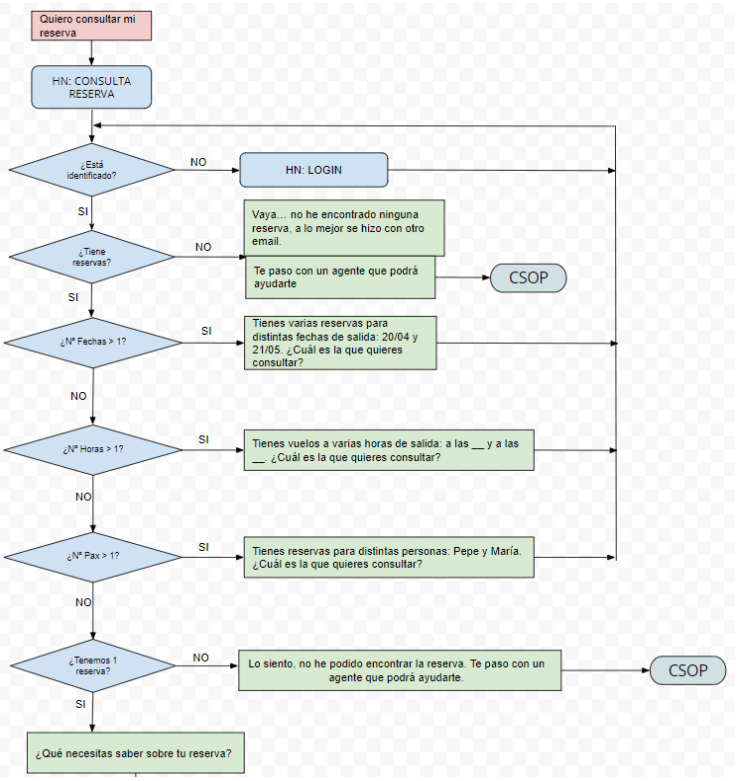

Figura 10. Muestra de un diseño de flujo conversacional hecho con Google Drawings

Según este diseño funcional, lo primero que se debe hacer es filtrar sus reservas. Este paso no es decisión mía, es una decisión por parte de negocio, se quiere que el filtrado sea así. Esto me lo hacen saber en la primera conversación que tengo con el experto de negocio. Así pues, se hace una comprobación de las reservas del usuario, y si no hay únicamente 1 reserva, se debe hacer un filtrado hasta conseguir 1. Los recuadros de color verde muestran los mensajes que le dice Luigi al usuario. Finalmente, cuando el usuario especifica a cuál de sus reservas se refiere, Luigi le pregunta qué es lo que necesita saber. A partir de ahí, cada una de esas consultas será una nueva intención: puede querer saber de qué terminal sale su vuelo, o si tiene algún tipo de retraso en el horario. Eso será un nuevo diseño funcional para cada una de esas intenciones.

4.2.2. Creación de *intents*

Una vez hecho el diseño funcional hay que crear el o los *intents* que intervendrán. Como he explicado en el apartado *4.1.,* un *intent* es la intención del usuario.

Los *intents* los creamos en la consola de Dialogflow. Para crear un *intent* se debe tener en

cuenta: los parámetros y las entidades con las que se detectarán estos parámetros, las frases de entrenamiento y los contextos. A continuación explicaré cada uno de estos componentes que conforman un *intent*.

4.2.2.1. Parámetros y entidades (*entities*)

En este apartado se describen varios conceptos: parámetro, entidad, valor del parámetro, valor de la entidad y sinónimos.

Se puede definir un parámetro como aquello que puede variar dentro de una misma frase y que da información relevante que necesitamos recoger. Por ejemplo, un parámetro puede ser una fecha. Dentro de una frase del tipo "¿Puedes buscarme un vuelo para el día 10 de febrero?", "10 de febrero" es un parámetro que puede variar.

Por ello, cuando se crea un *intent* hay que pensar cuáles van a ser los parámetros que habrá que recoger. Los parámetros se recogen con entidades. Una entidad permite recoger distintos parámetros dentro de una misma frase y poder clasificarlos para que le llegue al desarrollador la información que necesita. Gracias a las entidades, no hay que crear una frase de entrenamiento (Véase apartado 4.2.2.2) por cada caso concreto. En la figura 11 se muestra dónde se encuentran las entidades en Dialogflow.

