UNIVERSIDAD COMPLUTENSE DE MADRID

FACULTAD DE INFORMÁTICA

GRADO EN INGENIERÍA INFORMÁTICA

DEPARTAMENTO DE SISTEMAS INFORMÁTICOS Y COMPUTACIÓN

Trabajo Fin de Grado

CNV BOT

Un asistente virtual para Comunicación No Violenta

Emilio Chico Muñoz

Tutores:

Ámbar Tenorio Fornés

Directores:

Samer Hassan Collado

Álvaro Embid

Septiembre 2021

Resumen

Conocer la importancia de la Comunicación no Violenta (CNV) en la actualidad es muy significativo, pues es un método de comunicación que facilita la interrelación en la sociedad, por lo que crear un asistente virtual que pueda encarnar este tipo de comunicación supone un proyecto muy interesante.

Los objetivos principales del proyecto son:

- Investigación sobre el funcionamiento de asistentes virtuales.
- Integrar bots en Telegram.
- Desarrollar una aplicación que ayude al usuario a aprender a usar los cuatro pasos que forman la comunicación no violenta.
- Evaluar la adecuación de bots conversacionales para el aprendizaje de CNV.
- Detección de sentimientos y necesidades para guiar conversaciones basándose en el marco de la CNV.

La CNV es una herramienta muy útil que pocas personas conocen y generalmente ligada al mundo de la psicología, al tratarse de un término escondido en la actualidad la gente no suele recurrir a la CNV como solución a los problemas ocasionados en varios aspectos de la vida recurriendo a otros métodos más conocidos.

El estado del arte de este proyecto profundiza en la historia, tipos de asistentes virtuales y aplicaciones de desarrollo para entender su funcionamiento, conocer su estructura y así poder diseñar un asistente funcional orientado a la CNV.

Hemos utilizado una metodología ágil como Scrum ya que se caracteriza por proporcionar un método de trabajo formado por herramientas y funciones que nos ayudan a estructurar el trabajo a la vez que vamos obteniendo resultados.

Una vez establecidos los requisitos sobre los cuales vamos a trabajar, elaboramos el diseño del asistente mediante un diagrama de flujo entre los componentes de la CNV que podemos ver en el anexo y el capítulo de diseño.

Recopilada la información sobre los tipos de Bots y las distintas herramientas generadoras de Bots hemos decidido usar **Rasa Open Source** para el desarrollo (ver capítulo 6) ya que ofrece los elementos necesarios para la creación de un bot capaz de complementar el aprendizaje automático con el reconocimiento del lenguaje.

En cuanto a la arquitectura, es importante explicar las herramientas de las que dispone Rasa, para ello hemos utilizado un diagrama donde se pueden ver los elementos conectados entre sí. Una vez que tenemos un resultado final, comienzan las tareas de evaluación sobre las cuales vamos a poner a prueba nuestro asistente.

En conclusión, se ha creado un proyecto basado en un concepto de comunicación poco conocido como es la CNV con el que se espera que se pueda ayudar a las personas que necesiten una alternativa para afrontar y solucionar los problemas de una manera diferente, de una forma mucho más empática.

Abstract

Knowing the importance of Nonviolent Communication (NVC) today is very significant, since it is a communication method that facilitates interrelation in society, so creating a virtual assistant that can embody this type of communication is a very important and interesting project.

The main objectives of the project are:

- Research on the operation of virtual assistants.
- Integrate bots in Telegram.
- Develop an application that helps the user learn to use the four steps that make up non-violent communication.
- Evaluate the suitability of conversational bots for NVC learning.
- Detection of feelings and needs to guide conversations based on the NVC framework.

NCV is a very useful tool that few people know and is generally linked to the world of psychology, as it is a currently hidden term, people do not usually resort to CNV as a solution to problems caused in various aspects of life by resorting to other more known methods.

The state of the art of this project delves into the history, types of virtual assistants and development applications to understand their operation, know their structure and thus be able to design a functional assistant oriented to the CNV.

We have used an agile methodology such as Scrum since it is characterized by providing a working method made up of tools and functions that help us to structure the work while we are obtaining results.

Once the requirements on which we are going to work have been established, we elaborate the wizard's design using a flow diagram between the CNV components that we can see in the annex and the design chapter.

When the information on the types of Bots and the different tools that generate Bots has been compiled, we have decided to use Rasa Open Source for development (see chapter 6) since it offers the necessary elements for the creation of a bot capable of complementing machine learning with recognition of language.

As for the architecture, it is important to explain the tools available to Rasa, for this we have used a diagram where you can see the elements connected to each other.

Once we have a final result, the evaluation tasks on which we are going to test our assistant begin.

In conclusion, this project has been created based on a little known communication concept such as NVC with which it is hoped that it can help people who need an alternative to face and solve problems in a different way, in a much more empathetic way.

Índice

Capítulo 1: Introducción

El propósito de este capítulo es dar a comprender la causa para elaborar este proyecto y así exponer la relación que tiene con la sociedad a día de hoy.

El uso de los asistentes virtuales llamados chatbots está en auge (Velasquez, 2020) y cada vez son más las empresas que destinan estos desarrollos a formar parte de su negocio, la asistencia al cliente es uno de los campos donde son más utilizados. Gran parte de este éxito recala en la interacción con el cliente, pues estos programas están diseñados para simular una conversación totalmente humana entre éste y la corporación, lo que supone un estímulo tanto a nivel económico como en la aceleración de los servicios.

Aprovechando las características que tienen los bots conversacionales, consideramos que sería muy interesante crear un bot que sirva de herramienta para mejorar la interacción en la sociedad y el desarrollo personal basándose en los pilares de la *Comunicación No Violenta*.

La *Comunicación No Violenta* (CNV) es una propuesta de Marshall Rosenberg que señala la manera de entender la vida desde la honestidad, de manera efectiva y empática, libre de juicios morales y recriminaciones expresándonos más desde el yo, y los propios sentimientos (Rosenberg, 2006).

1.1 Objetivos

Los objetivos principales del proyecto son:

- Investigación sobre el funcionamiento de asistentes virtuales.
- Aprender a utilizar la implementación de Bots en Telegram.
- Desarrollar una aplicación que ayude al usuario a familiarizarse con los métodos que forman la comunicación no violenta.
- Detección de sentimientos y necesidades para guiar conversaciones basándose en el marco de la CNV.
- Evaluar la adecuación de bots conversacionales para el aprendizaje de CNV.

Con el desarrollo de este trabajo el principal problema que se pretende resolver es el de ofrecer una aplicación que sirva para enseñar y aplicar los pasos de la CNV, consiguiendo así darle mayor visibilidad a este tipo de herramienta.

1.2 Estructura del trabajo

La memoria de este proyecto está dividida en capítulos donde el ***primer capítulo*** se inicia con una breve introducción y las motivaciones que me han llevado a realizarlo durante el curso actual. El ***segundo capítulo*** introduce y explica el concepto de comunicación no violenta (CNV) sobre el que nos hemos basado para realizar nuestra aplicación.

En el ***tercer capítulo*** encontramos el estado del arte, este apartado cuenta con toda las información relacionada con la historia de los Bots, las aplicaciones estudiadas para generar asistentes y como se aplican las herramientas de desarrollo personal en la actualidad .

La metodología que se ha utilizado para trabajar en este proyecto está definida en el ***cuarto capítulo***. En el ***quinto capítulo*** se citan los requisitos necesarios para el funcionamiento tanto por el usuario como por el servidor y se detalla el diseño completo realizando un recorrido por la aplicación para mostrar la funcionalidad general.

El ***sexto capítulo*** lo forman las herramientas que hemos decidido utilizar para la generación del asistente.

A continuación, en el ***séptimo capítulo*** se muestra la descripción e implementación del sistema y la arquitectura, donde estructuramos las diferentes partes que la forman explicando en detalle el funcionamiento de cada una.

El ***octavo capítulo*** está formado por las diferentes pruebas de evaluación con usuarios para obtener un feedback sobre el que trabajar para ver la usabilidad del asistente.

Tras finalizar este proyecto surgen futuras posibilidades de mejora y ampliación que se recogen en el ***noveno capítulo,*** además de establecer las conclusiones generales que han surgido tras el desarrollo.

Al final de la memoria podemos ver el anexo formado por una imagen que plasma el diseño de la aplicación y otra imagen referente al diagrama de Gantt. Por último, se encuentra la bibliografía, donde se encuentran las referencias a diferentes fuentes de información citadas en cada capítulo.

Capítulo 2: Comunicación No Violenta (CNV)

"La empatía nos permite percibir de una nueva forma nuestro mundo y movernos hacia adelante" (Comunicación No Violenta-Un Lenguaje de Vida Marshall Rosenberg, 2006).

La CNV es una herramienta de desarrollo personal que tiene como objetivo mejorar la comprensión de los sentimientos y las necesidades con uno mismo y con el resto.

Hace especial énfasis en la escucha profunda, expresión honesta y empatía (ACNV, 2020), la CNV busca prosperar la calidad de la interacción e impulsar una comunicación más empática con los demás y con uno mismo con el fin de aportar más comprensión y empatía a las relaciones humanas. La CNV propone estructurar la comunicación en los siguientes cuatro apartados.

Observar sin evaluar

"Puedo aceptar que me digas lo que hice o lo que no hice. Y puedo aceptar que lo interpretes, pero, por favor, no mezcles las dos cosas" (Comunicación No Violenta-Un Lenguaje de Vida Marshall Rosenberg, 2006).

Al hacer observaciones pretendemos dar una descripción de la forma más objetiva posible sobre lo que ha ocurrido o hemos observado de forma clara y sincera, por ejemplo: ayer vi a tu hermano corriendo por el parque.

"Los juicios que hacemos sobre otras personas son expresiones alienadas de nuestras propias necesidades insatisfechas." (Comunicación No Violenta-Un Lenguaje de Vida Marshall Rosenberg, 2006).

En cambio, al realizar un juicio o evaluación, tendemos a utilizar términos subjetivos con los que reducimos las probabilidades de que reciban lo que realmente intentamos transmitir, por ejemplo: ayer vi a tu hermano enfadado, con prisa y por eso ni me saludó en el parque. Por eso la CNV recalca la separación de estos dos términos.

"Si combinamos la observación y la evaluación seguramente la otra persona escuchará una crítica." (Comunicación No Violenta-Un Lenguaje de Vida Marshall Rosenberg, 2006)

Identificar y expresar los sentimientos

"La persona madura es capaz de diferenciar los sentimientos estableciendo muchos matices, intensos y apasionados o delicados y sensibles, como si fueran los diferentes pasajes musicales de una sinfonía" (Rollo May, 1967).

En CNV se suele hacer referencia a sentimientos cuando nuestras necesidades están satisfechas: feliz, calmado, seguro; y cuando no están satisfechas: asustado, nervioso, triste.

Para ello elaboramos un vocabulario de sentimientos que nos permite nombrar o identificar de forma clara y precisa nuestras emociones, resultando más fácil conectarnos con los demás.

Distinguir los sentimientos de los pensamientos es un ejercicio fundamental en el proceso de CNV, pues con el lenguaje solemos dar pie a confusiones. Cuando expresamos nuestros sentimientos utilizamos palabras que hacen referencia a emociones específicas en lugar de utilizar palabras más generales, por ejemplo: "Me siento bien", puede significar "felicidad" pero da lugar a una interpretación incorrecta sobre los sentimientos que queremos transmitir.

Identificando correctamente los sentimientos pretendemos evitar juicios o falsos sentimientos como, por ejemplo: me sentí "agredido". Además, palabras como "bien" o "regular" impiden que la comunicación refleje lo que sentimos realmente y para ello la CNV posee listas con un gran abanico de estados sentimentales y emociones.

Valorar nuestras necesidades

"Si expresamos nuestras necesidades es más probable que podamos satisfacerlas" (Comunicación No Violenta-Un Lenguaje de Vida Marshall Rosenberg, 2006).

Este apartado consiste en expresar con claridad nuestras necesidades. En CNV una necesidad humana es cualquier cosa necesaria para que el ser humano pueda encontrar plenitud, además, se distinguen varios tipos de necesidades humanas como por ejemplo, de cercanía: afecto, amor; de paz: armonía, belleza; de conexión: aceptación, agradecimiento, la CNV cuenta con una lista de necesidades humanas que todos compartimos para ayudarnos a explicar mejor nuestra situación.

Cualquier crítica o juicio que manifestemos son expresiones de nuestras propias necesidades, por ejemplo, cuando alguien nos dice: "No me entiendes" significa que su necesidad de ser comprendido no está satisfecha.

Cuando manifestamos nuestras necesidades de forma indirecta y nos valemos de juicios, interpretaciones e imágenes, lo más probable es que los demás perciban una crítica y por lo tanto se defiendan, en cambio, cuanto más conectemos nuestros sentimientos con nuestras las necesidades y las expresemos de una forma positiva, más fácil será que respondan de manera compasiva.

Peticiones

El cuarto y último componente de la CNV trata la forma en la que deseamos pedir a los demás o a nosotros mismos para enriquecer nuestra vida.

> *"Las peticiones que no van acompañadas de los sentimientos y necesidades pueden parecer exigencias." (Comunicación No Violenta-Un Lenguaje de Vida Marshall Rosenberg, 2006).*

Las peticiones deben ser claras, realizadas en un lenguaje positivo y concreto ya que si utilizamos palabras vagas o ambiguas puede aumentar el nivel de confusión en la comunicación (Rosenberg, 2006). Debemos evitar recriminar ya que a menudo no somos conscientes de lo que pedimos por lo que debemos tener en cuenta el alcance y la magnitud de dicha petición.

Al realizar una petición también tenemos que abrirnos y aceptar una negativa por respuesta. Además, para asegurarnos de que se ha entendido nuestro mensaje, es recomendable incluir un mensaje de confirmación, en algunos casos bastará con una simple pregunta como: «¿Está claro?». En otros, nos hará falta algo más que una respuesta como: «Sí, te entiendo», para estar seguros de que la otra persona entendió el mensaje.

Por último, agradezcamos a nuestro interlocutor cuando nos confirme el mensaje que recibió y brindemosle empatía al que no desea confirmar el mensaje.

Tras haber estudiado estos cuatro componentes fundamentales de la CNV podemos decir que su objetivo no consiste en modificar a la gente ni cambiar su mentalidad, si no crear relaciones sinceras basadas en la empatía que nos permitan satisfacer las necesidades de todos.

2.1 CNV, enfoques prácticos en la actualidad

Empresas

> *¨La comunicación es para la empresa el equivalente al sistema circulatorio del organismo animal o humano; permite que la sangre, que en este caso es la información, llegue a todos los rincones del cuerpo y les proporcione el oxígeno necesario para su funcionamiento y, por lo tanto, para la supervivencia misma del sistema. Si no hay una buena irrigación, sobrevendrán enfermedades que llevarán finalmente a la muerte¨* (Andrade, 2005).

Existen estudios donde los participantes que estudian las técnicas de la CNV mantienen una actitud más equilibrada y pueden recordar el modelo transcurrido un año del aprendizaje (Baesler & Lauricella, 2014). Esto quiere decir que la CNV fomenta un cambio duradero en las actitudes sobre la comunicación positiva, por ejemplo, un estudio cuantitativo de ejecutivos mostró que seis meses después de la capacitación de las técnicas de CNV, la eficiencia aumentó en un 50-80 % (Connor & Wentworth, 2012). Los efectos a largo plazo del método CNV también incluye una mayor capacidad de resolución de conflictos pacíficamente (Branscomb, 2011; Nash, 2007).

Centros de tratamiento juvenil y cárceles.

En 2007 se llevó a cabo un programa de capacitación de dos años sobre CNV para los integrantes del personal de un centro de tratamiento juvenil en Virginia y encontró una estadística significativa en el aumento de la resolución pacífica de conflictos entre el personal y los residentes, por el contrario, el personal no capacitado en CNV aumentó su tasa de conflictos violentos.

También se ha estudiado el uso de la CNV en centros penitenciarios, existen organizaciones como *Freedom Project*, encargados de capacitar a la población carcelaria en las técnicas de la CNV, el estudio realizado en el complejo correccional de Monroe (Washington) arrojó que los reclusos a los cuales se les aumentó sus horas de entrenamiento, disminuyeron considerablemente su tasa de reincidencia y además obtuvieron datos de un segundo estudio en el que se ve reflejado claramente la reducción de la ira ante la resolución de conflictos entre los empleados y los reclusos (Koopman & Seliga, 2021) (Suarez et al., 2014).

Aulas

La CNV ofrece herramientas para hacer que las clases sean menos violentas y lugares donde se puedan enseñar habilidades para construir la paz, hablar profundamente sobre las emociones sin juzgar y reconocer nuestras necesidades universales junto a las de los demás.

En el estado de Kent (Ohio) se presentó la CNV en una clase de estudiantes universitarios como una forma de fomentar el respeto en las discusiones, el método que emplearon para capacitarlos en CNV fue presentarles distintos temas de discusión como política, drogas y armas. Intercambiando los roles se percataron de que este método no consistía en convencer a alguien para cambiar de opinión sino de establecer una conexión a través de los sentimientos y las necesidades (Koopman & Seliga, 2021).

Una encuesta realizada tras finalizar el semestre arrojó que el 79% de los estudiantes habían utilizado componentes de la CNV fuera de las aulas, aunque estaban lejos de dominar esta habilidad, encontrándola claramente útil (Koopman & Seliga, 2021).

Capítulo 3: Estado del arte de Bots / asistentes conversacionales

En este capítulo se detalla la investigación sobre la historia de los bots conversacionales, tipos de bots, las aplicaciones de psicología y autoayuda además de las diferentes tecnologías que existen en el mercado, estudiando cuál es la que más se adecúa al proyecto y la razón por la que se ha incorporado en este proyecto.

3.1 Historia de los bots conversacionales

Los bots conversacionales no son un invento moderno, pues en 1950 el matemático y científico británico Alan Turing se cuestionó el pensamiento de las máquinas, puede que no fuera el primero en demostrar una fuerte curiosidad al respecto, pero sí el primero en desarrollar un método capaz de medir su inteligencia, el *Test de Turing* (Graham & Dowe, 2020).

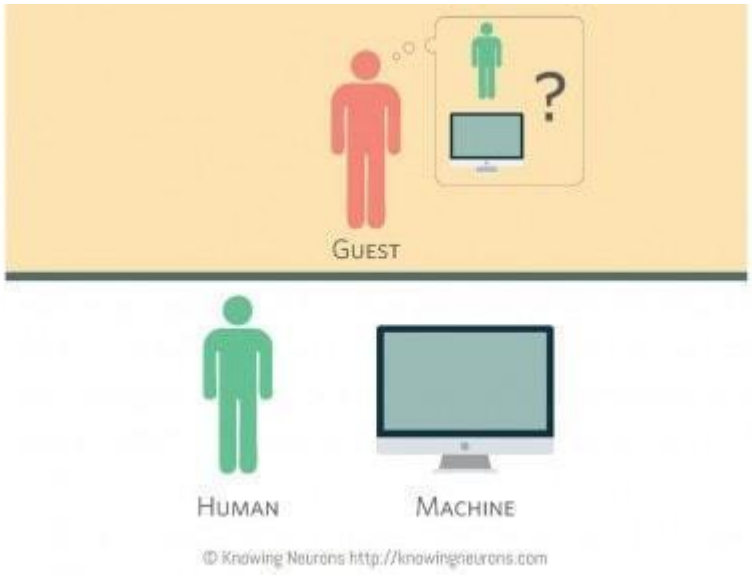


Imagen 1. Turing Test

Con este test, Turing quería comprobar si una máquina puede convencer a una persona de que la conversación se está produciendo entre dos humanos cuando en realidad se está realizando entre un humano y un ordenador.

Eliza, el primer bot

El trabajo de Turing sirvió de inspiración para muchos científicos informáticos como el alemán Joseph Weizenbaum que, en 1966, desde el MIT desarrolló un programa cuya finalidad era engañar a los humanos haciéndoles creer que estaban hablando con otra persona, este programa se llamaría *Eliza* .

El funcionamiento de *Eliza* se fundamenta en el reconocimiento de palabras clave que una vez detectadas servían para elaborar una pregunta a la persona como si de un psicólogo se tratase. Por ejemplo, si alguien mencionara un familiar en una sentencia, respondería pidiéndole que le contase más sobre su familia simulando un estado de entendimiento real (Heung-Yeung et al., 2018).

Imagen 2. Conversación con Eliza

Tras experimentar los desafíos que Turing expuso con su método, *Eliza* se convirtió en el primer bot conversacional de la historia.

Evolución

Durante las décadas posteriores a Eliza, los bots siguieron desarrollándose hasta que en 1989 Jim Aspnes inauguró *TinyMud*, una reimplementación del juego de mazmorras multiusuario *Mud*, de Richard Bartle que incluía conversaciones entre usuarios y escenarios simulados. Muchos de los jugadores controlados por las máquinas se empezaron a llamar bots, por abreviación de la palabra robots.

La mayoría de éstos bots fueron creados simulando el funcionamiento de Eliza, hasta el punto en que si un jugador entraba en una mazmorra determinada podía charlar con ella. Fue aquí cuando se decidió crear un reproductor controlado por una máquina llamado *ChatterBot*, que se encargaría de conversar con otros jugadores proporcionando información sobre el juego a través de su metodología basada en la secuencia If-then-else.

Con el paso de los años se fueron sofisticando, primero con *Alice* (Artificial Linguistic Internet Computer Entity) que en 1995 buscaba una conversación mucho más natural capaz de reunir ejemplos de lenguaje natural y organizarlo en categorías, pero utilizando una estructura muy similar a sus predecesores (Heung-Yeung et al., 2018).

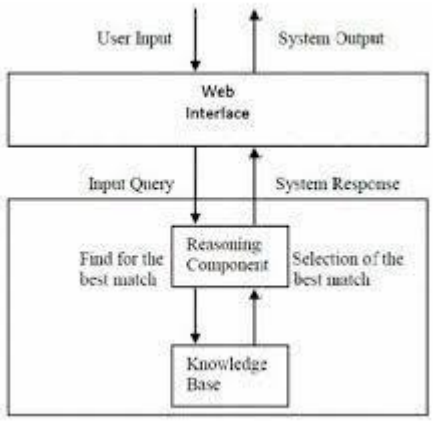


Imagen 3. Estructura Alice

En 1997 la compañía Microsoft crearía su primer agente de conversación disponible para Windows, *Clippy* diseñado para guiar a los usuarios a utilizar las herramientas de Office y posteriormente surgirían bots mucho más desarrollados como *SmarterChild*, considerado el precursor del actual *Siri*, capaz de conseguir conversaciones rápidas en plataformas como MSN Messenger (Klopfenste et al., 2017).

En la actualidad existen bots realmente sofisticados como los conocidos *Siri*, *Cortana*, *Alexa* y *Google Assistant* los cuales poseen integración de voz que sustituye al texto escrito de sus predecesores.

3.2 Tipos de Bots

En este proyecto la palabra Bot aparece constantemente por lo que es necesario conocer qué es un Bot, y qué tipos nos podemos encontrar.

Bot

> *"La palabra "bot" es una reducción de "robot" y los bots pueden considerarse robots incorpóreos. Estos autómatas de software realizan tareas en línea, actuando como sustitutos de los humanos y realizando algunas tareas informativas de memoria."* (Howard, 2018)

Los bots generalmente están diseñados para ahorrar tiempo y energía a un humano, porque analizan y organizan la información a gran velocidad, evitando que los humanos realicen el trabajo. Los primeros bots fueron diseñados por científicos informáticos para realizar tareas regulatorias simples dentro de plataformas cerradas, pero se extendieron rápidamente más allá de las tareas de mantenimiento, en concreto a interacciones sociales que pueden diseñarse.

Existen bots con numerosas funcionalidades, algunos están pensados para ayudar a los usuarios en cualquier tarea cotidiana y otros están diseñados para realizar tareas menos morales, pero igual de eficaces.

Bots Crawlers

Conocidos como *"bots rastreadores"* son los encargados de inspeccionar páginas web que utilizan los buscadores.

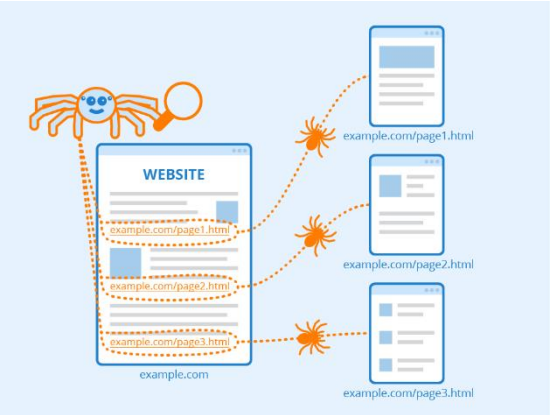


Imagen 4. Bots Crawlers

Estas páginas con motores de búsqueda cuentan con potentes rastreadores que analizan los sitios web creando así una gran base de datos con toda la información recopilada para ofrecerle un contenido más relevante al usuario, tanto es así que la

política planteó la restricción de los contratos que restringen la recopilación de datos. (Rosenfeld, 2001)

Un ejemplo de los bots que están actualmente detrás de los principales motores de búsqueda son: Google (Googlebot), Bing (Bingbot), Yandexbot en Rusia y Baidu Spider en China.

Bots en redes sociales

Las redes sociales forman parte del día a día de las personas y el objetivo de estos bots consiste en potenciar el contenido de las redes sociales consiguiendo elevar el alcance y su repercusión al público.

Tanta es la importancia que se le da, que existen aplicaciones especializadas en la venta de bots para ganar seguidores y bots personalizados para ganar visibilidad en vídeos publicando una gran cantidad de respuestas, sin embargo, nadie sabe cuál es el potencial real de un bot cuyo propósito es controlar una comunidad, manipular los datos o influir en el comportamiento de los usuarios.

A pesar de que los bots son fuente de muchos ataques peligrosos como los mencionados en el párrafo anterior y portadores de mensajes no deseados como el spam, son una herramienta muy valiosa para los analistas que ejecutan continuamente recopilaciones de datos (Aiello et al., 2012).

ChatBots

> *"Un chatbot se define como un programa informático diseñado para simular una conversación con usuarios humanos, especialmente a través de Internet".*

Este tipo de bots están diseñados para mantener una conversación mediante lenguaje natural, especialmente por internet, ofreciendo respuestas previamente estructuradas en función de las intenciones del usuario. Se caracteriza por utilizar en la mayoría de su desarrollo la inteligencia artificial (IA).

Imagen 5: ChatBot

Esta inteligencia es la encargada de analizar el sentido de dicha conversación y no sólo para generar una respuesta, si no para elaborar todo tipo de informes sobre los posibles clientes del entorno donde esté integrado (Adamopoulou, 2020).

VoiceBots

Son los más conocidos como asistentes virtuales por voz, su función es guiar al usuario y resolver sus dudas mediante una conversación como lo haría cualquier persona. Muchas empresas como Microsoft, Apple y Amazon han optado por incluir este tipo de bots como son Cortana, Siri y Alexa para conseguir una gran experiencia de servicio.

Su creación ha permitido que las empresas puedan asegurar una atención de calidad y soporte al usuario 24 horas al día en cualquiera de sus consultas, según Forbes, el 85% de las interacciones relacionadas con la atención al cliente se van a desarrollar a través de bots (Brandtzaeg, 2017).

Otros tipos de bots

Existen otros tipos como los *bots Spam*, encargados del envío masivo de publicidad muy presentes en todo tipo de plataformas, los *bots Hackers* cuyo diseño está destinado a recopilar información valiosa de los usuarios como contraseñas, correos, imágenes, vídeos, contactos y los bots integrados en juegos que básicamente están configurados para simular movimientos y jugar contra ellos.

3.3 Plataformas generadoras de bots

En primera instancia, sin conocer cómo íbamos a crear nuestro asistente, necesitábamos una herramienta capaz de generar un bot de manera sencilla por lo que recurrimos a varias aplicaciones que no nos ofrecieron el resultado esperado, las explicamos a continuación.

Los chatbots no sólo son herramientas de comunicación pues también se pueden desarrollar con diferentes finalidades. Debido a este gran potencial están haciendo una gran incursión en el mundo del comercio pues cada vez son más las empresas que optan por sus servicios.

ChattyPeople

ChattyPeople es una plataforma de inteligencia artificial con la que crear un bot de texto y voz cuya característica principal a parte de la simplicidad, pues no requiere tener conocimientos de programación, es la integración en el comercio electrónico.

Cuenta con diferentes funcionalidades como hacer que un bot sea tan simple como responder a unas preguntas como si de un servicio al cliente se tratase o integrarlo con otras aplicaciones para monetizar tu propia página web. Empresas punteras como Facebook se benefician de sus servicios y actualmente ha sido absorbida por Mobile Monkey debido a su gran crecimiento y versatilidad en el sector financiero. En este caso un bot comercial no nos ofrece los resultados esperados por lo que ésta alternativa dejó de ser una opción.

Botsify

Botsify es una plataforma de chatbot completamente administrada capaz de crear asistentes virtuales inteligentes para webs, páginas de Facebook y aplicaciones de mensajería. También se caracteriza por destacar con los negocios pues está orientado a captar clientes con un diseño interactivo y personalizado. Su funcionamiento destaca por su estructura categórica e intuitiva de"*arrastrar y soltar*", además del uso de formularios los cuales almacenan la información del usuario en una hoja de cálculo o en su propia CRM. También incluye herramientas para personalizar todos los aspectos de la interfaz y analizar en profundidad los resultados obtenidos del chatbot creado.

"El objetivo es mantener la propiedad de los chatbots dentro del segmento empresarial de las organizaciones" (Co-Ceo at City Innovate Jay Nath, 2018).

Es capaz de conversar en más de 190 idiomas y actualmente más de 80.000 empresas están utilizando la experiencia de chat con esta plataforma.

Chatfuel

Se trata de una empresa creada en 2015 cuya finalidad es facilitar el desarrollo de los chatbots a cualquier persona. Su funcionamiento se caracteriza por la división del panel de control ya que está organizado de tal manera que recoge unas reglas conversacionales que tendremos que detallar para que reconozca las sentencias del usuario. Además, no es necesario tener alguna experiencia en la codificación de código para elaborar un bot en esta plataforma. Hoy día, es la plataforma líder en Facebook Messenger.

3.4 Desarrollo de bots en docencia.

Aunque la educación presencial sigue siendo mayoritaria que la educación a distancia, las metodologías están cambiando haciéndose notar en los entornos web o sistemas de gestión de cursos (LMSs). (D.knowlton,2000).

Debido a su popularidad y proliferación en las universidades y otros centros educativos, se han recogido informes que indican las ventajas pedagógicas que ofrecen sobre los alumnos y educadores. Estos estudios indican también que el método tradicional supone una carga de trabajo excesiva para los educadores, repercutiendo en el tiempo de respuesta.

Una de las opciones que proponen estos entornos web son los foros como método de resolución de dudas permitiendo fomentar la participación del alumnado pero no siempre garantiza su comprensión por lo que hay estudios que proponen el uso de bots para solucionar este tipo de problemas.

3.5 Psicología y autoayuda

En psicología, se denomina autoayuda al soporte que una persona se ofrece a sí misma cuando atraviesa una situación complicada o con la intención de obtener un bienestar personal sin tener que acudir a psicólogos u otra ayuda externa durante dicho proceso.

Los mensajes de autoayuda se remontan a tiempos donde no existía la escritura, pero la primera referencia conocida del término aparece en un libro publicado en 1859 llamado *Self-Help* (Samuel Smiles), basado en un curso que el propio autor había dado a trabajadores donde ofrecía pautas y consejos para tener éxito.

Herramientas de autoayuda

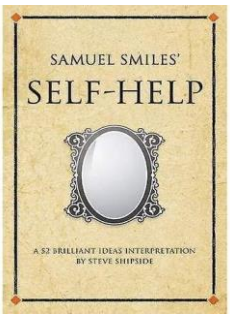
En los últimos años han aparecido gran cantidad de métodos que ofrecen ayuda al desarrollo personal de manera específica. En este proyecto hemos realizado un trabajo de investigación sobre las que describimos a continuación.

Libros

Los libros de psicología y autoayuda son herramientas que pretenden mejorar las emociones y actitud de forma independiente, además de ofrecer una lectura inspiradora buscan la conexión con nuestras preocupaciones y lo que sucede a nuestro en nuestro entorno.

A continuación, detallamos dos libros referencia. Como hemos mencionado antes, este fue el primer referente de ayuda sobre el papel.

Self Help, Samuel Smiles



"Esto, sobre todo: sé fiel a ti mismo, y a eso seguirá, como la noche al día, que no podrás ser falso para nadie. "(William Shakespeare, Hamlet, 1603).

Esta frase resume el libro que se convirtió en la piedra angular de una corriente que no ha dejado de crecer hasta nuestros días, está considerado el primer libro de autoayuda que dio con la fórmula del género; la descripción de una serie de historias reales de éxitos y fracasos para obtener de ellas lecciones que logren el éxito humano.

Poder Sin Límites, Anthony Robbins

Se trata de uno de los referentes dentro del género del desarrollo personal que nos invita a la acción, hace referencia a las barreras del poder de la mente y posibilidad de alcanzar aquello que deseamos.

Aplicaciones en psicología

El rápido desarrollo de las computadoras y la tecnología de la información durante la última década ha tenido un impacto en la psicología, que en este contexto ha pasado de las aplicaciones informáticas locales a las aplicaciones de red que aprovechan Internet (Barak, 2005).

Las aplicaciones sobre psicología que hemos estudiado y podemos encontrar en el mercado de aplicaciones actual nos invitan a realizar actividades para sentirnos bien, centrándose en mantener hábitos de vida saludables para conseguir una estabilidad física y mental e incluso experimentando terapias online, donde es posible realizar sesiones terapéuticas por medio de estas aplicaciones.

Varios ejemplos que podemos encontrar sobre aplicaciones de terapia psicológica son "*Mentavio*", plataforma que ofrece sesiones psicológicas online, terapia a través de webcam, chat o e-mail y coaching, y "*TherapyChat*", que es una aplicación de terapia online destinada a compañarte en diferentes problemas psicológicos como la ansiedad, depresión y autoestima entre muchos mas.

Aplicaciones sobre desarrollo personal y autoayuda

La última década ha sido partícipe del auge de las aplicaciones de autoayuda basadas en internet, en el campo de la salud mental. Muchas tienen como objetivo proporcionar información y estrategias para empoderar a las personas con autoayuda, pero solo unas pocas aplicaciones agregan un componente de contacto profesional o algún tipo de soporte para complementar las estrategias sugeridas en estas aplicaciones.

"*Fabulous:¡Motívame!*" es una aplicación caracterizada por ofrecer un plan en el que te propondrán metas e irás superando objetivos inteligentes y saludables con un enfoque científico para inculcar hábitos saludables en tu día a día y así conseguir una plena salud física y mental.

Guías de autoayuda

Se denomina guía de autoayuda a aquella aplicación que cuenta con una serie de pautas que nos sirven de ayuda para enriquecer y mejorar nuestro equilibrio mental. Muchas de estas aplicaciones proporcionan una evaluación de los problemas y una

estimación de su gravedad para posteriormente elaborar un plan orientado hacia un objetivo constructivo.

Desde un punto de vista médico, la elección de este tipo de aplicaciones supone un desafío pues cada aplicación tiene un alcance y una gestión de la motivación diferente dificultando conocer cuándo avanzar hacia una intervención de mayor necesidad.

Para encontrar una aplicación adecuada una estrategia común es mirar las más valoradas en los mercados de aplicaciones, pero existen organizaciones encargadas de valorar las aplicaciones más útiles a través de una serie de calificaciones internas, por ejemplo: PsyberGuide (Mehrotra et al., 2017).

Una de las aplicaciones más valoradas y útiles según estudios realizados *(Possemato, Kuhn, Johnson, Hoffman y Brooks, 2017)* es Post-Traumatic Stress Symptoms Coach (PTSD Coach).

"*PTSD Coach*" es una aplicación móvil (app) diseñada para ayudar a las personas que tienen síntomas de trastorno de estrés postraumático (PTSD) a comprender mejor y a controlar sus síntomas .

Otra de las aplicaciones que mayor valoración ha obtenido de este tipo de organizaciones se trata de "*MyLife Meditation: Meditate, Relax & Sleep Better*" es una aplicación que proporciona atención especializada desarrollando hábitos simples para poder encontrar un mejor estado mental.

Apps de CNV

Actualmente existen muy pocas aplicaciones centradas en los métodos que recoge la CNV, y mucho menos ofrecen algún tipo de interacción para completar los pasos que la componen. Investigando el mercado de aplicaciones podemos encontrar una que ofrece algunos recursos que se recogen dentro de la comunicación no violenta a modo de información, llamada *Ser Girafa* y otra aplicación más teórica llamada *Las bases de la CNV.*

Ser Girafa – CNV

Ser Girafa es una aplicación que muestra los cuatro componentes que forman la CNV, primero te permite realizar una observación y tras la que podemos elegir si nos produce un sentimiento positivo o negativo, en cada caso muestra una lista de sentimientos donde tenemos que buscar cual es el que experimentamos. Una vez encontrado, elegimos nuestra necesidad a través de una lista de necesidades universales. En el siguiente paso, nos invita a realizar una petición que pueda satisfacer nuestra necesidad. Para finalizar, muestra un resumen con todas las elecciones y los datos introducidos.

Las bases de la CNV

Se trata de una aplicación teórica dividida en cinco apartados, el primero define los cuatro pasos de la CNV, el segundo nos muestra una lista de sentimientos, el tercero muestra la lista de necesidades universales, en el cuarto apartado podemos ver los principales problemas de la comunicación en la CNV y en el quinto, los juicios enmascarados.