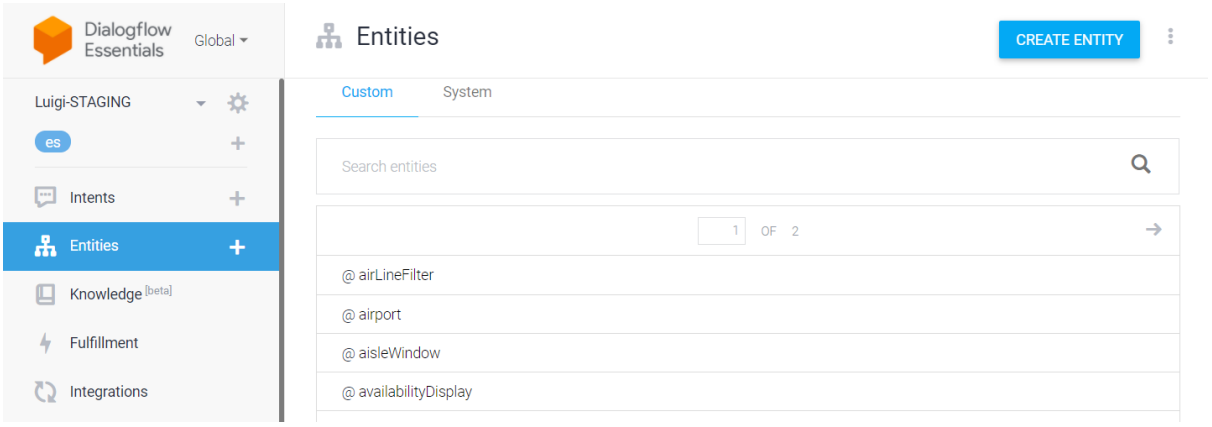

Figura 11. Apartado *Entities* en la consola de Dialogflow

En DialogFlow hay entidades ya definidas por el sistema. Estas entidades se reconocen fácilmente porque siempre van precedidas del prefijo "@sys."

Los usuarios de Dialogflow tienen la opción de crear, además, sus propias entidades en el apartado "Custom" que se puede ver en la figura 11.

En una frase del tipo "quiero un vuelo a Madrid", "Madrid" se recoge gracias a una entidad que engloba lugares del mundo (@sys.location) y se cataloga de esta forma como lugar. Si la frase es "quiero un vuelo a Madrid mañana", entonces hay dos parámetros a tener en cuenta, que se clasifican con las entidades: "Madrid" es un lugar (@sys.location) y "mañana" es una fecha (@sys.date) (Véase figura 12).

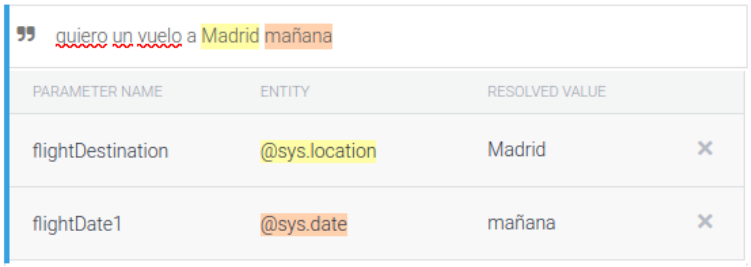

Figura 12. Muestra de una frase con parámetros recogidos con distintas entidades

Las entidades se definen dentro del propio *intent.* Se decide qué entidades van a recoger los parámetros de las frases del *intent*, y hay que decidir también el valor de los parámetros. Si se recoge con la entidad @sys.date "mañana", se está decidiendo que se trata de una fecha. Ahora bien, ¿es la fecha de ida o vuelta? Para concretar, se utilizan distintos valores de parámetros. En la figura 13, a la izquierda del nombre de la entidad se puede observar el nombre del valor del parámetro, flightDate1, que hace referencia a la fecha de ida y flightDate2 a la de vuelta (Véase figura 13).

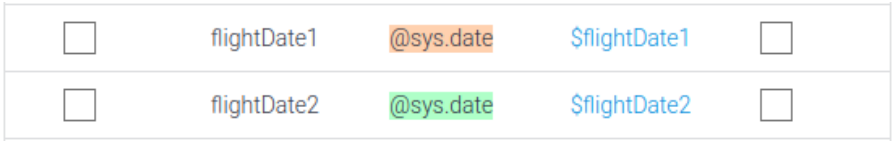

Figura 13. Entidades con sus respectivos valores de parámetro

Otra cosa a tener en cuenta a la hora de crear una entidad es los valores de esa entidad. Una entidad puede tener varios valores. Por ejemplo, la entidad @flightOW recoge si el vuelo es solo de ida o si es ida y vuelta. Los valores de la entidad en este caso es *True* o *False*. Es *True*, si efectivamente, tal y como indica el nombre de la entidad, el vuelo es "sólo ida

(OneWay)". Será *False*, si es un vuelo de "ida y vuelta" (Véase figura 14).

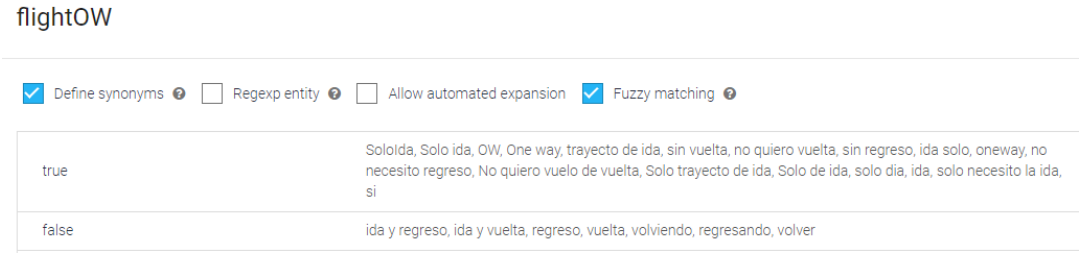

Figura 14. Valores de la entidad "@flightOW"

En la figura 14, lo que se ve a la derecha son los sinónimos. Los sinónimos indican todas las formas posibles que tendrá el usuario de decir lo que se traducirá a un valor en concreto, que será lo que le llegará al desarrollador. Es decir, hay muchas maneras de decir que "quiero un vuelo solo ida". El usuario puede decir, por ejemplo: "no quiero vuelta", "sin vuelta"o "solo necesito la ida". Hay que ser capaz de interpretarlo. De esta forma el desarrollador no tiene que interpretarlo y le llega un único valor.