Capítulo 4. Metodología

4.1 Scrum

Se ha decidido utilizar una metodología ágil en todo el proceso de desarrollo ya que un método iterativo nos ofrece una mayor flexibilidad para realizar cambios a medida que vamos avanzando y modificar los requisitos y especificaciones que se hayan establecido.

El marco de trabajo que más se ajusta a nuestro proyecto es Scrum ya que nos proporciona resultados de forma rápida para así poder modificar el desarrollo del proyecto según el feedback obtenido.

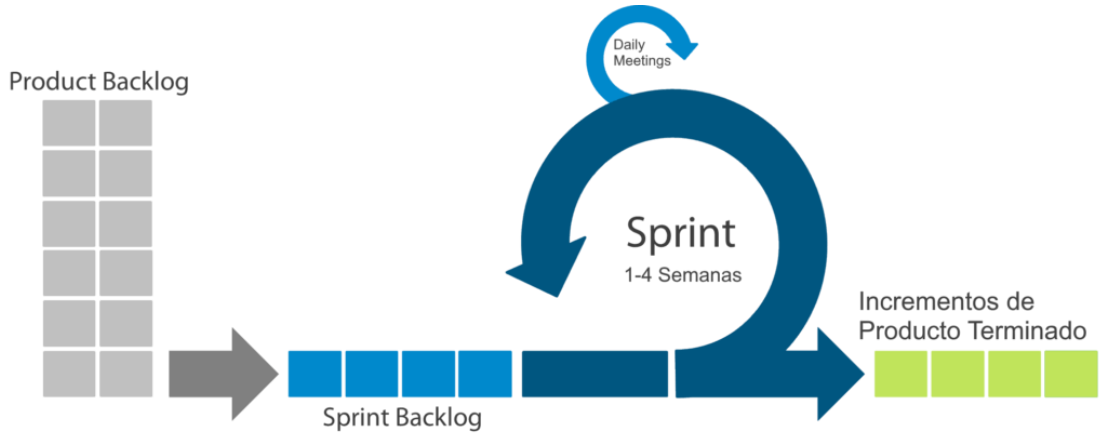


Imagen 6. Scrum

Para definir esta metodología es necesario conocer cómo hemos encajado nuestro proyecto en cada uno de sus apartados.

Product backlog es una lista de elementos necesarios para el funcionamiento del asistente, en nuestro caso se trata de una lista de requisitos formados en un principio por los requisitos mejor atendidos que posteriormente evolucionarán según crezca el proyecto y su entorno. Los requisitos están definidos en el siguiente capítulo.

El *Sprint Backlog* lo forman la lista de elementos anteriores que han sido seleccionados para el Sprint, es decir, se decide qué funcionalidad pasará a formar parte del siguiente incremento y el trabajo necesario para terminarlo.

El *Sprint* es la parte de la metodología donde se sigue el desarrollo de la funcionalidad elegida para formar parte del incremento, estos *Sprints* suelen tener una duración de 1 a 4 semanas, como es nuestro caso, y además ofrece la posibilidad de hacer reuniones diarias para proyectar la posibilidad de conseguir el objetivo del *Sprint*.

Para terminar, el *incremento del producto terminado* es la suma de todos los Sprints terminados y tiene que estar completamente funcional independientemente de si el dueño del producto decide liberarlo o no.

Para ver de una forma más clara cómo se han desarrollado estos hitos, mostramos el inicio del proceso de la metodología en una tabla tipo Scrum.

Pendiente	En progreso	Terminado
Asistente CNV (RF-01)		
Detectar Observación (RF-02)		
Detectar Sentimiento (RF-03)		
Detectar Necesidad (RF-04)		
Detectar Peticion (RF-05)		
Detectar evaluacion oculta (RF-06)		
Mostrar resumen (RF-07)		
Generar camino mínimo (RF-08)		
Guardar Slots (RF-09)		
Funciones auxiliares (RF-10)		
Conexión con Telegram (RF-11)		

Imagen 7. Tabla Scrum

Esta tabla se caracteriza por mostrar los tres apartados, *inicial o pendiente*, donde se encuentran de inicio todos los requisitos que hemos establecido, *En progreso,* que significa que la tarea se encuentra en desarrollo y *Terminado,* cuando se ha validado el trabajo y finalmente se ha añadido al incremento. Podemos ver como avanza la tabla en función de las necesidades del proyecto en la siguiente imagen.

Pendiente	En progreso	Terminado
Asistente CNV (RF-01)		
	Detectar Observación (RF-02)	
		Detectar Sentimiento (RF-03)
	Detectar Necesidad (RF-04)	
Detectar Peticion (RF-05)		
Detectar evaluacion oculta (RF-06)		
Mostrar resumen (RF-07)		
Generar camino mínimo (RF-08)		
	Guardar Slots (RF-09)	
	Funciones auxiliares (RF-10)	
		Conexión con Telegram (RF-11)

Imagen 8. Tabla Scrum con avance

4.2 Enfoque y desarrollo

Se ha decidido seguir un enfoque de desarrollo abierto, al tener el código libre en un repositorio Github bajo licencia GPL (General Public License), con el siguiente enlace: https://github.com/emicm/CNVBot.git.

En todo el ámbito de este proyecto hemos decidido utilizar programas open source, que no requieran ningún tipo de coste para generar la aplicación, por lo que se han descartado varias de las aplicaciones generadoras de Bots estudiadas al requerir de acceso premium para la utilizar la plenitud de sus funciones.

Para ver mejor la planificación de la metodología es necesario ver donde hemos aplicado todos los esfuerzos. Las tareas principales son:

- **Reuniones**: reuniones periódicas entre el tutor para establecer los requisitos.
- **Estudio**: una vez se obtienen los requisitos, son objeto de estudio para evaluar riesgos y proponer alternativas.
- **Desarrollo y Pruebas**: forma la parte más larga y costosa del proyecto, es donde se realiza el código del asistente y se realizan las pruebas pertinentes para comprobar su funcionamiento.
- **Documentación**: Es la parte destinada a documentar toda la información obtenida durante las diferentes fases del proyecto para elaborar la memoria.

A continuación, vamos a mostrar la planificación de una forma más detallada.

Design Thinking

Los inicios de este proyecto comienzan por la asistencia a reuniones semanales donde Álvaro, tutor, director y formador CNV nos acercó este tipo de concepto que desconocíamos. Tras estudiar y entender cómo funciona, estructuramos cada parte y pusimos en común los conocimientos, ideas e investigaciones de cada uno.

Una vez establecidas las bases sobre las que íbamos a trabajar, tuvimos que definir los objetivos principales y requisitos necesarios para alcanzarlos. Para ello utilizamos Design Thinking, una metodología de diseño que se centra en ofrecer una solución real ante la necesidad del usuario.

Realizado el paso anterior, es importante conocer el marco sobre el que vamos a trabajar detectando las necesidades de los usuarios que vayan a utilizar nuestra aplicación.

El siguiente paso de esta metodología consiste en definir lo que realmente nos aporta valor de toda la información recogida, y así poder aproximar el alcance del proyecto.En

nuestro caso esta información está formada los diferentes componentes de la CNV y las herramientas que existen a la hora de desarrollar un bot.

Una vez ideado un plan de seguimiento, para ello se realizó un estudio sobre las diferentes tecnologías actuales para poder crear un prototipo como marca esta metodología y comprobar los fallos de diseño antes de la implementación.

Los primeros prototipos de nuestro proyecto fueron sobre la interfaz de usuario, en las reuniones con el tutor y el director, formador de CNV, planteamos el diseño en papel utilizando dibujos y material de diseño ya que es la parte que más se puede modificar a priori y que caracteriza a Design Thinking.

Al tratarse de un asistente conversacional sobre la CNV, este asistente debe caracterizarse por ofrecer un comportamiento empático, sin dar la sensación de que es una máquina por lo que la detección de sentimientos y necesidades es uno de los apartados más complejos de nuestro proyecto, y para ello hemos utilizado uno de los componentes que ofrece Rasa, relacionado con el reconocimiento del lenguaje como es Rasa NLU.

En cuanto a la elección de la aplicación encargada de albergar el Bot hemos elegido Telegram ya que ofrece una conexión con Rasa Open Source a través de su configuración, además al ser una aplicación de mensajería instantánea, cumple la función de dar más visibilidad a la herramienta CNV.

Para finalizar con la metodología de diseño, tenemos que testear el producto final, durante esta fase se ha testeado con un conjunto de usuarios que no estaban familiarizados con el proceso CNV.

En cuanto a las pruebas con usuario (ver capítulo 9), se han llevado a cabo varias reuniones con 10 participantes, en la primera se les explicó el proceso CNV y se les dió vía libre para interactuar con el asistente, y en la segunda, se les pidió que utilizaran el lenguaje recogido por la CNV para realizar el recorrido.

Para documentar bien los resultados, se les pidió a los participantes que nos dieran feedback de la aplicación, y si les gustaría que se introdujera algún cambio. Estos resultados se incorporan en nuestro proyecto a la hora de pensar en el trabajo futuro que puede desempeñar nuestro asistente, así como las modificaciones realizadas.

Capítulo 5: Análisis y Diseño

Dentro de este capítulo vamos a definir la funcionalidad del proyecto estableciendo los requisitos de software e instalación sobre los cuales hemos desarrollado este proyecto y el diseño correspondiente.

5.1 Ámbito del sistema

El objetivo del proyecto es crear un bot basado en la CNV dentro de la plataforma de mensajería Telegram aprovechando que su API permite el desarrollo de asistentes. Se trata de una aplicación totalmente online cuya finalidad es guiar al usuario mediante los pasos fundamentales de la CNV para así proporcionarle la ayuda que necesite en ese momento.

El propio bot es el encargado de recopilar la información mediante funciones internas y herramientas de reconocimiento del lenguaje natural para así decidir en qué punto de la conversación se encuentra y mostrar la respuesta más adecuada.

5.2 Requisitos

En el siguiente apartado vamos a definir los requisitos de software e instalación que han sido necesarios para el desarrollo del proyecto que posteriormente nos permitirán realizar pruebas de desarrollo para demostrar si el asistente los satisface o no.

Requisitos de Software

A continuación, vamos a mostrar los requisitos necesarios para el correcto desarrollo de nuestro asistente, diferenciando entre requisitos funcionales, que especifican la funcionalidad del sistema y no funcionales, que son aquellos que especifican las restricciones que debe cumplir el sistema.

Tabla 1. Definición de requisitos.

Identificador	Identificador del requisito
Título	Nombre
Tipo	Categoría: Funcional o No Funcional
Origen	Origen del requisito
Descripción	Explicación detallada
Verificación	Proceso que se ha seguido para comprobar su funcionamiento
Prioridad	Indica la prioridad, siendo alta, media o baja

A continuación, se detallan los requisitos siguiendo la notación de la tabla anterior.

Tabla 1.1. RF-01

Identificador	RF-01
Título	Asistente CNV
Tipo	Funcional
Origen	Tutor
Descripción	El asistente deberá guiar al usuario por los apartados que forman la CNV.
Verificación	El usuario hará que el asistente pase por los puntos principales.
Prioridad	Alta

Tabla 1.2. RF-02

Identificador	RF-02
Título	Detectar Observación
Tipo	Funcional
Origen	Tutor
Descripción	El asistente deberá detectar una observación recogida por el proceso CNV realizada por el usuario.
Verificación	El usuario introducirá una observación por mensaje de texto.
Prioridad	Alta

Tabla 1.3. RF-03

Identificador	RF-03
Título	Detectar sentimientos
Tipo	Funcional
Origen	Tutor
Descripción	El asistente deberá detectar los sentimientos registrados por la CNV.
Verificación	El usuario introducirá un sentimiento por mensaje de texto.
Prioridad	Alta

Tabla 1.4. RF-04

Identificador	RF-04
Título	Detectar necesidades
Tipo	Funcional

Origen	Tutor
Descripción	El asistente deberá detectar las necesidades universales recogidas en el proceso CNV.
Verificación	El usuario introducirá una necesidad por mensaje de texto.
Prioridad	Alta

Tabla 1.5. RF-05

Identificador	RF-05
Título	Detectar Petición
Tipo	Funcional
Origen	Tutor
Descripción	El asistente deberá detectar la petición realizada por el usuario siguiendo el proceso CNV.
Verificación	El usuario realizará una petición al asistente.
Prioridad	Alta

Tabla 1.6. RF-06

Identificador	RF-06
Título	Detectar Evaluación oculta
Tipo	Funcional
Origen	Alumno
Descripción	El asistente deberá detectar cuando el usuario introduzca una evaluación oculta.

Verificación	El usuario introducirá una evaluación oculta recogida por la CNV.
Prioridad	Media

Tabla 1.7. RF-07

Identificador	RF-07
Título	Mostrar resumen
Tipo	Funcional
Origen	Tutor
Descripción	El asistente deberá mostrar un resumen de los datos introducidos por el usuario.
Verificación	El usuario terminará el recorrido por el asistente para que muestre el resumen.
Prioridad	Alta

Tabla 1.8. RF-08

Identificador	RF-08
Título	Generar camino mínimo
Tipo	Funcional
Origen	Alumno
Descripción	El asistente deberá seguir un recorrido mínimo previamente establecido.
Verificación	El asistente deberá empezar reconociendo el saludo y terminar por el resumen o despedida.
Prioridad	Media

Tabla 1.9. RF-09

Identificador	RF-09
Título	Guardar Slots (Secciones de memoria)
Tipo	Funcional
Origen	Alumno
Descripción	El asistente deberá guardar el contenido de las entidades.
Verificación	En vista del desarrollador se podrán ver las entidades reconocidas.
Prioridad	Alta

Tabla 1.10. RF-10

Identificador	RF-10
Título	Funciones de respuesta personalizada
Tipo	Funcional
Origen	Alumno
Descripción	El asistente deberá ejecutar las funciones correctas según el camino establecido.
Verificación	El usuario recorrerá el asistente para que ejecute las acciones definidas.
Prioridad	Media

Tabla 1.11. RF-11

Identificador	RF-11
Título	Conexión con Telegram
Tipo	Funcional

Origen	Alumno
Descripción	El asistente deberá mantener una conexión estable con Telegram cuando expongamos nuestra URL en el servidor local.
Verificación	El usuario podrá abrir Telegram en cualquiera de sus dispositivos e iniciar el asistente.
Prioridad	Alta

Tabla 1.12. RNF-12

Identificador	RNF-012
Título	Rasa
Tipo	No Funcional
Origen	Alumno
Descripción	El asistente está creado mediante Rasa Open Source.
Verificación	No está disponible.
Prioridad	Alta

Una vez mostrados los requisitos de software, citamos los requisitos de instalación necesarios para poder ejecutar el asistente.

Requisitos de instalación

Dentro de este apartado vamos a hablar de los requisitos de instalación mínimos para un correcto funcionamiento. Los vamos a dividir en dos partes, los requisitos por parte del usuario para que lo pueda ejecutar, y los requisitos por parte del servidor para que se pueda instalar.

Requisitos por parte del usuario

- Tener una cuenta de usuario en Telegram.
- Aplicación de Telegram actualizada a la última versión.
- Conexión a internet estable.

Requisitos por parte del servidor

- Basta con instalar la versión 1.8.3 o superior de Rasa Open Source.
- Requiere Python 3.6, 3.7 o 3.8.
- Requiere Ngrok 2.3 en adelante.

5.3 Diseño de los componentes de la aplicación

Los pilares de la CNV (Observación, Sentimientos, Necesidades y Peticiones) forman la estructura principal del diseño de nuestro proyecto pues la intención que tenemos es que el usuario siempre pueda encontrar una respuesta acorde a sus necesidades. A continuación, se muestra la explicación del diagrama de la aplicación donde se ve la estructura que hemos desarrollado y que podemos ver con detalle en el anexo.

El ***inicio*** de la aplicación comienza ejecutando el bot en Telegram, este mostrará un mensaje de bienvenida al usuario preguntándole cómo se encuentra para así poder iniciar una conversación y seguir el diagrama de flujo en función de las intenciones del usuario.

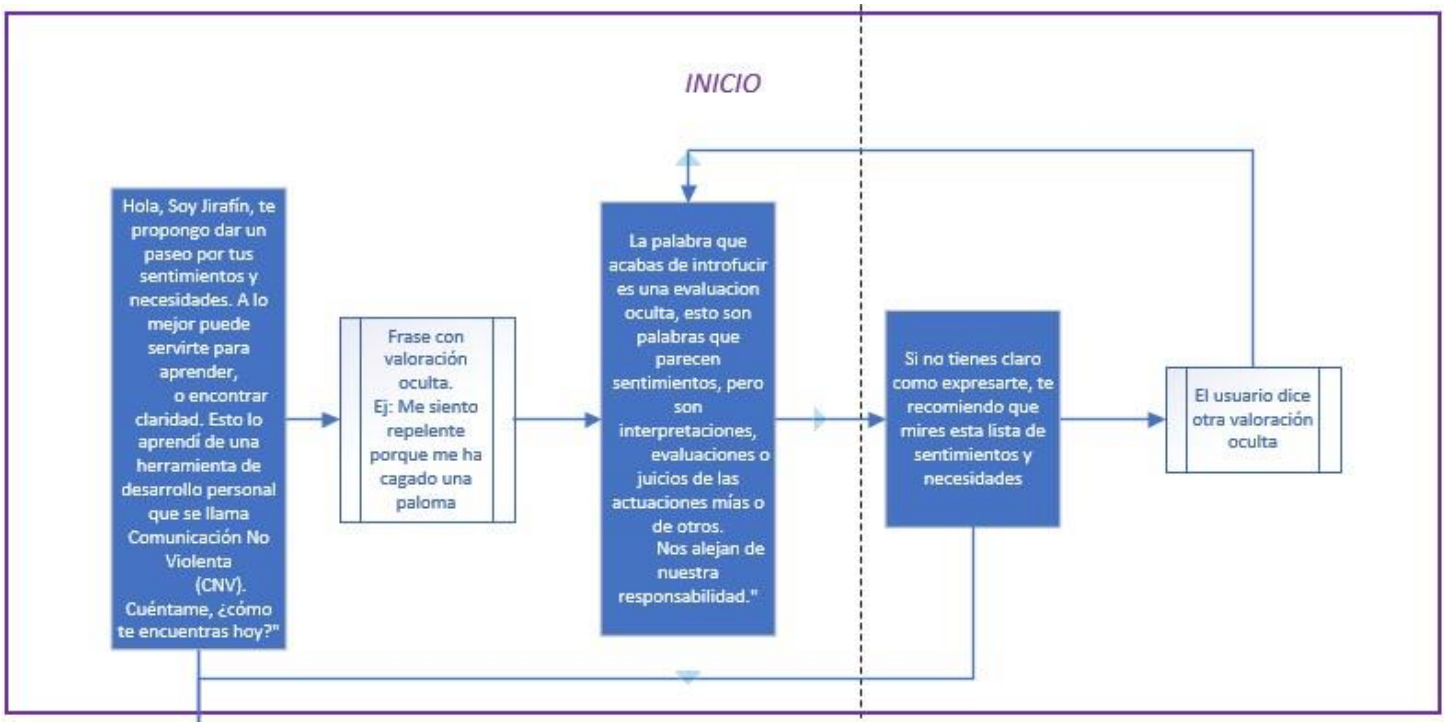


Imagen 9: Diseño saludo

Para ver mejor cómo se integra este flujo de conversación en nuestro asistente, lo vamos a acompañar con una serie de capturas que se corresponden con cada parte del proceso. A continuación, vemos la ventana resultante de iniciar el bot, mostrando el saludo.