4.2.2.2. Frases de entrenamiento

Una vez se ha definido qué entidades van a recoger los parámetros en el *intent,* llega el momento de entrenarlo con frases de entrenamiento. Se pueden definir las frases de entrenamiento como las distintas formas que tiene el usuario de hacer una misma petición o pregunta. Son clave a la hora de diferenciar los *intents* entre sí.

El buen mantenimiento de estas frases es esencial para la correcta detección del *intent* al que corresponde, y para que las entidades recojan correctamente los parámetros. Es importante que las frases de entrenamiento sean variadas, para ello tendremos en cuenta el léxico y las estructuras sintácticas de las frases.

A la pregunta ¿cuánto entrenamiento es necesario?, es tentador basarse en el principio "cuánto más, mejor". Esto es un error. Es cierto que cuantas más frases de entrenamiento haya, será mejor, pero lo realmente importante es la variedad. Sin variedad, una cantidad elevada de frases sólo entorpece el entrenamiento. Hay muchas formas distintas de decir una misma cosa. Por ejemplo, si el usuario quiere pedir una camiseta, puede decir: "quiero una camiseta", "necesito una camiseta" o "enséñame las camisetas que tienes". Las tres frases

hacen referencia a la misma intención, las 3 son buenas opciones de frases de entrenamiento. Sería un error querer poner como frases de entrenamiento: “quiero una camiseta”, “quiero algunas camisetas”, “quiero camisetas”. Una buena herramienta de CLN debe ser capaz de entender las 3 frases una vez ya tiene la estructura, sin necesidad de introducir manualmente cada frase nueva que diga el usuario con cada uno de los parámetros que puedan variar. Es decir, que una vez introducida la estructura “quiero una camiseta”, no es necesario introducir “quiero un pantalón”.

Otra cuestión a tener en cuenta para entrenar un *intent* es que es beneficioso pensar en términos de “cuál es el problema” y no de “cuál es la solución”. Es decir, lo más probable es que el usuario use frases del tipo “¿cuándo va a llegar mi camiseta?” en vez de decir “enséñame cuál es el estado de mi pedido”. El chatbot debe ser capaz de interpretar qué problema tiene el usuario para identificar la solución acorde a su problema (Treml 2021).

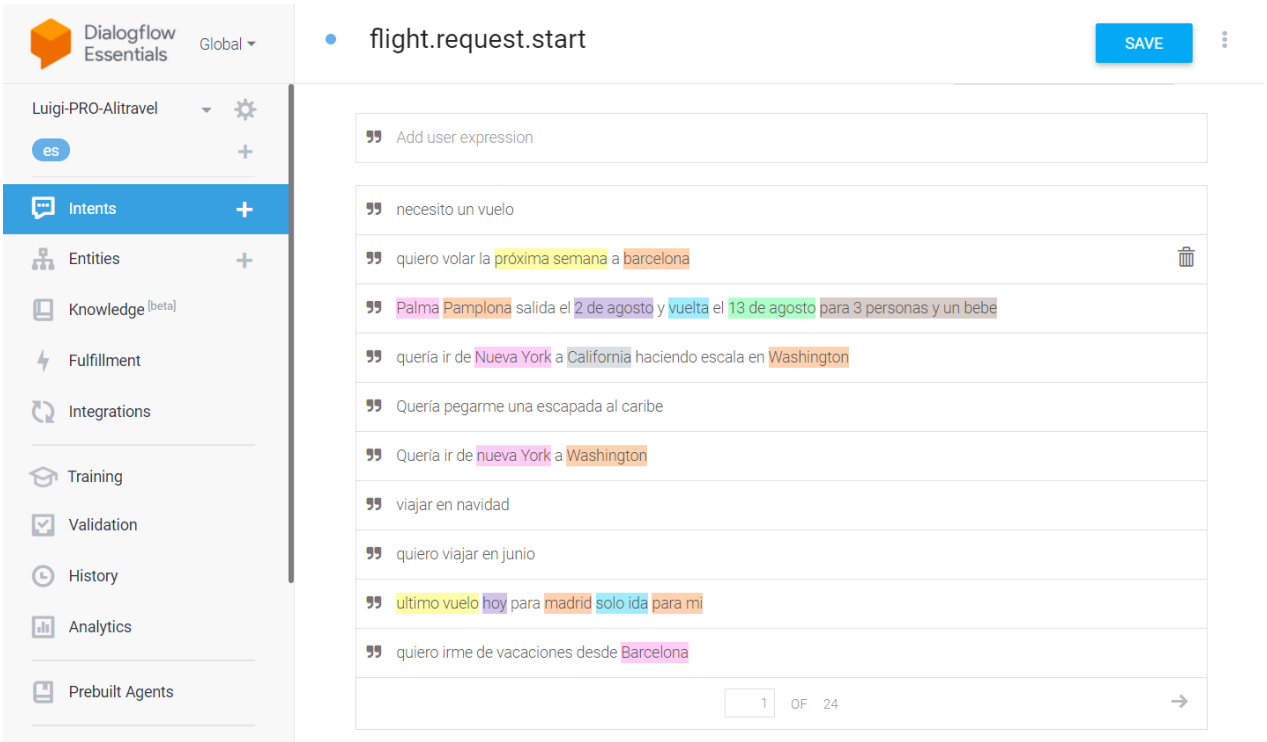

Figura 15. Un ejemplo de frases de entrenamiento de un *intent*

En la figura 15 se puede observar la variedad de frases de entrenamiento del *intent flight.request.start.* Este *intent* es el que recoge la intención del usuario de reservar un vuelo.

Los distintos colores que se ven en las frases son los distintos parámetros que se recogen con distintas entidades. Este *intent* es extremadamente complejo, ya que hay muchos parámetros que pueden recogerse en una misma frase, concretamente 16 parámetros distintos. Esto quiere decir que requiere de mucho entrenamiento, en este momento tiene 200 frases de entrenamiento.

4.2.2.3. Contextos

Los contextos son la memoria del chatbot para mantener el flujo conversacional, permiten tener la conversación bajo control. Gracias a los contextos se puede mantener una conversación con varias interacciones y no se convierte en interacciones de pregunta-respuesta sin ningún hilo conductor. Se puede definir cuál será la vida tendrá el contexto, es decir, cuántas interacciones durará.

Dado que hay muchas frases de entrenamiento que son exactamente igual en varios *intents*, sería imposible mantener una conversación con el chatbot si no hay contextos activados. Por ejemplo, las frases del tipo "no", "si", "no, gracias", se utilizan en muchos *intents* distintos.

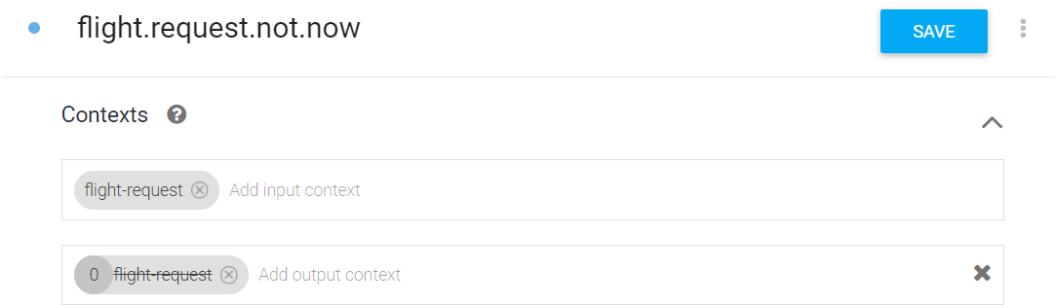

Figura 16. Contextos del *intent: flight.request.not.now*

Cuando el usuario hace una petición, automáticamente se activa el contexto que se ha decidido previamente que se ha de activar si se activa un *intent* determinado.

Para activar los contextos se pueden activar directamente desde el propio *intent* en Dialogflow. Para ello, hay que tener en cuenta el contexto de entrada y el de salida. En la figura 16 podemos ver los contextos del *intent*: *flight.request.not.now.* Este *intent* recoge cuando el usuario quiere dar por finalizada la conversación en mitad de una búsqueda de un

vuelo.

El usuario puede decir, por ejemplo, *"no quiero seguir"*. Se recoge esa intención y se elimina el contexto, como se puede ver en la figura 16. Se hace esto porque el usuario ha pedido terminar la conversación. De esta forma, se limpia el contexto y la conversación puede empezar de 0 si lo que quiere es iniciar una nueva. Es importante que la opción de abandonar la conversación esté siempre disponible para el usuario, de forma que no quede atrapado en la conversación. Se le puede dar la vida que se quiera al contexto de salida, todo dependerá de las necesidades del *intent*, de las interacciones que sean necesarias para resolver la petición.

Activar contextos en un *intent* no es obligatorio. Cuando se trata de *FAQs*, no es necesario activar contextos ya que se tratan de conversaciones pregunta-respuesta, no hay una conversación con más de 1 o 2 interacciones.

Los contextos también se pueden activar desde el código si necesitamos que siga una lógica de negocio. Las automatizaciones desarrolladas por los programadores de las decisiones tomadas por parte del experto en negocio, son activadas o desactivadas en el orquestador (véase figura 9). Estos contextos pasan a Dialogflow para que el flujo de la conversación pueda seguirse correctamente. Dialogflow debe diponer siempre de las actualizaciones que se hagan desde los responsables de negocio, si no es imposible mantener el hilo de la conversación.