Imagen 10: Diseño saludo interfaz

Es posible que el usuario al principio no sepa plasmar sus emociones en texto y es por eso que es necesario crear un apartado donde le guiemos para una correcta expresión, estas expresiones incorrectas en CNV se denominan *evaluaciones ocultas*.

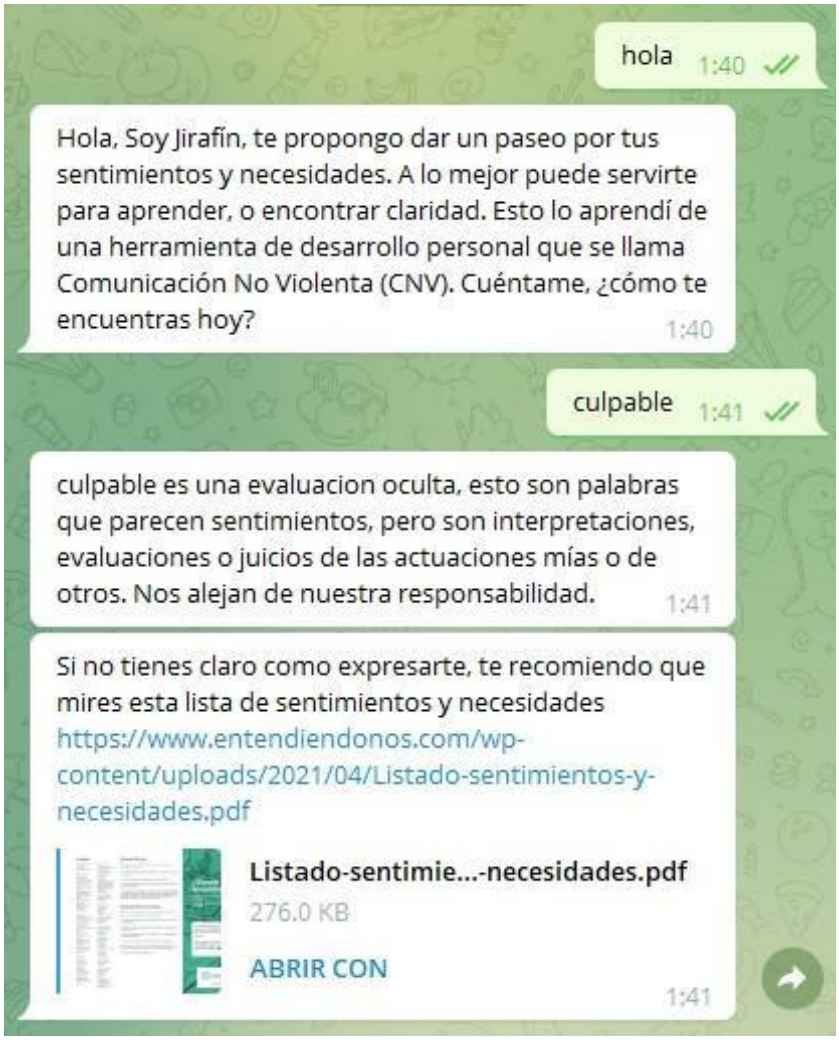


Imagen 11: Diseño evaluación oculta

El siguiente componente, lo forman los ***sentimientos***. En este apartado la intención del bot consiste en empatizar con el usuario preguntándole cómo se encuentra para que nos muestre qué sentimientos experimenta en ese momento.

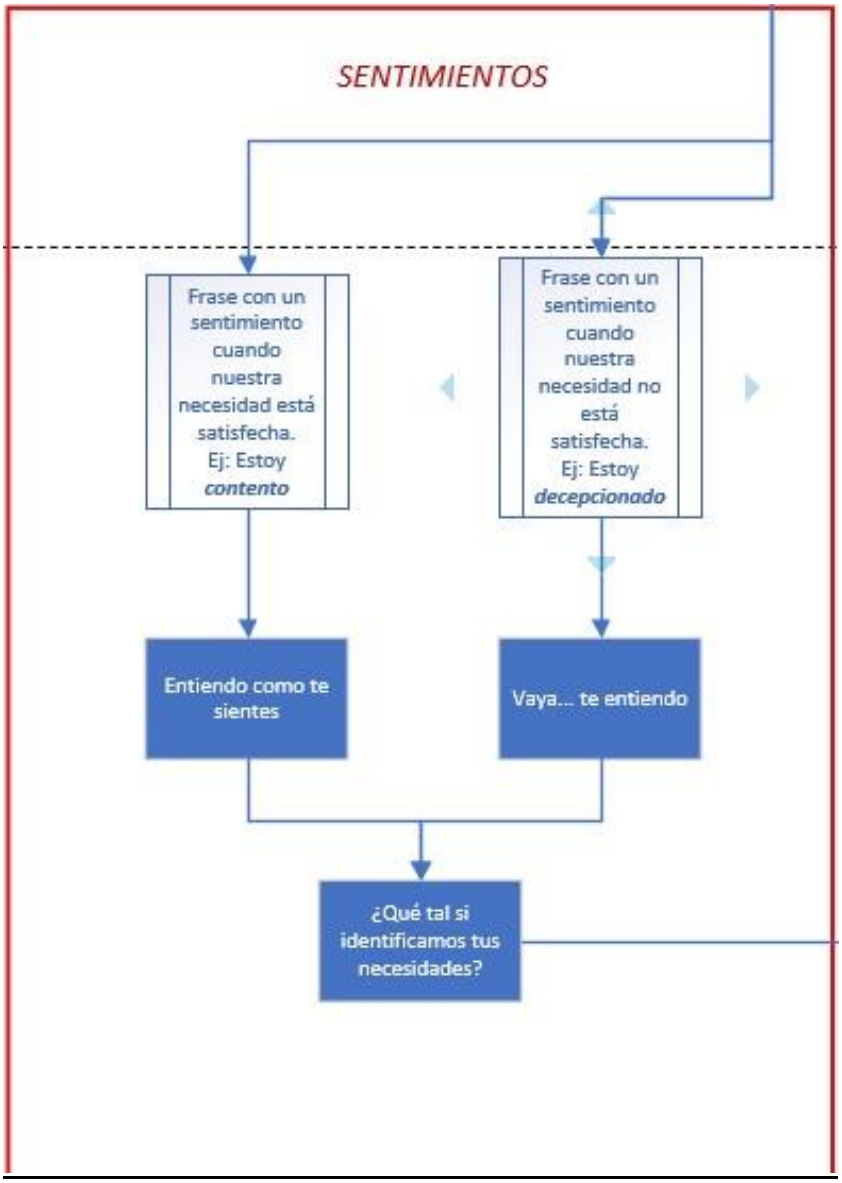


Imagen 12: Diseño sentimientos

Mostramos la ventana resultante de nuestro asistente al detectar un mensaje con un sentimiento. Como marca el diseño, el asistente puede mostrar diferentes mensajes dependiendo de si el sentimiento refleja una necesidad satisfecha o insatisfecha. Este comportamiento está definido por los ficheros de configuración que veremos en el apartado de implementación.

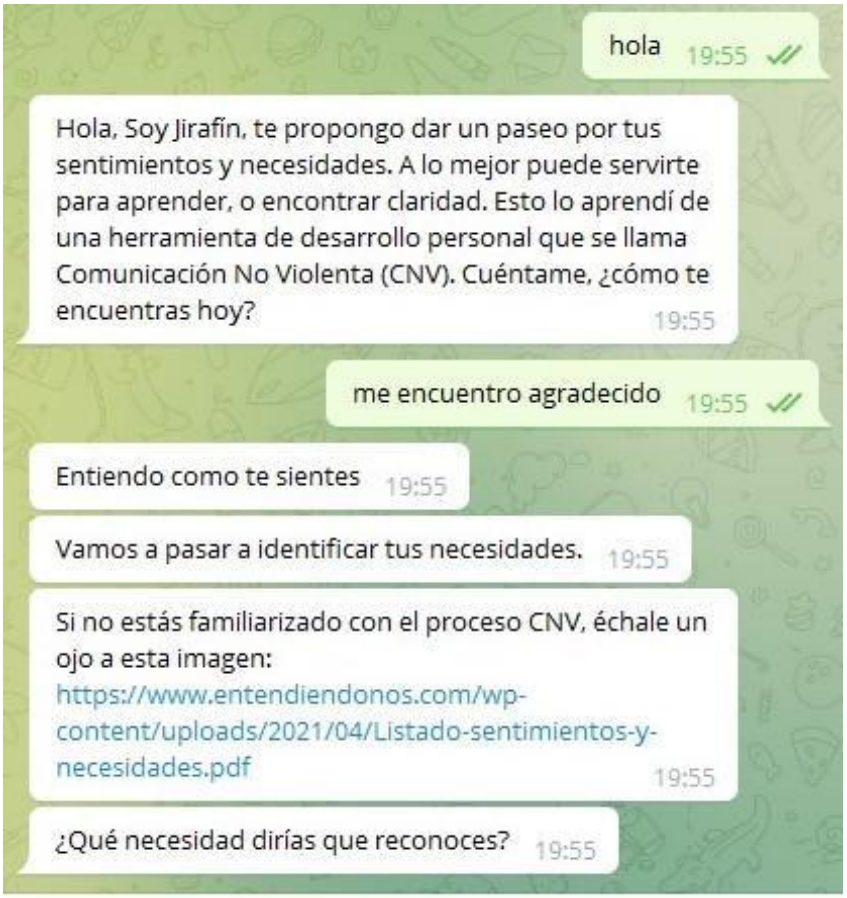


Imagen 13: Diseño sentimientos interfaz

Una vez definido el paso anterior, el bot guiará al usuario al siguiente estado, las ***necesidades,*** cuyo flujo conversacional viene detallado en la siguiente imagen. Es complicado que el usuario identifique su necesidad en un primer instante, por lo que queremos que nuestro bot muestre una serie de listas con las necesidades universales recogidas en las bases de la CNV.

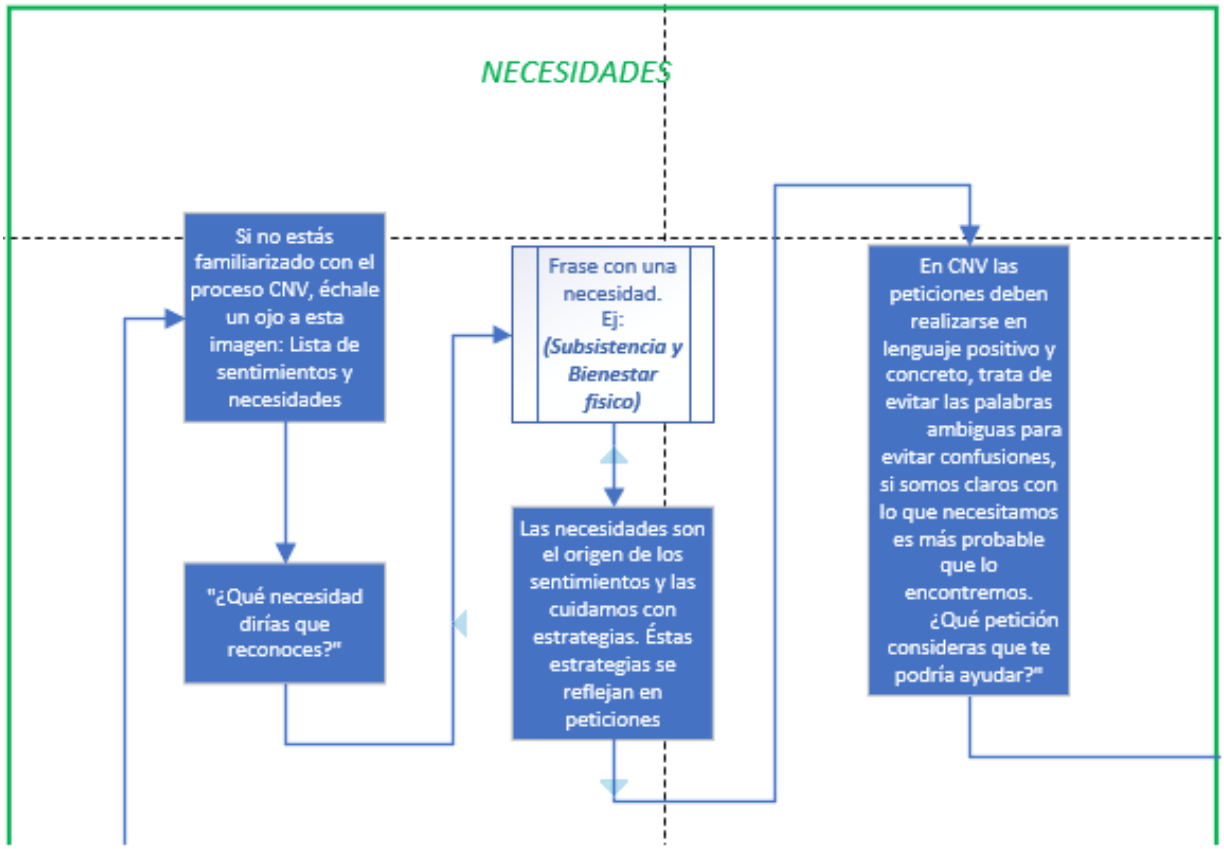


Imagen 14: Diseño necesidades

Podemos observar en la siguiente imagen como quedaría la ventana resultante del apartado de necesidades en el asistente invitando primero al usuario a visitar una lista con las necesidades universales recogidas por la CNV para ayudarle a continuar hasta el siguiente paso.

Imagen 15: Diseño necesidades interfaz

Pues bien, el último componente de la CNV son las ***peticiones***, puede parecer sencillo realizar peticiones cuando tenemos identificada nuestra necesidad, pero no siempre encontramos la manera óptima de realizarlas. En este caso el bot muestra una serie de pautas para realizar una petición correcta en el ámbito CNV.

Imagen 16: Diseño peticiones

Una vez visto el diseño del flujo conversacional del componente que representa las peticiones, mostramos la ventana que muestra el asistente.

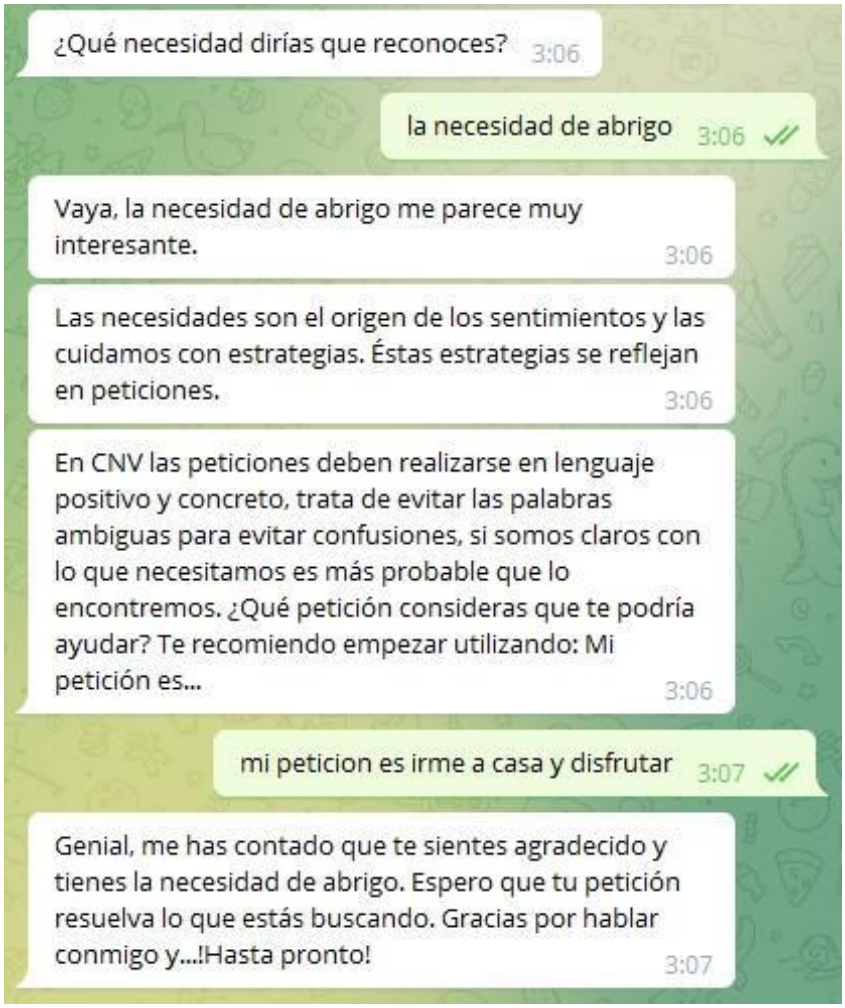


Imagen 17: Diseño Peticiones interfaz

Para finalizar, nuestro asistente recoge los datos obtenidos durante los pasos anteriores de la conversación y los muestra a modo de resumen con un mensaje de despedida.

Capítulo 6: Tecnologías y herramientas de desarrollo

En este apartado vamos a detallar las herramientas que hemos utilizado para llevar a cabo este proyecto, desde la aplicación de desarrollo hasta los editores de texto.

6.1 Elección de la aplicación generadora de bots.

Uno de los principales problemas fue elegir la aplicación de desarrollo ya que, aunque existen varias alternativas, la mayoría constan de pocas funciones en su versión free y requiere de una suscripción para obtener la totalidad de sus funciones.

Decidimos elegir Rasa Open Source (explicada en la siguiente subsección) ya que es una herramienta de código abierto que utiliza aprendizaje automático y totalmente gratuita que cuenta con funciones de reconocimiento de palabras clave y entrenamiento del lenguaje, además de una comunidad bastante grande formada por diferentes foros de ayuda que nos han resultado bastante útiles a la hora de desarrollar este proyecto.

Nos hemos decantado por utilizar Telegram como plataforma integradora de nuestro bot ya que cuenta con un desarrollo de terceros muy amplio gracias a su aplicación Bot Father (explicado en la siguiente subsección). Además, Telegram permite conexión con Rasa, que es el programa de aprendizaje automático que hemos elegido para desarrollar el bot.

Rasa

Rasa es una plataforma de código abierto que utiliza aprendizaje automático para desarrollar conversaciones de texto y voz, proporcionando la infraestructura y las herramientas necesarias en la creación de asistentes virtuales (*Rasa Docs*, n.d.).