Finalmente, otra forma de activar contextos es mediante un evento. Un evento sirve para que se puedan activar contextos sin que el usuario introduzca un texto. Es decir, si el usuario dice "quiero un vuelo", el *intent flight.request* recoge la intención y se activa el contexto *flight-request.* Una vez el usuario ya ha hecho la búsqueda, se debe pasar al *intent flight.availability.* Este *intent* es el que se encarga de enseñar la disponibilidad de vuelos según la búsqueda que ha hecho el usuario, y le da la opción al usuario de pedir otros filtros, de esta forma el usuario puede pedir, por ejemplo, "vuelos solo por la tarde" o "enséñame vuelos más baratos". Para llegar a este *intent* hay que pasar por *flight.request* obligatoriamente, por eso se activa el contexto a partir de un evento. De esta forma, se pone la conversación bajo un contexto, por lo que si el usuario necesita pedir cambios en la fecha o quiere vuelos más baratos, se puede entender lo que quiere y a qué está haciendo referencia.

4.3. Revisión de conversaciones

Para mejorar el sistema, se hacen revisiones diarias de las conversaciones del día anterior. En Luigi, la filosofía es que el usuario marca la evolución del chatbot. Los problemas que se detectan deben ser solucionados. Las peticiones que haya hecho el usuario para las cuales no hay una solución automatizada, o una respuesta, marcan una nueva tarea para el botmaster, que puede ir desde accionar la automatización de procesos que hacemos manuales (Véase apartado 4.2.2.), hasta crear una respuesta para esa petición, si no requiere de desarrollo.

Para esta revisión de conversaciones hay 2 herramientas: la sección *Training* de Dialogflow, y el visor de conversaciones Conversation Viewer, que es un programa creado por el equipo de Quonversa.

En este apartado se explica la utilidad de cada una de estas herramientas, y cuáles son las soluciones que se pueden dar a los problemas que se detecten.

4.3.1. Detección de problemas: Conversation Viewer

Conversation Viewer es un programa creado por el equipo de Quonversa. Este programa permite la visualización de las conversaciones que tienen los usuarios con el chatbot.

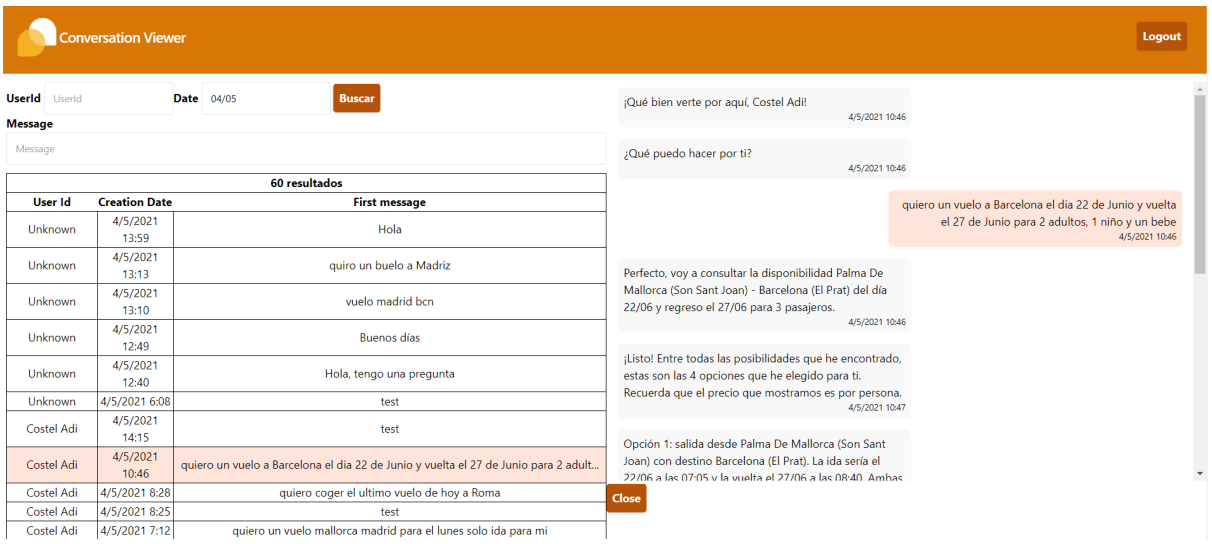

Figura 17. Interfaz del programa Conversation Viewer

En la figura 17 se puede ver la interfaz del Conversation Viewer. A la izquierda se ve el número de resultados de conversaciones en 1 día. Se pueden hacer búsquedas filtrando por fecha y el *UserId*, que es la identificación que se le da al usuario. Esta identificación variará en cuestión de si es un usuario registrado o no. Si está registrado, se le identifica con el nombre por el que se ha registrado. También se puede filtrar por el primer mensaje que escribe el usuario en el chatbot, es decir, el inicio de la conversación.

A la derecha, se puede ver la conversación completa que se selecciona a la izquierda. Los mensajes en naranja son las peticiones o preguntas del usuario. Los recuadros en gris son las respuestas del chatbot Luigi.

Al revisar las conversaciones desde esta plataforma, es muy sencillo detectar si ha habido una conversación poco eficiente, sea por malentendidos, porque el chatbot no ha entendido lo que el usuario le ha pedido (*fallback*), preguntado o contestado, o por fallos técnicos. Un *fallback* es el camino que sigue un mensaje del usuario no entendido por el chatbot. Cuando no se entiende lo que ha dicho el usuario, el chatbot le contesta con el mensaje "Creo que no te entiendo, ¿puedes repetirlo con otras palabras, por favor?". Si se visualiza este mensaje cuando se está revisando una conversación, no siempre es señal de que ha habido un problema. Puede que el usuario no haya contestado a la pregunta que le ha formulado el chatbot, y que su input esté descontextualizado. En ese caso, se le da una nueva oportunidad al usuario para que introduzca de nuevo su respuesta o petición.

En el caso en el que el usuario haya introducido un mensaje que se considera lógico y dentro del contexto de la conversación, y haya saltado el *fallback*, es un problema a resolver.

En el momento que se detecta un problema en la conversación, si no se ve claro cuál está siendo el fallo, se deben hacer pruebas. Para ello, en Luigi está el entorno de *staging*, un entorno de prueba al que solo puede acceder el equipo (Véase figura 18).

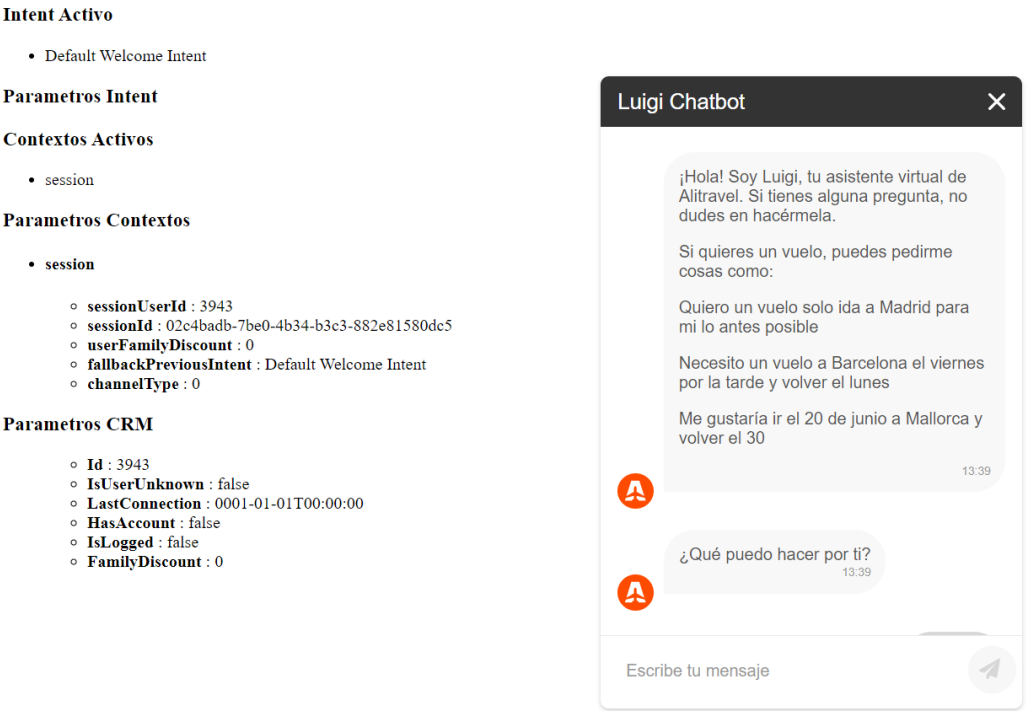

Figura 18. Entorno de *staging* de Luigi

En la figura 18 se puede ver la interfaz del entorno de *staging*. Lo de la izquierda es la información que se necesita para entender qué se está interpretando y qué datos se están recogiendo. Se puede ver el *intent* en ese momento (*intent* activo), los parámetros que se han recogido (parámetros *intent*), qué contextos hay activos (contextos activos), qué parámetros se mantienen mientras el contexto sigue vivo (parámetros contextos) y qué información se tiene del usuario (parámetros CRM).

En este entorno se puede reproducir una conversación ineficiente que se haya detectado en el Conversation Viewer. A medida que avance la conversación, se puede ver de una forma precisa qué se está recogiendo e interpretando, gracias a la información de la izquierda.

4.3.2. Detección de problemas: Training de Dialogflow

En la consola de Dialogflow hay un apartado llamado Training (Véase figura 19). Esta es una herramienta que detecta los *intents* por los que ha entrado cada mensaje del usuario. En el momento que el mensaje no ha sido asignado a ningún *intent,* querrá decir que se ha ido por el *fallback.* Esta herramienta se encarga de detectar cuántas veces ocurre esto.