Nos proporciona dos herramientas principales para la generación de Bots, la primera es la comprensión del lenguaje natural (Rasa NLU), pues podemos convertir texto sin formato en datos estructurados analizando la intención del usuario y organizando las claves importantes mediante bibliotecas de procesamiento de lenguaje natural y aprendizaje automático.

La segunda hace referencia a la gestión del diálogo (Rasa CORE), es el aprendizaje automático y junto RASA NLU nos dice que respuesta dar según los datos que haya recopilado en el paso anterior, para ello se guía por una estructura interna donde se han definido las historias y las palabras clave.

Su arquitectura es modular en cuanto a diseño, lo que permite una fácil integración con otros sistemas, por ejemplo, Rasa CORE se puede utilizar como administrador de diálogo junto con servicios de comprensión del lenguaje distintos de Rasa NLU. Si implementamos el código en un lenguaje como Python, ambas herramientas pueden ofrecer las API HTTP para que pueda ser utilizado por proyectos que utilicen otros lenguajes de programación.

Consta de varios foros de ayuda y cuenta con un portal para desarrolladores a través del cual podemos empezar a desarrollar un asistente desde cero entre otras muchas opciones.

Rasa nos permite también conectarnos a los principales canales de mensajería como son Slack, Facebook, Telegram, a bases de datos, APIs y otras fuentes de datos, además, gracias al uso de dockers podemos desplegar Rasa en las instalaciones o mediante nuestro proveedor en la nube.

BotFather

Telegram es una aplicación de mensajería que se caracteriza por su rapidez y la seguridad de sus conversaciones. Es totalmente gratuita y está disponible para la mayoría de los dispositivos. Se creó en el año 2013 y alcanzó millones de descargas que a día de hoy siguen creciendo. Una característica especial es que es de código abierto pues invita a todos los desarrolladores a crear sus propias aplicaciones.

Es aquí cuando hablamos de BotFather, es un Bot que se integra con Telegram gracias a que Telegram cuenta con una API para el desarrollo de bots. Para empezar, tenemos que abrir una conversación con él, una vez abierta, nos mostrará toda la lista de comandos sobre la cual podremos crear y posteriormente configurar nuestro propio Bot.

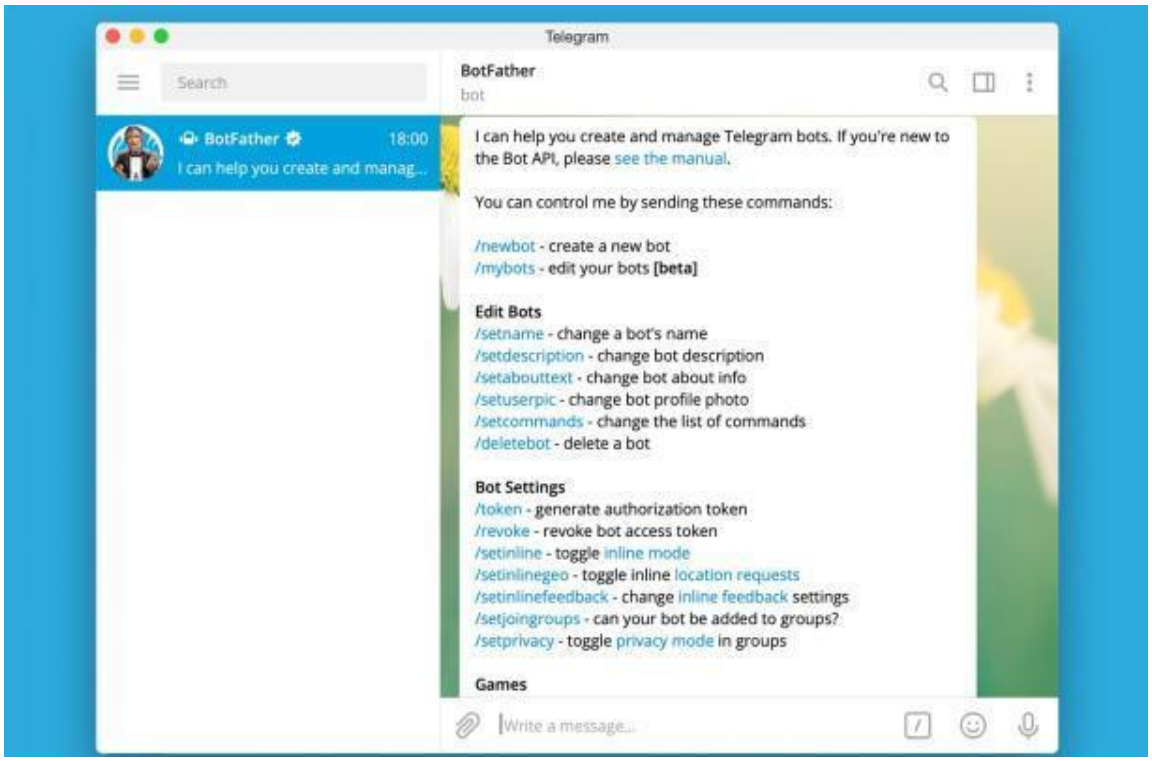


Imagen 18: Interfaz BotFather

6.2 Lenguajes de programación

Python

Python es un lenguaje de programación interpretado con el fin de ser legible por cualquier persona con conocimientos básicos de programación y posee unas características que le confieren ventajas sobre otros lenguajes.

Podemos ejecutarlo en cualquier plataforma ya que es compatible con diferentes sistemas operativos como Windows o Linux.

```
class Rectangle(Blob):

    def __init__(self, width, height,
                 color='black', emphasis=None, highlight=0):
        if width == 0 and height == 0 and \
           color == 'red' and emphasis == 'strong' or \
           highlight > 100:
            raise ValueError("sorry, you lose")
        if width == 0 and height == 0 and (color == 'red' or
                                           emphasis is None):
            raise ValueError("I don't think so")
        Blob.__init__(self, width, height,
                      color, emphasis, highlight)
```

Imagen 19: Código Python

Es un lenguaje multiparadigma ya que soporta parcialmente la orientación a objetos, la programación imperativa y la programación funcional lo que le permite ser muy flexible a la hora de integrarlo en diferentes marcos como las aplicaciones web o la inteligencia artificial.

Es totalmente gratuito pues se trata de un lenguaje de código abierto, no requiere ningún tipo de licencia y está respaldado por una gran comunidad.

6.3 Editores de texto

Sublime Text

Sublime es un editor de texto multiplataforma que posee soporte para casi todos los lenguajes de programación, además cuenta con remarcado sintáctico y otras herramientas para desarrolladores.
También destacar que, gracias a la API en Python de Sublime, se pueden desarrollar plugins para expandir el editor de texto y aumentar sus funcionalidades, existen varios muy útiles como el que permite integrar Git en Sublime y hacer commits desde el.

6.4 Control de versiones

Git

A la hora de trabajar y debido a la distancia de cada uno de los miembros del grupo, empezamos a usar una aplicación que controlara las versiones y nos facilitase la información sobre cada cambio.
Podemos controlar quién ha modificado cada elemento y cuando lo ha hecho, también observar cómo ha cambiado el proyecto con el tiempo, con la ayuda de las etiquetas podemos controlar sus versiones y poder retomar alguna antigua por si no nos satisfacen los resultados obtenidos.

6.5 Diagramas de Flujo

Visio

Visio es una herramienta de Microsoft que permite realizar diagramas gráficos, escalas de tiempo, planos de planta, y otros muchos esquemas para trabajar de una forma más visual, capaz de organizar ideas complejas. Cuenta con un gran abanico de plantillas con las que podemos empezar desde cero y contiene herramientas que permiten establecer las relaciones entre los elementos de los diagramas, además permite conectar datos desde otros orígenes como son Excel, Access o SharePoint.

Permite guardar diagramas en la nube y compartirlos a través de un navegador, sin necesidad de tener instalado Visio y visualizar proyectos desde cualquier dispositivo móvil.

Capítulo 7. Implementación y arquitectura

En este capítulo se detalla la implementación y arquitectura que se ha utilizado para cumplir con los requisitos que se han establecido en las secciones anteriores.

7.1 Arquitectura

Una vez especificado el diseño de nuestro proyecto, se desarrolla un diagrama que servirá de guía para poder continuar con la explicación de una forma más detallada.

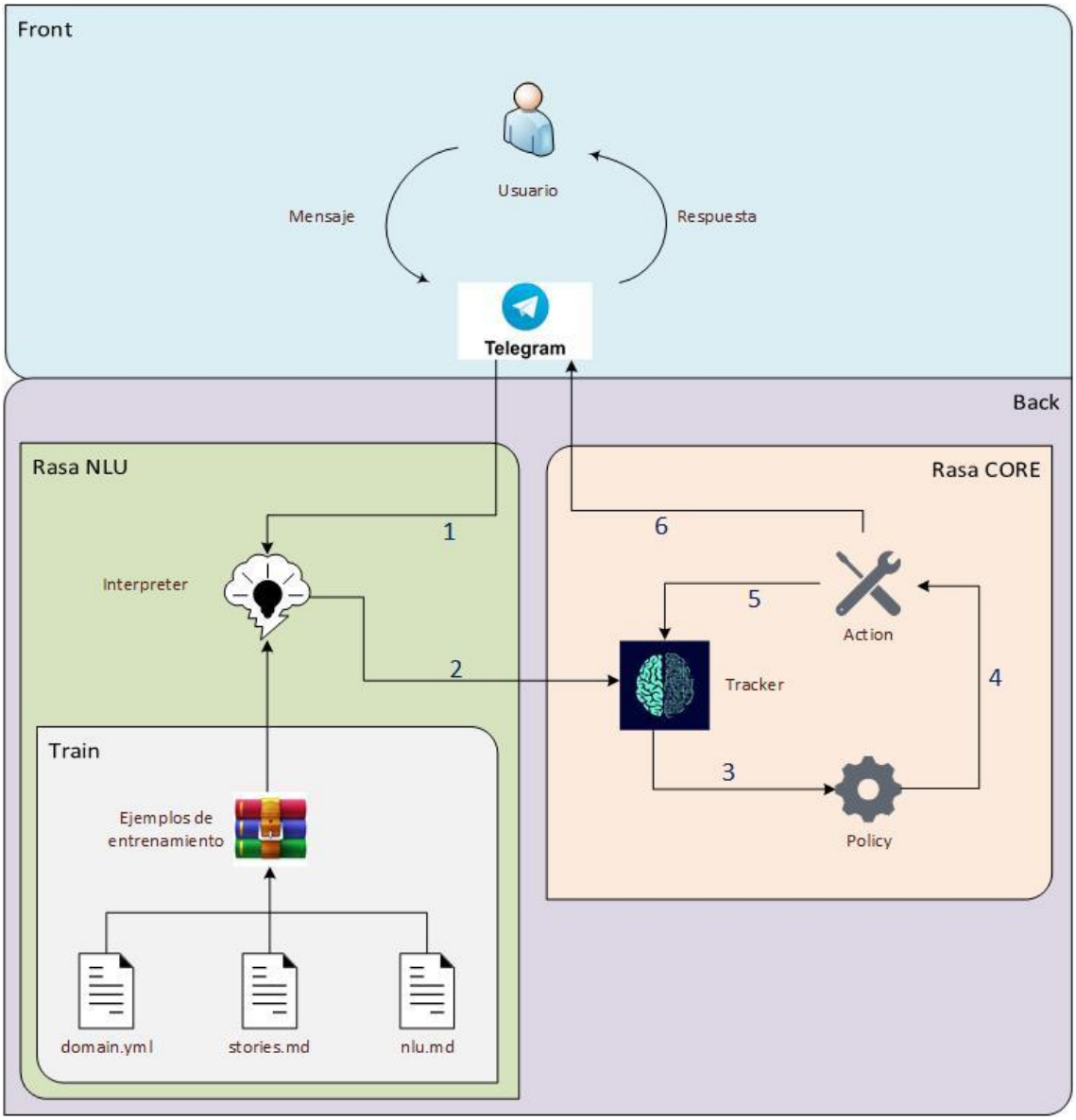


Imagen 20. Arquitectura Bot

A continuación, vamos a mostrar detalladamente el recorrido de un mensaje a través de cada uno de los componentes que forman parte de la arquitectura de nuestro asistente.

El ***número uno*** indica el mensaje que envía el usuario a través de la interfaz de Telegram, el intérprete (Rasa NLU), recibe este mensaje para extraer la intención, una intención o "intent" es un conjunto de palabras expresadas por el usuario y las entidades o "entity" que son piezas estructuradas de información que se pueden extraer del mensaje. Por ejemplo, si el mensaje que introduce el usuario es: me siento "*agradecido"*; Rasa NLU va a detectar que se está utilizando "*agradecido"* una palabra clave catalogada como "*intent"* en la categoría de sentimientos, por lo que clasificará está palabra como sentimiento cuando tenemos la necesidad satisfecha.

El siguiente componente por el que atraviesa el mensaje viene dado por el ***número dos***, el mensaje llega al Tracker, que es la parte lógica encargada de mantener el estado de la conversación, éste recibe una notificación de que ha llegado un nuevo mensaje y la información asociada, es decir, NLU envía la "*intent"* en este caso "*agradecido"* y la "*entity*" clasificada como "*sentimientos*" para que después pueda enviarlos a la *Policy* con el resto de información.

En el ***número tres,*** la política recibe el estado de la conversación que le envía el tracker. La *Policy* es la encargada de decidir qué acción debe tomar el bot con la información obtenida, esta acción se evalúa teniendo en cuenta la configuración del fichero ***config.yml***, más adelante veremos un ejemplo de este fichero, donde definimos qué políticas vamos a utilizar para regir el comportamiento del asistente. Por ejemplo: en este asistente una de las políticas que hemos utilizado es TEDPolicy, que utiliza una arquitectura de transformación para predecir qué diálogo tener en cuenta y cuál ignorar para hacer predicciones, en nuestro caso, esto nos ayuda cuando un usuario decide volver a repetir un estado anterior en el proceso CNV, por ejemplo, si estamos en el apartado de necesidades y la respuesta del usuario es "*me siento agradecido"* o cualquier otro sentimiento recogido en el proceso CNV, el asistente volverá al apartado de sentimientos.

En el ***número cuatro,*** la política elige la acción que nuestro asistente debe utilizar, este conjunto de acciones vienen definidas en los ficheros ***actions.py*** y ***domain.yml***. Éstos ficheros alojan las posibilidades de respuesta que hemos definido, en este caso, al detectar la intención "*agradecido*"y clasificarla como "*sentimiento"*, nuestra política decide enviar el mensaje para continuar hacia el siguiente componente CNV, que son las necesidades. Nuestro Bot mostrará: *Entiendo cómo te sientes, aquí tienes una lista de necesidades. ¿Podrías identificarla?*

El ***número cinco*** define la actualización del estado de la conversación, el *Tracker* guarda la acción elegida por la *Policy*, en este caso el mensaje de respuesta del Bot. Si la predicción obtenida por el *Tracker* no es correcta,es decir, si la respuesta no coincide

con el apartado donde nos encontramos, por ejemplo, *Policy* detecta que la siguiente acción a mostrar es un mensaje relativo al apartado de las peticiones, encontrándonos en sentimientos, el *Tracker*, siguiendo el camino que hemos descrito en la lógica dentro del fichero de configuración, internamente vuelve al paso tres, donde volverá a evaluar la intención y su posible respuesta como hemos definido en los pasos anteriores.

Para finalizar, en el ***número seis*** se ejecuta la acción obtenida, puede ser un mensaje de texto como respuesta, o bien cualquiera de las funciones que hayamos definido en el fichero ***actions.py*** como mostrar una lista, un botón o un cuadro de elección. En este caso mostrará el mensaje de respuesta descrito anteriormente en ***domain.yml*** .

Una vez visto el recorrido de un mensaje, vamos a explicar en más detalle cómo funciona cada uno de los componentes y las conexiones que se establecen entre los módulos que lo forman.

Componente de entrenamiento (Train)

Para que el asistente reconozca las intenciones del usuario independientemente de cómo exprese su mensaje, debemos establecer unos mensajes de ejemplo que sirvan de patrón para que el asistente pueda aprender.

Estos mensajes se denominan "**Intents**" (intenciones). Por ejemplo, una persona puede interpretar el "*intent"* saludo de muchas formas: *hola, buenos días, buenas tardes* o *hey* entre muchas otras.

Por otro lado, hay mensajes que contienen entidades que es necesario analizar para ofrecer una respuesta mucho más personalizada. Por ejemplo, si tenemos un *intent* asociado a un sentimiento como "Me siento activo", donde *activo* forma parte de una entidad "**entity**", este servirá ofrecer una mejor respuesta y estructurar los datos proporcionados por el usuario o poder utilizarlo mediante una API externa.

Nlu.md , **stories.md** y **domain.yml** son los ficheros principales a la hora de generar los ejemplos de entrenamiento, éstos ficheros sirven de guía para que Rasa pueda formar ejemplos de conversaciones que posteriormente puedan ser utilizados por nuestro asistente.

Veamos la organización e implementación de los ficheros que recogen los componentes para generar el bot final.

Nlu (Natural Language Understanding)

Los *intents* y las *entities* se encuentran definidos en un fichero llamado ***nlu.md***. En la figura que se muestra a continuación podemos ver un ejemplo de cómo está organizado. Las intenciones son todos los ejemplos de palabras que el usuario puede introducir, y se agrupan en *intents,* sin embargo como podemos ver en la siguiente imagen, podemos asociar una *entity* (marcada en azul) a un conjunto de *intents.*

```
## intent:saludo
- hola
- hey
- buenas
- buenos dias
- buenas tardes
- buenas noches
- que tal
- como estas
- saludos

## intent:despedida
- adios
- nos vemos
- hasta luego
- chao
- hasta mañana

## intent:evaluacionesOcultas
- [Abandonada](evaluacionOculta)
- [Acorralada](evaluacionOculta)
- [Acosada](evaluacionOculta)
- [Acusada](evaluacionOculta)
- [Aislada](evaluacionOculta)
- [Alejada](evaluacionOculta)
- [Amenazada](evaluacionOculta)
- [Antipática](evaluacionOculta)
- [Apartada](evaluacionOculta)
- [Aplastada](evaluacionOculta)
- [Atacada](evaluacionOculta)
- [Atrapado](evaluacionOculta)
- [Avergonzado](evaluacionOculta)
- [Burlado](evaluacionOculta)
- [Censurado](evaluacionOculta)
```

Imagen 21. nlu.md

Stories

Dentro de este fichero se encuentran todos los ejemplos de conversaciones que hacen capaz al asistente de devolver las respuestas adecuadas en función de lo que ha propuesto anteriormente el usuario. Cada historia viene identificada por un *## título* y está formada por uno o varios ** intent* y acciones o respuestas previamente definidas en sus ficheros correspondientes.

En Rasa todas las conversaciones están definidas dentro de un archivo llamado *stories.md*

```
## camino saludo
* saludo
  - utter_saludo

## camino satisfecha
* SentimientosNecesidadSatisfecha
  - utter_your_Satisfecha
  - utter_your_paso_necesidades
  - utter_conocer_CNV
  - utter_necesidad2

## camino necesidades
* Necesidades
  - utter_necesidad
  - utter_peticiones
  - utter_peticiones2

## camino peticion
* peticion
  - utter_fin

## camino nosatisfecha
* SentimientosNecesidadNoSatisfecha
  - utter_your_Satisfecha
  - utter_your_paso_necesidades
  - utter_conocer_CNV
  - utter_necesidad2

## camino evaluacionOculta
* evaluacionesOcultas
  - utter_oculta
  - utter_oculta2
```

Imagen 22. stories.md

Domain

El dominio define el contexto sobre el que trabaja el bot. En el fichero *domain.yml* se especifican todas las partes del proceso NLU como son las *entities*, *intents*, *actions*, *slots, forms y requests.* En la siguiente imagen podemos ver un ejemplo de cómo está organizado dicho fichero.

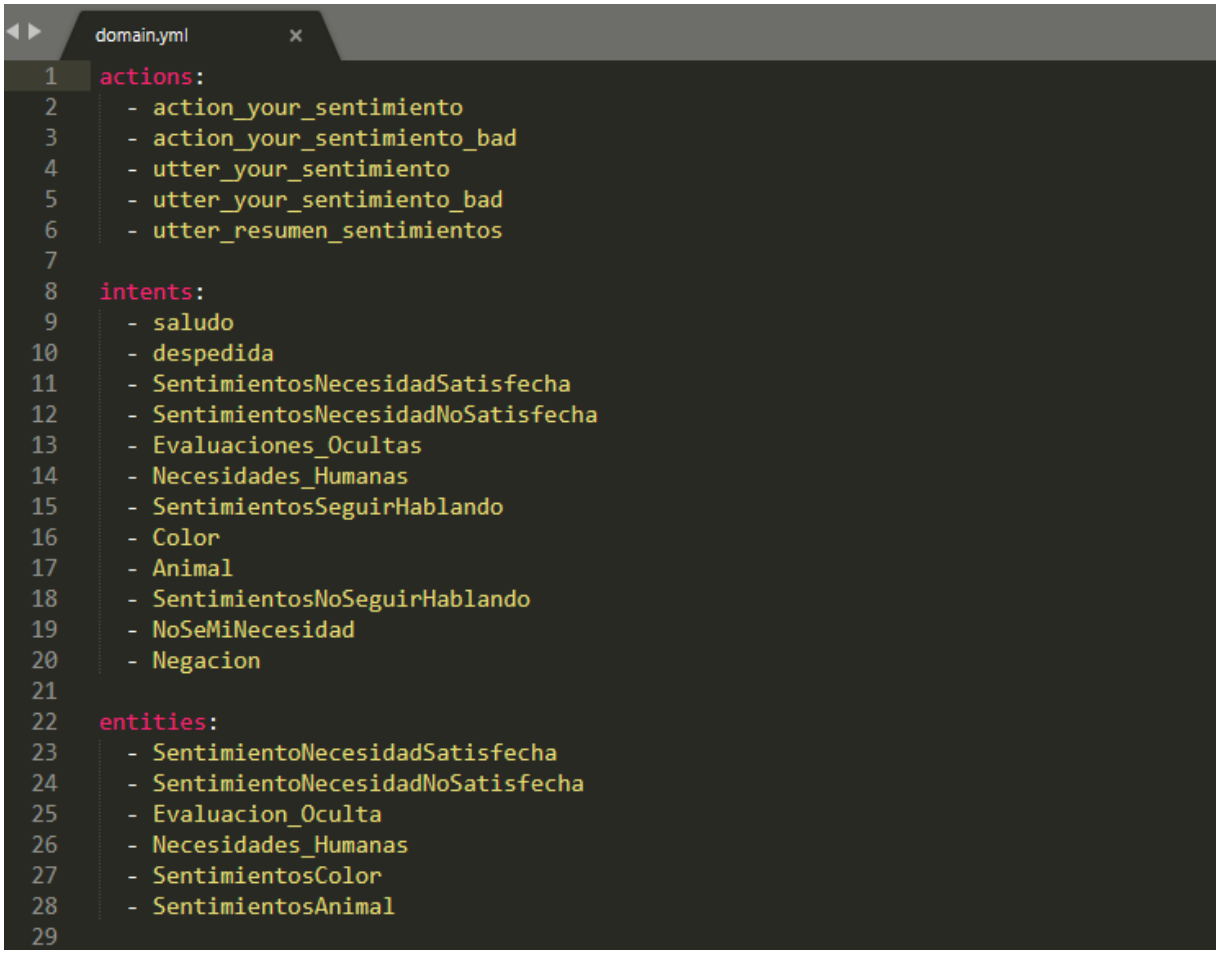


Imagen 23. domain.yml

Todas las respuestas que el asistente puede ofrecer ante las intenciones del usuario se encuentran definidas en el apartado *request*. Este apartado puede ser opcional ya que muchas de las respuestas las podemos definir mediante la implementación de acciones personalizadas.

Comprensión y procesamiento de texto (Rasa NLU)

Una vez hemos creado un ejemplo de entrenamiento con el componente *Train*, Rasa tiene la capacidad de utilizarlos para mantener actualizado el diagrama de flujo. Todos estos ejemplos de entrenamiento están almacenados en la carpeta *models*.

Dentro de este componente destacamos los dos módulos más importantes:

Podemos observar las diferentes etapas desde que el usuario envía un mensaje, que atraviesa el conector encargado de reconocer el tipo de mensaje, en este caso estamos trabajando con mensajes de texto pero podrían ser audios si hemos configurado la

aplicación para ello, y trasladarlo al *Broker*, encargado de gestionar las comunicaciones enviando la información relevante del mensaje entre Rasa NLU y Rasa CORE, una vez el mensaje llega a Rasa NLU lo traslada al *Interpreter*, que con su lógica interna obtiene los *intent* y *entities* reconocidos para devolverlos al *Broker*, que finalmente lo dirige a Rasa CORE de una forma estructurada.

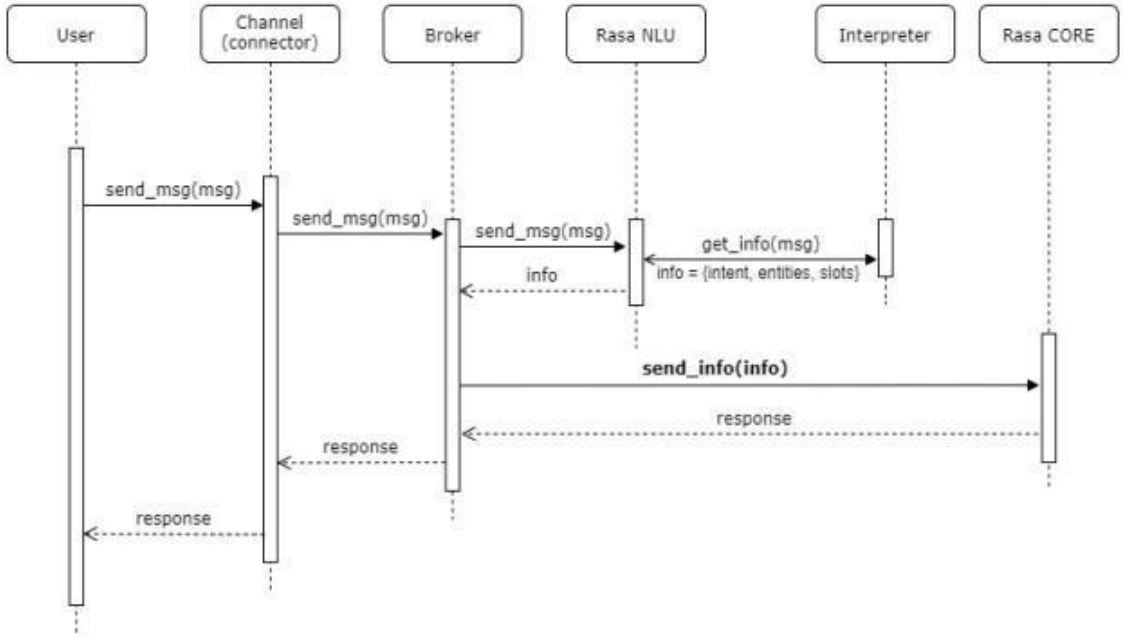


Imagen 24. Estructura NLU

Núcleo de flujo conversacional (Rasa CORE)

Una vez adquiridos los ejemplos de entrenamiento, las intenciones y entidades obtenidas por NLU junto con las acciones personalizadas, envía la información al *Tracker,* que se encarga de mantener la información del estado de la conversación y enviarla al *Policy* definido anteriormente. Según las políticas que hayamos decidido incluir en nuestro fichero de configuración, se decide qué acción debe tomar el Bot a partir de los componentes obtenidos.

Acto seguido, *Tracker* guarda la decisión tomada y ejecuta la acción correspondiente que formará parte de la respuesta final que CORE proponga al usuario.

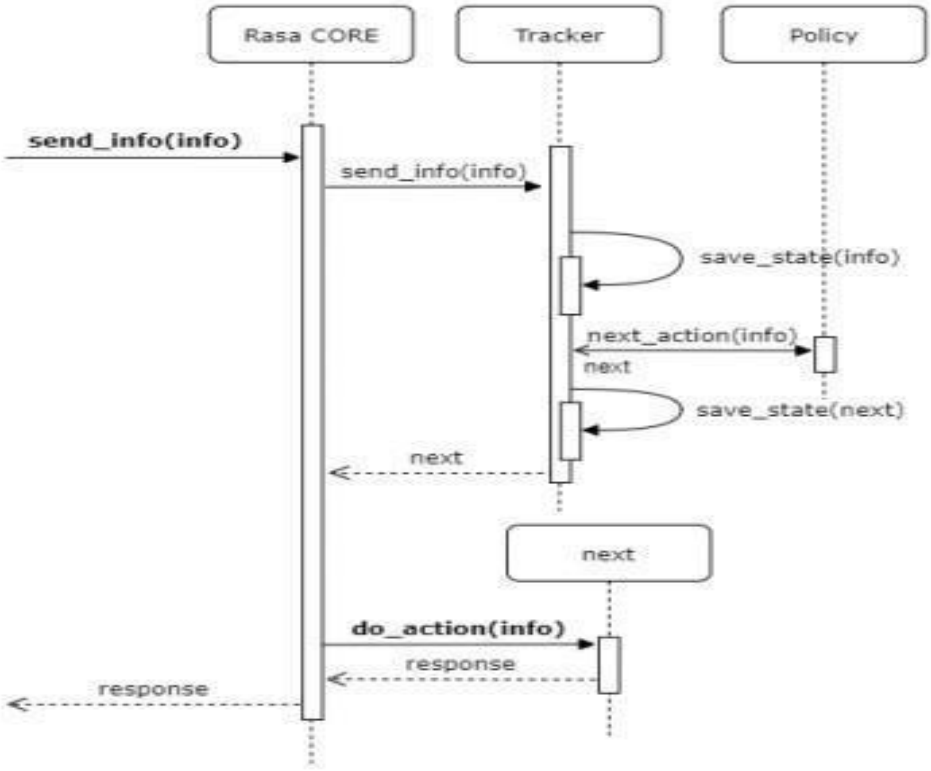


Imagen 25. Estructura CORE

7.2 Conexión con el servidor local

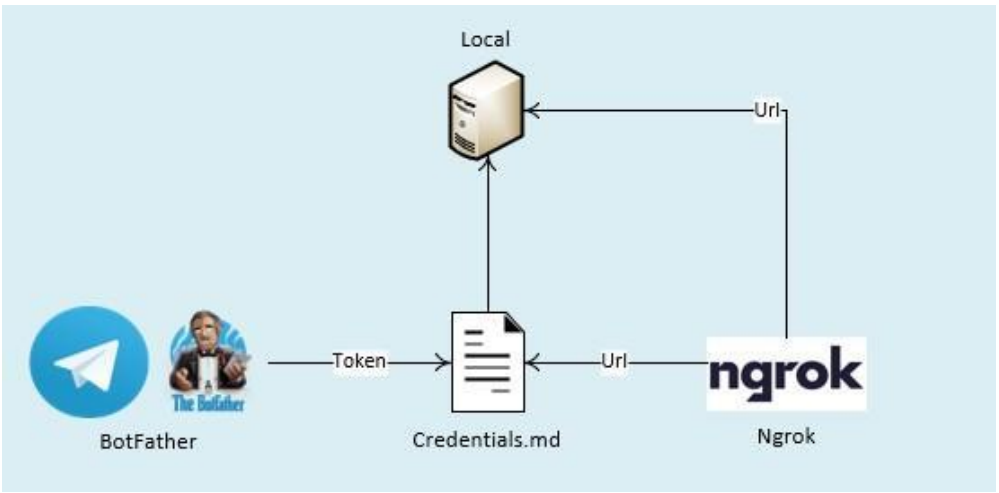


Imagen 26. Conexión con Telegram

Para realizar la conexión con Telegram debemos seguir 2 sencillos pasos:

- Dentro del archivo *Credentials.yml*, debemos introducir la URL donde Telegram deberá enviar los mensajes, reemplazando el host y el puerto con los valores apropiados del servidor donde estemos utilizando Rasa.
- Cuando creamos un bot con BotFather, se genera un código de identificación único llamado *Token,* que debemos introducir junto con nuestro nombre de usuario.

Aquí podemos ver un ejemplo de los valores de cada elemento anterior:

```
 8
 9  telegram:
10    access_token: "951112957:AAEHTO_bGdOi3GZer6h??????????"
11    verify: "CNVBot"
12    webhook_url: "https://b4988b40ba39.ngrok.io/webhooks/telegram/webhook"
13
```

Imagen 27. credentials.yml

Al ejecutar Rasa en local, la mayoría de los canales externos no van a poder encontrar la URL del servidor, por lo que es necesario utilizar una herramienta que nos permita exponer a internet nuestra URL generada dinámicamente como lo hace ***ngrok***. Basta con instalar *ngrok* en el servidor local y podremos empezar con la inspección de tráfico.

Como podemos observar en la siguiente imagen, hemos realizado correctamente la conexión y podemos ver las estadísticas de las conexiones a nuestro bot.

```
ngrok by @inconshreveable                                                    (Ctrl+C to quit)

Session Status                online
Account                       emiliochico11@gmail.com (Plan: Free)
Update                        update available (version 2.3.40, Ctrl-U to update)
Version                       2.3.35
Region                        United States (us)
Web Interface                 http://127.0.0.1:4040
Forwarding                    http://b4988b40ba39.ngrok.io -> http://localhost:80
Forwarding                    https://b4988b40ba39.ngrok.io -> http://localhost:80

Connections                   ttl     opn     rt1     rt5     p50     p90
                              0       0       0.00    0.00    0.00    0.00
```

Imagen 28. Estado conexiones

Una vez conectadas todas las partes podemos interactuar con nuestro bot mediante la terminal de nuestra máquina en local o bien desde la interfaz de Telegram.

Capítulo 8: Verificación y evaluación

En el siguiente capítulo se muestran las pruebas de verificación y evaluación que se han realizado para el proyecto, primero se procederá a hablar sobre las pruebas de verificación para saber si se ejecuta de forma correcta y después las pruebas de evaluación para conocer el comportamiento del asistente.

8.1 Pruebas de verificación

En las pruebas de verificación se ha diseñado una tabla que recoja las justificaciones que sirven para comprobar que el sistema realiza las acciones que definen los requisitos.

Tabla 2. Pruebas de Verificación.

Identificador	Identificador del requisito
Título	Nombre
Descripción	Breve explicación de la prueba a realizar.
Requisito	Requisitos involucrados en la prueba.
Desarrollo	Desarrollo utilizado para comprobar su funcionamiento.
Previsión	Indica cuál debería ser el resultado final.
Evaluación	Verificación de la prueba mediante Correcto o Error

Tabla 2.1. PV-01.

Identificador	PV-01

Título	Ejecución del asistente
Descripción	El asistente se inicia correctamente
Requisito	RF-01
Desarrollo	Iniciar el asistente.
Previsión	El asistente no debe mostrar ningún error al inicializar.
Evaluación	Correcto

Tabla 2.2. PV-02.

Identificador	PV-02
Título	Introducir Observación
Descripción	El asistente debe ser capaz de reconocer una observación según las bases de la CNV.
Requisito	RF-02
Desarrollo	El usuario realiza una observación
Previsión	El asistente debe mostrar el siguiente paso CNV. (Sentimientos)
Evaluación	Error.

Tabla 2.3. PV-03.

Identificador	PV-03

Título	Introducir Sentimiento
Descripción	El asistente debe ser capaz de reconocer un sentimiento según las bases de la CNV.
Requisito	RF-03
Desarrollo	El usuario introduce una observación.
Previsión	El asistente debe mostrar el siguiente paso CNV. (Necesidades)
Evaluación	Correcto.

Tabla 2.4. PV-04.

Identificador	PV-04
Título	Introducir Necesidad
Descripción	El asistente debe ser capaz de reconocer una necesidad según las bases de la CNV.
Requisito	RF-04
Desarrollo	El usuario introduce una necesidad
Previsión	El asistente debe mostrar el siguiente paso CNV. (Peticiones)
Evaluación	Correcto.

Tabla 2.5. PV-05.

Identificador	PV-05
Título	Introducir Petición
Descripción	El asistente debe ser capaz de reconocer una petición según las bases de la CNV.
Requisito	RF-05
Desarrollo	El usuario realiza una petición.
Previsión	El asistente debe mostrar un resumen con los pasos elegidos anteriormente y un mensaje de despedida.
Evaluación	Correcto.

Tabla 2.6. PV-06.

Identificador	PV-06
Título	Introducir Evaluación Oculta
Descripción	El asistente debe ser capaz de reconocer una evaluación oculta recogida por la CNV.
Requisito	RF-06
Desarrollo	El usuario introduce una evaluación oculta.
Previsión	El asistente debe pedirle que introduzca un sentimiento válido.
Evaluación	Correcto.

Tabla 2.7. PV-07.

Identificador	PV-07
Título	Mensaje de resumen y despedida
Descripción	El asistente debe ser capaz de mostrar el mensaje relacionado con el resumen y despedirse del usuario ya que ha finalizado el proceso CNV.
Requisito	RF0-2, RF-03, RF-04, RF-05, RF-07
Desarrollo	El usuario termina el proceso CNV.
Previsión	El asistente debe mostrar el resumen del proceso y la despedida.
Evaluación	Correcto.

Tabla 2.8. PV-08.

Identificador	PV-08
Título	Implementación del recorrido
Descripción	El asistente debe ser capaz de pasar por casa paso del camino mínimo marcado..
Requisito	RF0-2, RF-03, RF-04, RF-05, RF-07,RF-09 RF-10, RNF-12
Desarrollo	El usuario completa el recorrido desde el saludo hasta la despedida.
Previsión	El asistente debe mostrar correctamente todas las respuestas establecidas en casa paso.
Evaluación	Correcto.