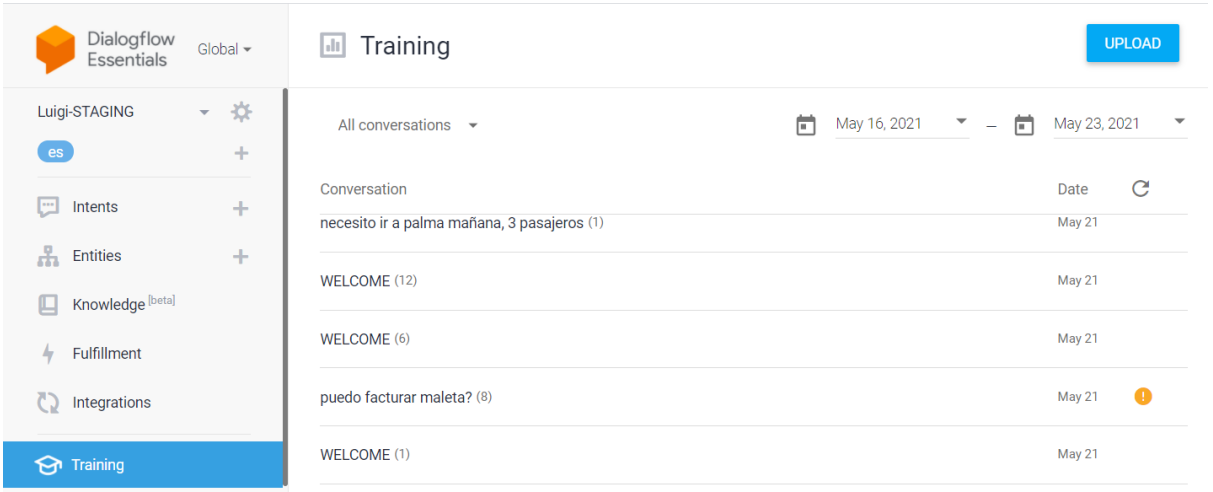

Figura 19. Training en Dialogflow

El problema de esta herramienta es que no aparecen las conversaciones completas, únicamente los mensajes que ha dicho el usuario, el *intent* asignado, y el contexto que se activa en ese momento (Véase figura 20). Por eso, en Quonversa se ha creado el Conversation Viewer.

Figura 20. Visualización de una conversación en el Training

4.3.3. Soluciones

Las soluciones pueden ser muy variadas. Hay soluciones rápidas y simples, y otras mucho más complejas y que requieren de desarrollo. Si el problema se está dando en algo que ya está implementado, y es una cuestión de mantenimiento de frases, entidades o *intents,* la solución se puede aplicar al momento. Si el problema es que no está implementado lo que el usuario pide, entonces habrá que discutir si se quiere o no implementar lo que pide el usuario, y en caso afirmativo, comenzar el proceso para desarrollarlo (Véase 4.2.).

Así pues, las soluciones que el botmaster aplica diariamente son las siguientes:

-Creación de nuevos *intents.* Si el usuario pregunta algo para lo que no hay una respuesta creada, se crea una nueva FAQ (Véase 4.2.1. y 4.2.2.).

-Introducir nuevas frases de entrenamiento en el *intent* que corresponda. El usuario puede introducir una frase que el chatbot no comprende porque no ha entrenado con esa estructura, o tal vez utiliza un léxico que no estaba contemplado. En ese caso, se introduce esa nueva frase de entrenamiento. Esta es la mejor manera de entrenar al chatbot, con frases reales de los usuarios.

-Introducir un nuevo sinónimo en un valor de una entidad (Véase 4.2.2.1.). Si el chatbot ha detectado bien el *intent,* pero no ha conseguido recoger un parámetro o más, el problema puede ser que falte algún sinónimo en un valor de una entidad. Por ejemplo, si el usuario dice: “Quiero coger el próximo vuelo a Madrid”. Si el chatbot no ha recogido la fecha que ha indicado el usuario, probablemente es porque falte introducir “próximo vuelo” como sinónimo de “hoy”. En una situación así, es importante hacer una reflexión previa para decidir cuál va a ser la interpretación que se le quiere dar. En el caso de Luigi, se interpreta de esta manera.

-Cambiar frases desde el CMS (*Content Management System*) o en Dialogflow. CMS es el sistema de gestión de contenido, en Luigi se utiliza Strapi[7]. En esta plataforma están guardadas todas las frases que dice el chatbot que requieren de desarrollo previo. Es decir,

[7] https://strapi.io/

cuando una respuesta tiene que pasar por una lógica de negocio (Véase 4.2.), las frases se guardan en este gestor de contenido. Si son respuestas que no requieren desarrollo, como las FAQs, las respuestas se generan directamente desde Dialogflow. En el caso de que se piense que el problema que se está dando es por un mal planteamiento de una frase, la solución debe pasar por una mejora en la manera de preguntar o informar. Tal vez la frase es muy larga y compleja, el léxico no es el adecuado o se le están preguntando demasiadas cosas a la vez al usuario.