Tabla 2.9. PV-09.

Identificador	PV-09
Título	Gestión de memoria
Descripción	El asistente debe ser capaz de estructurar las intenciones y entidades de forma correcta.
Requisito	RF-09
Desarrollo	El usuario termina el proceso CNV.
Previsión	El asistente debe mostrar en modo desarrollador las entidades recogidas.
Evaluación	Correcto.

Tabla 2.10. PV-10.

Identificador	PV-10
Título	Implementación de funciones
Descripción	Las funciones deben estar correctamente estructuradas.
Requisito	RF-10
Desarrollo	El usuario invoca una acción mediante su paso por el recorrido CNV.
Previsión	El asistente debe ejecutar la acción invocada y mostrar la respuesta correcta según el tipo de función.
Evaluación	Correcto.

Tabla 2.11. PV-11.

Identificador	PV-11
Título	Validación de conexión
Descripción	El asistente debe estar conectado con Telegram.
Requisito	RF-11
Desarrollo	En el fichero "*credentials.yml*" se establecen los patrones de conexión necesarios. Se inicia el asistente.
Previsión	El asistente debe iniciarse correctamente en Telegram.
Evaluación	Correcto.

Tabla 2.12. PV-12.

Identificador	PV-12
Título	Implementación con Rasa
Descripción	El asistente está completamente desarrollado mediante la herramienta Rasa Open Source.
Requisito	RF-10, RNF-12, RNF-13, RNF-14
Desarrollo	Comprobar que todos los ficheros cumplen con el diseño y la estructura marcados por la documentación de Rasa.
Previsión	El asistente solo contiene métodos descritos en la memoria.

Evaluación	Correcto.

8.2 Evaluación

En este apartado vamos a evaluar el comportamiento del asistente ante diferentes situaciones, como por ejemplo la introducción de errores y el uso de lenguaje inclusivo. Todas estas pruebas han sido realizadas por diferentes usuarios para poder albergar la mayor información posible sobre las respuestas generadas por el bot para encontrar su posterior solución. Vamos a estructurar el desarrollo de estas pruebas en varias tablas definidas a continuación mostrando en un apartado diferente los resultados obtenidos para poder explicarlos mejor.

Tabla 3. Pruebas Evaluación

Número de usuarios	Número de usuarios que han participado en las pruebas.
Tiempo (en horas)	Tiempo requerido en la realización
Descripción	Métodos utilizados durante la prueba.
Resultados	Resultados obtenidos durante la prueba.
Observaciones	Información obtenida por parte de los usuarios participantes.

Para la primera prueba de valoración se ha reunido a 10 jóvenes, en concreto 4 mujeres y 6 hombres, de entre 25 y 30 años que no estaban familiarizados con el proceso CNV, la mayoría de los participantes (7 de ellos) disponía de cuenta en Telegram ya que se trataba de estudiantes relacionados con la informática, los que no, tuvieron que bajarse la aplicación y crear una cuenta.

Tabla 3.1. Evaluación 1

Número de usuarios	10
Tiempo (en horas)	5h

Descripción	La primera prueba se realizó con usuarios que no estaban familiarizados con el proceso CNV, por lo que primero era necesario explicarles este concepto. Segundo, una vez explicado el proceso CNV se les dió vía libre para interactuar con el asistente. Para finalizar, fueron entrevistados para obtener el feedback sobre el asistente.
Resultados	Resultados de la Evaluación 1.
Observaciones	En esta primera evaluación, con el uso libre de los usuarios sobre el asistente, obtenemos un feedback poco satisfactorio ya que la mayoría de los usuarios experimentaron sensaciones de confusión ante los resultados generados por el asistente.

Resultados obtenidos en la Evaluación 1.

Uno de los principales problemas que surgen a la hora de evaluar el comportamiento del bot viene dado cuando los usuarios, al ofrecerles un uso libre de la aplicación no encuentran las palabras exactas para comunicarse, por ejemplo, un usuario no sabe qué necesidad o sentimiento experimenta y aun viendo la lista que ofrece nuestro asistente no es capaz de introducir una frase que se pueda estudiar. Por ejemplo, en este caso el usuario introduce el siguiente mensaje: No lo sé.

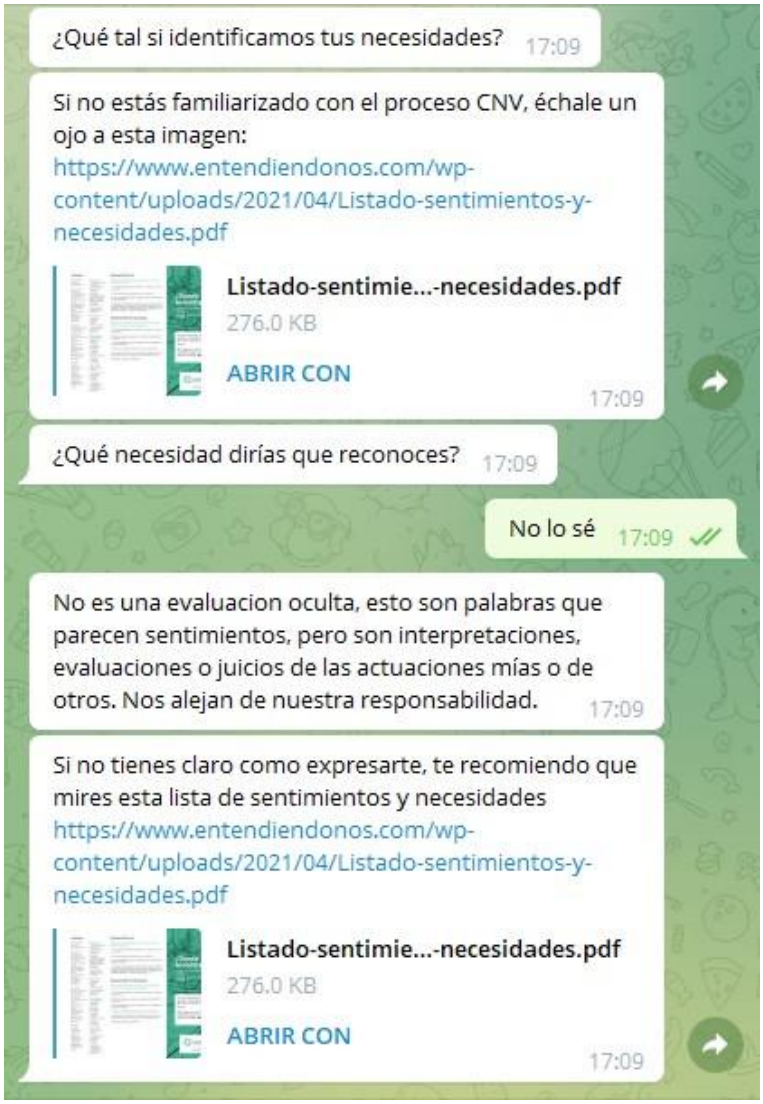


Imagen 29. Evaluación 1

El asistente espera encontrar una necesidad para su posterior extracción, al no encontrarla introduce la primera palabra en la siguiente posición de memoria libre asociada a las valoraciones ocultas, produciendo un comportamiento incoherente.

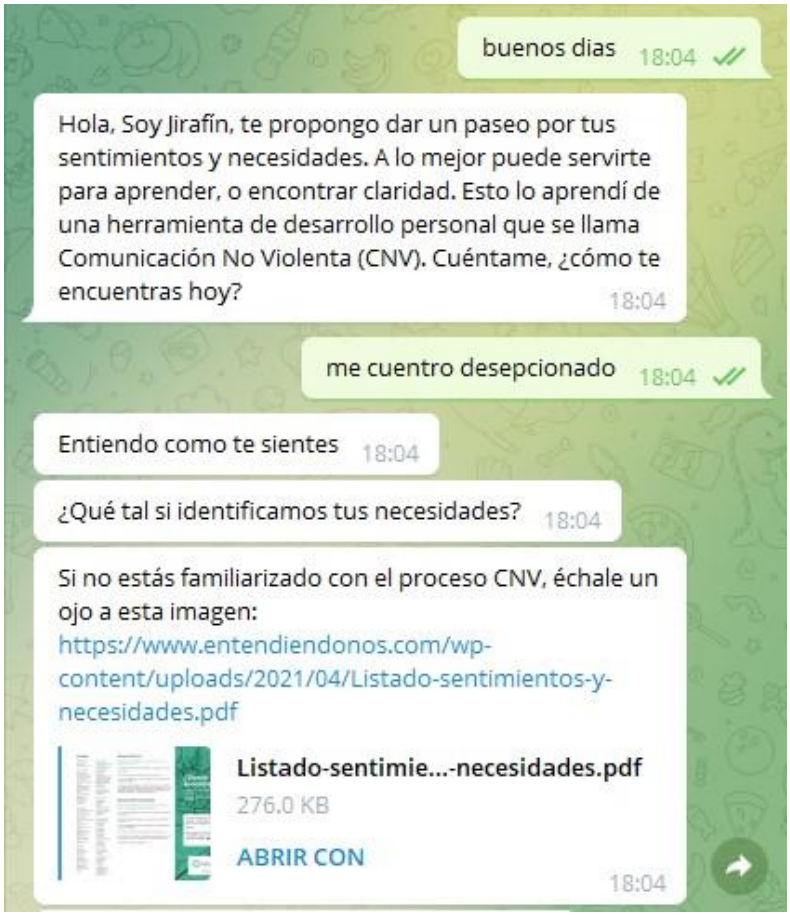


Imagen 30. Evaluación 2

En este ejemplo, cuando el usuario comete faltas de ortografía el asistente es incapaz de extraer correctamente la información que necesita, por lo que la entidad estará vacía y quedará en estado *{None},* solamente visible para el desarrollador. Con esto el asistente muestra un comportamiento incorrecto tomando como referencia los valores introducidos anteriormente.

Cuando el usuario introduce una frase y en ésta no se encuentra la intención requerida en el paso actual, el asistente carece de la entidad necesaria por lo que recoge la palabra que más probabilidad tenga de coincidir con los siguientes pasos de la CNV.

Tabla 3.2. Evaluación 2

Número de usuarios	10
Tiempo (en horas)	5h
Descripción	La segunda prueba consistió en pedirles a los mismos usuarios, que hicieran el recorrido del asistente una vez entendida la metodología CNV. Se les pidió que utilizaran un lenguaje inclusivo y cuidaran la longitud de sus mensajes. Para finalizar, fueron entrevistados para obtener el feedback sobre el asistente.
Resultados	Resultados de la Evaluación 2.
Observaciones	En esta segunda evaluación, una vez los usuarios utilizaron términos recogidos por la CNV pudieron completar el recorrido por el asistente con valoraciones satisfactorias, a la mayoría de los participantes les pareció una herramienta muy interesante, sin embargo, a una minoría no les pareció una buena experiencia.

Resultados obtenidos en la Evaluación 2.

Uno de los ejercicios de evaluación trata sobre el lenguaje inclusivo, donde el usuario pueda expresarse libremente independientemente de su género, por ejemplo, el usuario puede utilizar sentimientos como "*contento*" o "*contenta*" indistintamente. Actualmente se encuentra en pruebas, ya que la política establecida anteriormente está en proceso de actualización al migrar a la nueva versión de Rasa Open Source. En este caso podemos ver en la siguiente imagen que el mensaje introducido es: *Me encuentro muy contenta*. Continuando hasta el final del recorrido podemos ver que el resumen se muestra con resultado óptimo.

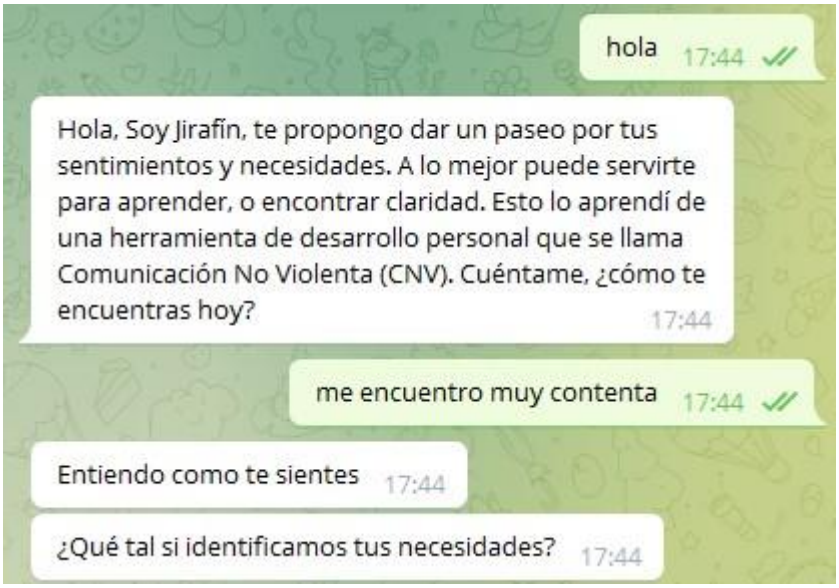


Imagen 31. Evaluación 3

Una vez finalizado el proceso, el asistente mostrará el resultado final.

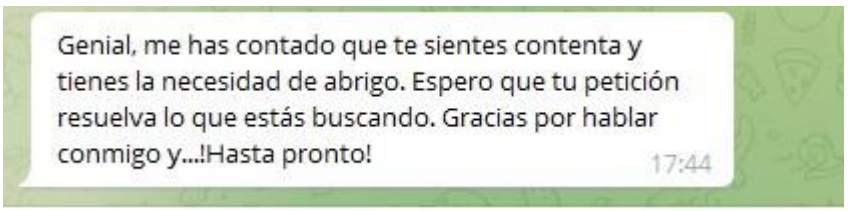


Imagen 32. Evaluación 4

Como podemos observar, aunque las entidades están definidas en un solo género, el asistente es capaz de reconocer la raíz de la entidad debido a las políticas establecidas en la configuración.

Otras Observaciones

Un apunte que hicieron varios de los participantes fue que el asistente no distingue el énfasis de los mensajes enviados, y que esto podría otorgarle una mayor experiencia de usuario, por lo que, investigando, hemos visto que Rasa cuenta con políticas capaces de detectar las interrogaciones y exclamaciones que son las encargadas de añadir énfasis a estos mensajes.

Rasa cuenta con varias herramientas de estudio con las que podemos ver el flujo de conexiones y la transferencia de datos realizada entre el usuario y el asistente, gracias

a esto pudimos observar que cuando varias personas realizaban una conexión simultánea el asistente ofrecía un comportamiento erróneo por lo que queda pendiente de estudio la solución a este problema.

Capítulo 9: Conclusiones y futuras ampliaciones

9.1 Conclusiones

El principal problema consiste en tener una herramienta muy útil como es la comunicación no violenta (CNV) pero sin la visibilidad suficiente como para utilizarla con más frecuencia en la sociedad.

El objetivo es realizar una aplicación capaz de guiar al usuario a través de la CNV, ya que existen muy pocas al ser un concepto poco conocido.

En cuanto a la elección del tipo de aplicación nos decidimos por utilizar un bot en Telegram ya que las aplicaciones de mensajería forman parte de nuestro día a día y así poder hacerlo mucho más accesible.

A la hora de desarrollar el proyecto se ha utilizado el marco de trabajo Scrum ya que una metodología ágil nos ofrece respuestas rápidas durante las fases del proyecto que requirieron más tiempo como el desarrollo del Bot, el proceso de reconocimiento del lenguaje, la conexión con Telegram y las pruebas de verificación y evaluación.

El trabajo muestra la viabilidad técnica del uso de agentes conversacionales para el aprendizaje de la CNV, así como las limitaciones encontradas que están detalladas en los episodios anteriores. Sin embargo, la gran cantidad de ejemplos de entrenamiento necesarios nos plantea utilizar otro tipo de librerías que optimicen el reconocimiento.

Gracias a las pruebas de verificación, hemos percibido que el requisito RF-01 (Detección de observación) no ha arrojado los resultados esperados por lo que se ha decidido cambiar el diseño del asistente para preguntarle al usuario como se encuentra, evitando que realice una observación capaz de generar un comportamiento inesperado del asistente y se ha fijado como el paso principal en el trabajo futuro.

Una vez realizado el cambio de diseño e implementación, los requisitos funcionales principales (RF-03, RF-04, RF-05, RF-06, RF-07, RF-09, RF-11) quedan satisfactoriamente cumplidos, otorgando al asistente la capacidad de reconocer los siguientes pasos de la CNV y mostrar un resumen con los datos obtenidos durante el recorrido. Aun así somos conscientes de que este Bot tiene límites, pues se centra en una conversación estructurada para guiar al usuario a través del proceso CNV sin darle opción a un intercambio de mensajes mayor.

De este proyecto hemos aprendido a trabajar en equipo siguiendo metodologías ágiles, la historia de los asistentes conversaciones y sus aplicaciones en la actualidad, a desarrollarlos utilizando las herramientas que aporta Rasa, a gestionar la conexión

entre el asistente creado y una aplicación de mensajería y sobre todo, a conocer la CNV, que es la herramienta de comunicación que centra nuestro proyecto.

Con la evaluación hemos aprendido que nos falta mucho trabajo para conseguir un asistente completamente sólido, pues desde otros puntos de vista se han detectado muchos más fallos de los que creíamos, pero las valoraciones de los participantes nos dejan buenas sensaciones con el trabajo realizado ya que la mayoría ha sido feedback positivo.

Llegamos a la conclusión de que estamos ante una herramienta muy potente y que si se generan más Bots de este tipo, relacionados con la CNV, orientados al desarrollo personal o con aplicaciones que sirvan para ayudar a la gente, podrían ser muy relevantes y beneficiosos en nuestra vida ya que se trata de herramientas de fácil acceso y entendimiento.

Tras el esfuerzo realizado, las pruebas de evaluación sobre el objetivo principal del proyecto que es guiar al usuario mediante el proceso CNV queda cumplido, aunque con mucha capacidad de mejora, con un bot capaz de guiar al usuario con un aspecto limpio y listo para ser utilizado.

Project's conclusions

The main problem consists of having a very useful tool such as non-violent communication (NVC), but without sufficient visibility to use it more frequently in society.

The objective is to make an application capable of guiding the user through the CNV, since there are very few as it is a little unknown concept.

We decided to use a bot in Telegram since messaging applications are part of our day to day and thus be able to make it much more accessible.

When developing the project, the Scrum framework has been used since an agile methodology offers us quick answers during the phases of the project that required more time, such as the development of the Bot, the language recognition process, the connection with Telegram and the verification and evaluation tests.

The work shows the technical feasibility of using conversational agents for learning NVC, as well as the limitations found, which are detailed in the previous episodes. However, the large number of necessary training examples suggests that we use other types of libraries that optimize recognition.

Thanks to the verification tests, we have perceived that the RF-01 (Observation Detection) requirement has not yielded the expected results, so it has been decided to

change the wizard's design to ask the user how they are, preventing them from making an observation capable of generating unexpected wizard behavior.

Once the design and implementation changes have been made, the main functional requirements (RF-03, RF-04, RF-05, RF-06, RF-07, RF-09, RF-11) are satisfactorily met, granting the assistant the ability to recognize the next steps of the CNV and show a summary with the data obtained during the tour. Even so, we are aware that this Bot has limits, since it focuses on a structured conversation to guide the user through the CNV process without giving the option of a greater exchange of messages.

From this project we have learned to work as a team following agile methodologies, the history of conversations assistants and their applications today, to develop them using the tools provided by Rasa, to manage the connection between the created assistant and a messaging application and on everything, to know the CNV, which is the communication tool that focuses our project.

With the evaluation we have learned that we need a lot of work to get a completely solid assistant, since from other points of view many more failures have been detected than we thought, but the evaluations of the participants leave us good feelings with the work carried out since most have been positive feedback.

We conclude that we are facing a very powerful tool and that if more Bots of this type are generated, related to NVC, oriented to personal development or with applications that serve to help people, they could be very relevant and beneficial in our life as it is about tools of easy access and understanding.

After the effort made, the evaluation tests on the main objective of the project, which is to guide the user through the CNV process, is fulfilled, although with a lot of room for improvement, with a bot capable of guiding the user with a clean appearance and ready to be used.

9.2 Trabajo futuro

Incluir reconocimiento de observaciones.

En cuanto al diseño, es importante encontrar la solución al requisito RF-02, pues la detección de observaciones es uno de los pasos fundamentales en el proceso CNV. Además, gran cantidad de los usuarios con los que hemos realizado las pruebas lo consideran de gran importancia.

Con esto conseguiremos recorrer con exactitud los cuatro componentes de la CNV completando así una funcionalidad y experiencia completa.

Reconocimiento de patrones de lenguaje.

Como acción futura se pretende utilizar una biblioteca distinta capaz de optimizar el reconocimiento de las palabras, nosotros hemos estudiado "*Spacy*" como futura incorporación, permitiendo al usuario utilizar palabras más coloquiales para facilitar el seguimiento que ofrece el asistente.

También estudiar la posibilidad de conectarlo a una base de datos donde podamos recoger grandes cantidades de opciones de búsqueda para ofrecer al usuario respuestas mucho más ajustadas a sus intenciones. Por ejemplo, existen una gran cantidad de evaluaciones ocultas y sentimientos, aunque las más comunes están registradas en la CNV, son una lista en continua expansión. Nuestro asistente cuenta con un fichero donde se encuentran recogidas todas estas evaluaciones y sentimientos, pero cuanto más cantidad de ejemplos en estos ficheros, más cantidad de tiempo necesita para cargarlo, sin embargo, con una conexión a una base de datos reduciriamos el nivel de carga del fichero optimizando la búsqueda.

Diseño

Tras las pruebas con usuarios se cambió el diseño original, ya que suponía la pérdida del interés del usuario y un funcionamiento incorrecto del asistente por lo que optamos por hacer un diseño más sencillo simplificando la estructura y modificando los mensajes que puede devolver el asistente para conseguir una mayor empatía con el usuario. Actualmente seguimos ofreciendo cambios en el diseño como por ejemplo incluir enlaces a material extra para aprender, como cursos o podcasts e imágenes con listados de sentimientos y necesidades que el usuario puede desconocer.

Debido a las observaciones realizadas por los usuarios, nos pareció muy interesante poder mejorar la experiencia de usuario añadiendo las políticas necesarias para el reconocimiento del énfasis de los mensajes, por lo que queda como futura extensión de las funcionalidades.

Detección de emociones

Desde hace años se utiliza la Inteligencia Artificial para detectar emociones de los usuarios aunque no lo escriban explícitamente, por lo que la investigación de bibliotecas y herramientas que puedan hacer esto posible será unos de los puntos a estudiar en futuras actualizaciones.

Future extensions

Include recognition of observations.

Regarding the design, it is important to find the solution to the RF-02 requirement, since the detection of observations is one of the fundamental steps in the CNV process. In addition, many of the users with whom we have carried out the tests consider it of great importance.

With this, we will be able to go through exactly the four components of the CNV, thus completing a complete functionality and experience.

Recognition of language patterns.

As a future action, it is intended to use a different library capable of optimizing word recognition. We have studied "Spacy" as a future incorporation, allowing the user to use more colloquial words to facilitate the follow-up offered by the wizard.

Also study the possibility of connecting it to a database where we can collect large amounts of search options to offer the user answers much more adjusted to their intentions. For example, there are a large number of hidden evaluations and feelings, although the most common are registered in the CNV, they are an ever-expanding list. Our assistant has a file where all these evaluations and feelings are collected, but the lot of examples in these files take more time to load, however, with a connection to a database we would reduce the load level of the file optimizing the search.

Design

After the tests with users, the original design was changed, since it meant the loss of the user's interest and an incorrect behavior of the wizard, so we opted to make a simpler design by simplifying the structure and modifying the messages that the wizard can return to achieve greater empathy with the user. Currently we continue to offer changes in the design such as including links to extra material to learn, such as courses or podcasts and images with lists of feelings and needs that the user may not be aware of.

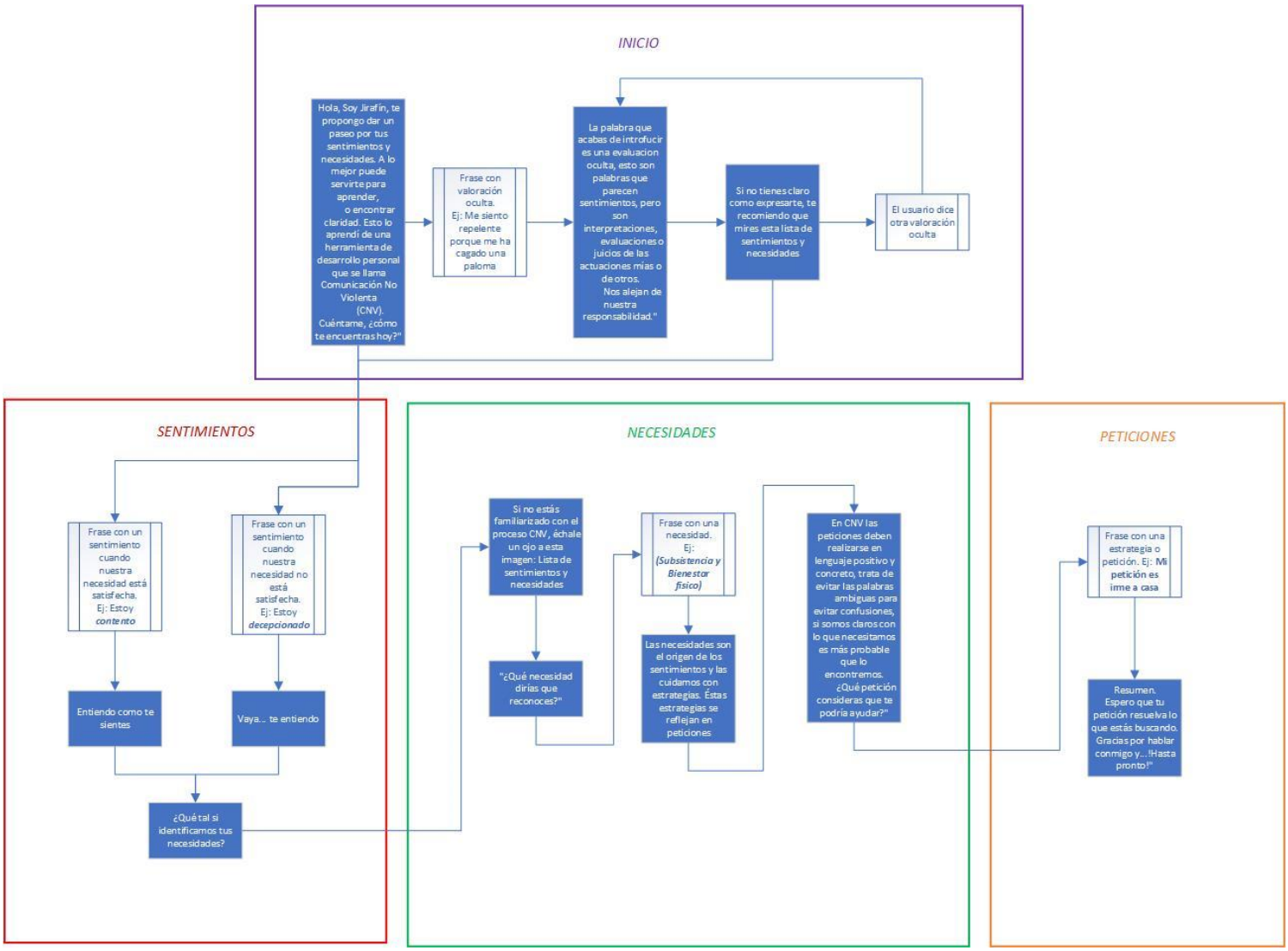
Due to the observations made by the users, we found it very interesting to be able to improve the user experience by adding the necessary policies for the recognition of the emphasis of the messages, so it remains as a future extension of the functionalities.

Emotion detection

Artificial Intelligence has been used for years to detect user emotions even if they do not write it explicitly, so the investigation of libraries and tools that can make this possible will be one of the points to be studied in future updates.

Anexo

Diseño



Bibliography

Referencias citadas

ACNV. (2020, Abril 1). *¿Qué Es La CNV?* Asociación para la Comunicación NoViolenta de España. https://www.asociacioncomunicacionnoviolenta.org/

Adamopoulou, E. (2020). *An Overview of Chatbot Technology*. Eleni Adamopoulou. https://link.springer.com/chapter/10.1007/978-3-030-49186-4_31

Andrade, H. (2005). *Comunicación Organizacional Interna: Proceso, Disciplina y Técnica*. Horacio Andrade. https://books.google.es/books?hl=es&lr=&id=bwelcBnPNuoC&oi=fnd&pg=PA13&dq=Comunicaci%C3%B3n+Organizacional+Interna:+Proceso,+Disciplina+y+T%C3%A9cnica&ots=goIkAtWRm_&sig=0gSYbvfjrT3I5nV2C48RSsoll8Q#v=onepage&q=Comunicaci%C3%B3n%20Organizacional%20Interna%3

Barak, A. (2005). *Psychological applications on the internet: A discipline on the threshold of a new millennium*. Azy Barak. https://www.sciencedirect.com/science/article/pii/S0962184905800381

Brandtzaeg, P. B. (2017, Noviembre 2). *Why People Use Chatbots*. springer. https://link.springer.com/chapter/10.1007/978-3-319-70284-1_30

Graham, O., & Dowe, D. (2020). *The Turing Test*. Stanford Encyclopedia of Philosophy. https://plato.stanford.edu/entries/turing-test/?source=post_page-----------------------

Aiello, L. M., Deplano, M., Schifanella, R., & Ruffo, G. (2012, Mayo 20). *People Are Strange When You're a Stranger: Impact and Influence of Bots on Social Networks*. aaai.org. https://www.aaai.org/ocs/index.php/ICWSM/ICWSM12/paper/view/4523/4961

Heung-Yeung, S., Xiaodong, H., & Di, L. (2018, Febrero). *From Eliza to XiaoIce: Challenges and Opportunities with Social Chatbots*. arxiv Conell University. https://arxiv.org/ftp/arxiv/papers/1801/1801.01957.pdf

Howard, P. (2018, Abril 11). *Algorithms, bots, and political communication in the US 2016 election: The challenge of automated political communication for election law and administration*. tandfonline. https://www.tandfonline.com/doi/full/10.1080/19331681.2018.1448735

Klopfenste, L., Delpriori, S., & Malatini, S. (2017). *The Rise of Bots: A Survey of Conversational Interfaces, Patterns, and Paradigms*. dl.acm. https://dl.acm.org/doi/abs/10.1145/3064663.3064672

Koopman, S., & Seliga, L. (2021). *eaching peace by using nonviolent communication for difficult conversations in the college classroom*. nsuworks.nova.edu. https://nsuworks.nova.edu/pcs/vol27/iss3/2/

Koopman, S., & Seliga, L. (2021). nsuworks.nova.edu. In *Teaching peace by using nonviolent communication for difficult conversations in the college classroom* (2nd ed., Vol. 27). https://nsuworks.nova.edu/pcs/vol27/iss3/2/

Mehrotra, S., Kumar, S., Sudhir, P., Rao, G., Thirthalli, J., & Gandotra, A. (2017). *Unguided Mental Health Self-help Apps: Reflections on Challenges through a Clinician's Lens*. journals.sagepub. https://journals.sagepub.com/doi/pdf/10.4103/IJPSYM.IJPSYM_151_17

Rasa Docs. (n.d.). Rasa Open Source. https://rasa.com/docs/rasa/

Rosenberg, M. (2006). *Comunicación No Violenta: Un Lenguaje De Vida*. Marshall Rosenberg.

Rosenfeld, J. (2001, Agosto 22). *Spiders and Crawlers and Bots, Oh My: The Economic Efficiency and Public Policy of Contracts that Restrict Data Collection*. arxiv, Cornell University. https://arxiv.org/abs/cs/0108015

Suarez, A., Y. Lee, D., & Rowe, C. (2014, Febrero 11). *Freedom Project: Nonviolent Communication and Mindfulness Training in Prison*. journals.sagepub. https://journals.sagepub.com/doi/full/10.1177/2158244013516154#_i7

Velasquez, A. (2020, junio 15). El auge de los chatbots. *periodicoviaje*. https://periodicoviaje.com/consejos/tecnologia/el-auge-de-los-chatbots/

Vinuesa Ramos, C. (2015, Enero 1). *El Auge y La Importancia De Las Redes Sociales En La Educación.* UVaDOC. https://uvadoc.uva.es/handle/10324/13451

Knowlton. A theoretical framework for the classroom: A defense and delineation of student-centered pedagogy. In D. S. . B. S. In R. E, Weiss, editor, Principles of effective teaching in the online classroom, pages 5-14. Jossey-Bass, San Franci sco,CA, 2000.

Otras referencias:

Contreras, Manu. “La Invasión De Los Bots: Qué Son y Para Qué Los Usaremos.” Clipset, 18 Apr. 2016, clipset.com/la-invasion-de-los-bots-que-son-y-para-que-los-usaremos/.

Chris Messina. “2016 Will Be the Year of Conversational Commerce.” Medium, Chris Messina, 3 Oct. 2017, medium.com/chris-messina/2016-will-be-the-year-of-conversational-commerce-1586e85e3991.

Bermúdez, Daniel. “La Importancia De Los Voicebots y Chatbots En La Atención Al Cliente.” Blog De Innovación De Enzyme, blog.enzymeadvisinggroup.com/voicebots-chatbots-atencion-al-cliente.

“Prueba De Turing.” Wikipedia, Wikimedia Foundation, 18 Feb. 2021, es.wikipedia.org/wiki/Prueba_de_Turing.

“La Sorprendente y Poco Conocida Historia De Eliza, El Primer Bot Conversacional De La Historia.” BBC News Mundo, BBC, www.bbc.com/mundo/noticias-44290222.

Simulación Eliza: http://deixilabs.com/eliza.html

Kumar, Ashish, et al. TOOL OF CONVERSATION AND REMINDER-AI CHATBOT, Apr. 2020. https://www.ftms.edu.my/journals/pdf/IJISE/Apr2020/118-129.pdf

Vista De Uso De Bots y Algoritmos Para Automatizar La Redacción De Noticias: Percepción y Actitudes De Los Periodistas En España, revista.profesionaldelainformacion.com/index.php/EPI/article/view/epi.2018.jul.04/40574.

“Crecimiento Personal y Autoayuda.” Psicologia, www.psicologia-online.com/crecimiento-personal-y-autoayuda/.

“Terapia Online En Psicología: TODO Lo Que Necesitas SABER.” CIDEPS, 10 July 2020, cideps.com/terapia-online/.

Flujas-Contreras, Juan Miguel, et al. “Un Programa De Bienestar Emocional Basado En Realidad Virtual y Terapia Online Para Enfermedades Crónicas En Infancia y Adolescencia: La Academia Espacial.” Sept. 2017. https://www.revistapcna.com/sites/default/files/17-16.pdf

Desarrollo de bots:

Periódico, El. “Los 5 Mejores Chat Apps Para Crear Un Bot.” Elperiodico, El Periódico, 31
July 2017, www.elperiodico.com/es/economia/20170730/los-5-mejores-chat-apps-para-crear-un-bot-6198685.

Redacción. “Herramientas Para Crear Chatbots Para Ecommerce.” E, E-Commerce Nation, 17 May 2018, www.ecommerce-nation.es/herramientas-crear-chatbots-ecommerce/.

Psicología y autoayuda. CNV, enfoque laboral:

L. Mauldin, Michael. CHATTERBOTS, TINYMUDS, and the Turing Test Entering the Loebner Prize Competition, L. Mauldin. https://www.semanticscholar.org/paper/CHATTERBOTS%2C-TINYMUDS%2C-and-the-Turing-Test%3A-the-Mauldin/bdd49b4a0b7de03b00412e3b807a855504e1d3af

Chiavenato, I. (2001). Administración: Procesos administrativos (3 Edición). Bogotá: Mc Graw-Hill. https://www.academia.edu/33028175/PROCESO_ADMINISTRATIVO_TERCERA_EDICION

Bimos Zambrano, A. M. (2018). Influencia de la comunicación no violenta para prevenir el mobbing dentro de una empresa privada (Bachelor's thesis, Quito). http://repositorio.usfq.edu.ec/handle/23000/7958

Correal, María Clara, & Bustos, Magaly, & Cuevas, Adriana Constanza, & Panqueva Bernal, María Claudia (2008). El lenguaje y la comunicación en los procesos organizacionales de la empresa. Revista Escuela de Administración de Negocios, (62),141-153.Recuperado de: https://www.redalyc.org/articulo.oa?id=206/20611457010.

Bimos Zambrano, Alejandra María. “Influencia De La Comunicación No Violenta Para Prevenir El Mobbing Dentro De Una Empresa Privada.” Repositorio Digital USFQ: Página De Inicio, Quito, 1 Jan. 1970, repositorio.usfq.edu.ec/handle/23000/7958.

Ngrok:

“Documentation.” Ngrok, ngrok.com/docs#config.

Botfather:

“Bots: An Introduction for Developers.” Telegram APIs, core.telegram.org/bots.

Rasa:

“Introduction to Rasa Open Source.” Rasa, 6 May 2021, rasa.com/docs/rasa/.

Python:

“Welcome to Python.org.” Python.org, www.python.org/.

Sublime:

“The Sophisticated Text Editor for Code, Markup and Prose.” Sublime Text, www.sublimetext.com/.

Git:

Git, git-scm.com/.

Visio:

“Software De Diagramación y Creación De Diagramas De Flujo: Microsoft Visio.”Microsoft, www.microsoft.com/es-es/microsoft-365/visio/flowchart-software.

Índice de imágenes