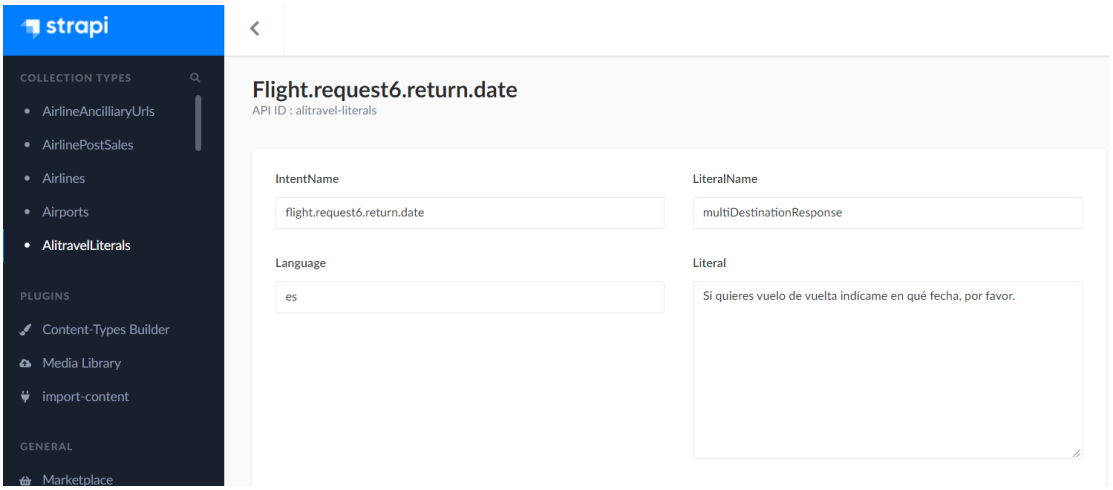

Figura 21. La interfaz del programa Strapi, el CMS de Luigi

En la figura 21 se puede observar la interfaz de Strapi. Concretamente, se puede ver una de las frases del chatbot. En el programa, a las frases se les llama “literales”. Este caso es un buen ejemplo para explicar un cambio en una frase. Anteriormente, esta frase era: “¿Quieres vuelo de vuelta o es solo de ida? Si quieres vuelo de vuelta indícame en qué fecha, por favor”. Resultaba un tanto larga y poco clara, ya que se estaban preguntando dos cosas distintas: qué tipo de trayecto es, y la fecha de la vuelta en el caso de que sea ida y vuelta. Querer aclarar de forma excesiva al usuario lo que esperamos que responda puede ser perjudicial en la comprensión. Confiar en su capacidad comunicativa es mucho más efectivo.

La modificación de este literal se puede hacer directamente en esta interfaz, se hace el cambio y se guarda. Los cambios se ven reflejados automáticamente en el chatbot.

En el caso en que este no haya indicado que el vuelo es solo de ida en su petición principal, ni tampoco haya dicho una fecha de vuelta, ahora Luigi le dice al usuario: “Si quieres vuelo de

vuelta indícame en qué fecha, por favor". De esta forma, el usuario puede contestar una fecha de vuelta, o contestar que "no quiero vuelo de vuelta" o cualquier sinónimo que haga referencia a ello.

-Hacer cambios en el flujo conversacional (4.2.1.). En un caso así, se debe hablar con el programador que haya hecho el desarrollo del flujo y plantear el nuevo cambio. Esta conversación deberá ir acompañada del diseño funcional, es importante documentar siempre los cambios que se hagan.

5. Conclusiones

Durante este trabajo he hecho una introducción al concepto de chatbot, y he clasificado los chatbots según las herramientas de desarrollo que utilizan (si tienen inteligencia artificial o no), y según su uso o no de botones. He explicado las ventajas y desventajas que se consideran a la hora de decidir qué camino coger para empezar a desarrollar un chatbot. Es muy importante tener claro qué camino se escoge, ya que la manera de trabajar e incluso el tiempo que se tardará en desarrollar el chatbot variará dependiendo de qué camino se decide seguir.

En el desarrollo del trabajo, he explicado los perfiles profesionales que intervienen en el desarrollo de un chatbot, y cuál es el papel del lingüista, concretamente en Luigi. He explicado las tareas que llevo a cabo como lingüista en Luigi, y qué herramientas utilizo para ello, especialmente la herramienta Dialogflow. En la explicación de mis tareas, se ve reflejada de qué manera mi formación en el grado de Lingüística suma valor en el desempeño de las tareas, y por qué elegir entonces un perfil lingüista para este puesto.

La creciente implementación de chatbots en las páginas web de las empresas trae consigo nuevos perfiles profesionales, y el lingüista es un profesional reclamado por este sector. Actualmente, la tendencia a comunicarse mediante mensajería instantánea crece en paralelo a una nueva manera de comunicarse. Las aplicaciones como Whatsapp o Telegram, o la comunicación a través de chats se están convirtiendo en una nueva vía de comunicación para las empresas, y la integración de chatbots en estos canales trae consigo el reto de conseguir una comunicación eficiente a través del lenguaje natural.

6. BIBLIOGRAFÍA

Artificial Solutions, 2020. *Chatbots: La guía definitiva*. Recuperado de https://www.artificial-solutions.com/es/chatbots/#1)

Treml, Florian, 2021. *4 DO's and 3 DON'Ts for Training a Chatbot NLP Model*. Recuperado de https://chatbotslife.com/4-dos-and-3-don-ts-for-training-a-chatbot-nlp-model-536f949e21ae